D3.5 Point cloud schema extension for the IFC model

DURAARK

FP7 – ICT – Digital Preservation
Grant agreement No.: 600908

Date: 2015-07-31
Version 1.0
Document id. : duraark/2015/D.3.5/v1.0

Grant agreement number	:	600908
Project acronym	:	DURAARK
Project full title	:	Durable Architectural Knowledge
Project's website	:	www.duraark.eu
Partners	:	LUH – Gottfried Wilhelm Leibniz Universitaet Hannover (Coordinator) [DE] UBO – Rheinische Friedrich-Wilhelms-Universitaet Bonn [DE] FhA – Fraunhofer Austria Research GmbH [AT] TUE – Technische Universiteit Eindhoven [NL] CITA – Kunstakademiets Arkitektskole [DK] LTU – Lulea Tekniska Universitet [SE] Catenda – Catenda AS [NO]
Project instrument	:	EU FP7 Collaborative Project
Project thematic priority	:	Information and Communication Technologies (ICT) Digital Preservation
Project start date	:	2013-02-01
Project duration	:	36 months
Document number	:	duraark/2015/D.3.5
Title of document	:	Point cloud schema extension for the IFC model
Deliverable type	:	Software prototype
Contractual date of delivery	:	2015-07-31
Actual date of delivery	:	2015-07-31
Lead beneficiary	:	TUE
Author(s) (alphabetic)	:	Jakob Beetz `<j.beetz@tue.nl>` (TUE) Thomas Krijnen `<t.f.krijnen@tue.nl>` (TUE) Raoul Wessel `<wesselr@cs.uni-bonn.de>` (UBO)
Responsible editor(s)	:	Jakob Beetz `<j.beetz@tue.nl>` (TUE)
Quality assessor(s)	:	Sebastian Ochmann `<ochmann@cs.uni-bonn.de>` Michael Panitz `<Michael.Panitz@tib.uni-hannover.de>` Martin Tamke `<Martin.Tamke@kadk.dk>`
Approval of this deliverable	:	Stefan Dietze
Distribution	:	Public
Keywords list	:	Point clouds, Industry Foundation Classes (IFC), Standardization

Executive Summary

In this deliverable, an extension to the IFC schema is proposed that allows the integration of point cloud data into instance models of the Industry Foundation Classes (IFC). The proposed schema extension allows point cloud representations to be stored and presented alongside existing explicit shape representations in the form of parametric or tessellated geometry.
In addition, the implementation of a serialization of IFC instances using the Hierarchical Data Format 5 (HDF5) is proposed according to ISO 10303-26 that allows a much more efficient means to handle and store large quantities of data usually involved in point cloud based workflows. Contrary to predominant serialization formats for IFC and STEP, namely SPF and XML, both being clear text encodings, the HDF5 file format offers a binary data format, indexed random access reading and writing and transparent partial compression.
The combined approach, outlined and implemented in this deliverable, offers a layered compression approach that yields an eventual file size of a combined model smaller than that of industry standard point cloud formats. Furthermore, it adds random access reading and writing capabilities not offered by prevalent clear-text IFC encoding formats.
The deliverable outlines promising paths to integrate the proposed schema extension and binary serialization into the DURAARK WorkbenchUI by means of *a*) offering a unified format that encapsulates both IFC instance data and point cloud scans, the two principle data types in the DURAARK archival workflow and *b*) a hierarchical binary serialization format that alleviates the current difficulties in streaming access to IFC and point cloud models for progressive and variable Level Of Detail visualization and retrieval of archive contents.

Table of Contents

1 Introduction

This document describes a unification of structured IFC building element data and unstructured point cloud scans. This is accomplished by means of a schema extension to the IFC standard that offers a semantic association relation and geometric coupling. The schema extension is detailed in Section 3. In addition a binary serialization format for IFC and associated point clouds is presented and outlined in Section 4. The sections that follow here describe background information and the rationale for the proposed unification. The conceptualization in the DURAARK project, adherence to use cases and integration into other development efforts is provided in Sections 1.5.2 and 1.6.

1.1 The contrasting nature of IFC and point clouds

The nature of the two data types are disparate in various aspects. Some key differentiating qualities are outlined below.

Semantics and structure In IFC, a building is described as a semantically rich assembly of building elements, which can be represented by various forms of geometry. More so, representations are merely one of the aspects conveyed in an IFC model. In addition, attributes related to performance, costing, use, etc. can be added.

On the other end, in many common data formats for the storage of point cloud data, such as E57 [1], PCD [19], metadata attachment is limited to the level of individual data sets. This metadata for example includes scanner positions or weather conditions that allude to aspects of the perceived physical world. In general however, the point data itself contains no grouping, decomposition or other information that relates the points to the semantic meaning of the real-world object that was scanned.

Data volume The magnitude of the data, which is typically found in point cloud data sets and IFC model populations, can be dramatically different for the two file types. A meaningful IFC file can have file sizes in the order of a few megabytes, but, depending on the amount of detail and precision, point cloud scans can easily amount to gigabytes of data.

Encoding Prevalent encoding mechanisms for IFC data typically follow ISO 10303-21 or ISO 10303-28, which offer a human readable clear-text encoding. Point cloud formats, such as E57 [1], are only available in a binary format, or sometimes offer both a binary and ASCII encoding of the data [19].

1.2 Scientific interest in the superposition of IFC and point clouds

Despite the fundamental differences in the two data types, there is a growing interest in the use of point cloud data in the architecture engineering and construction industry. Point clouds are a viable tool to precisely document a structure as it is physically built, rather than as it has been designed, and to detect differences between these "as-planned" and "as-built" states is interesting for many use cases. Superimposing, reconstructing, comparing and integrating "as-planned" and "as-built" models of buildings and other structures has been the subject of many research and development efforts [4, 18]. Practical use cases include the monitoring of construction processes [8, 24, 23, 6] or indoor navigation [16]. The increasing availability of affordable acquisition technology such as aerial and terrestrial laser scans [18, 22], image based approaches [4] or combinations [7] as well as the necessary processing and storage resources have contributed to this spike in interest amongst practitioners and researchers alike [6, 8, 23, 24].
As has been assessed at the outset of the DURAARK project, especially in the domain of digital preservation there is a confluence in the desire to document the design intent and intended use along with the physical edifice as it is delivered and changes over time [10]. Especially historical building often carry intricate details, including detailed plaster works, for which a definition using parametric surfaces is beyond what is typically the scope of the modeling effort in IFC. Such an edifice can be archived using hybrid representation forms, alternating explicit solid representations with annotated subsets of point cloud scans in varying levels of detail. An illustration that highlights superpositioned IFC instance data and point cloud scans, on the dataset used throughout this deliverable, is provided in Figure 1.

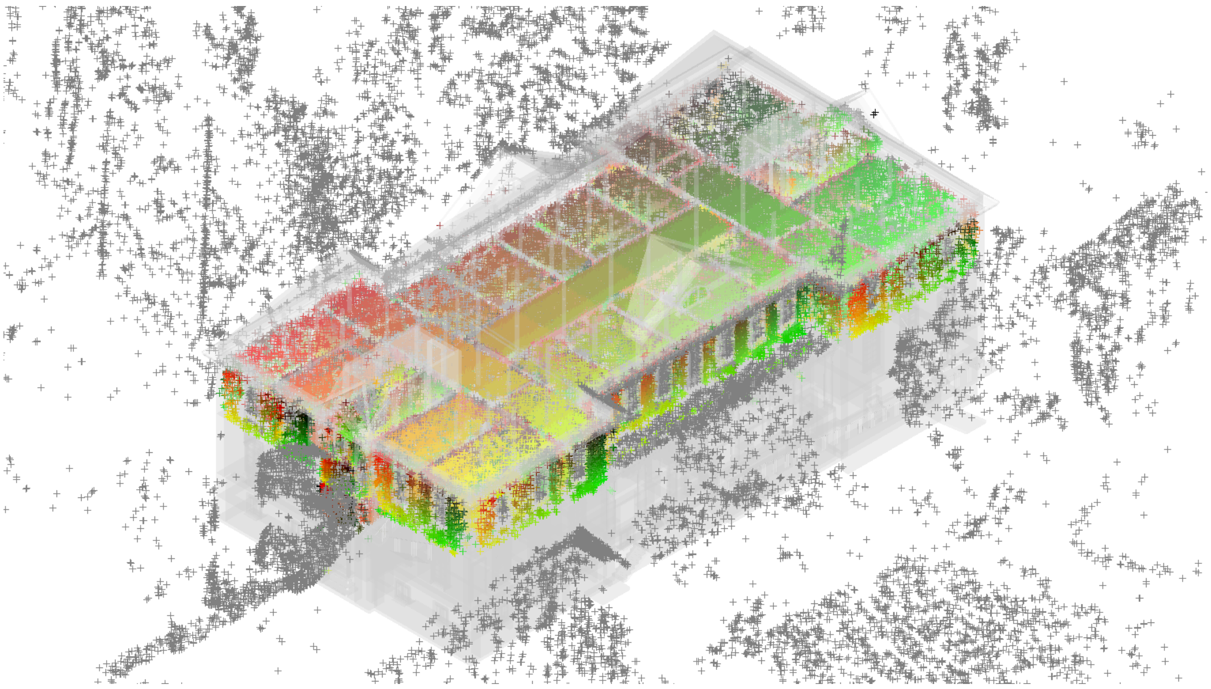


Figure 1: A point cloud geometrically aligned with an IFC file, obtained from the software prototype introduced in this deliverable. The coloring is reduced from the parametrization of the points on the surfaces of their associated building element. The unassociated points, for example representing exterior foliage and reflections from glass panes are colored in gray.

1.3 Benefits for point cloud visualization

Except for very small point clouds, their visualization requires the usage of certain acceleration structures like Octrees [17] or k-d-trees [2], as the entire amount of data is not manageable even by modern graphics hardware. Especially when points are browsed online, for example in a digital preservation system, the streaming nature of data transmission imposes additional constraints on effective clustering in order to conserve bandwidth and reduce load times.

Acceleration structures subdivide point clouds into hierarchical cells, each containing an exclusive subset of the points. During visualization, which points are contained within the observer's frustum can be more efficiently assessed by their bounding cells in the acceleration structure.

As point clouds are inherently unstructured, the traditional acceleration trees are not

tailored semantically to the data, for example in a way that a cell corresponds a room. Instead, the cell borders would rather arbitrarily cut through these semantic structures. This can lead to an overhead of points being rendered despite being invisible because of e.g. being positioned behind a wall that blocks the current view.
Associating points to IFC entity instances as is sketched in Figure 2 would enable to overcome such limitations and offer the possibility to construct semantically-driven and therefore intelligent acceleration structures.

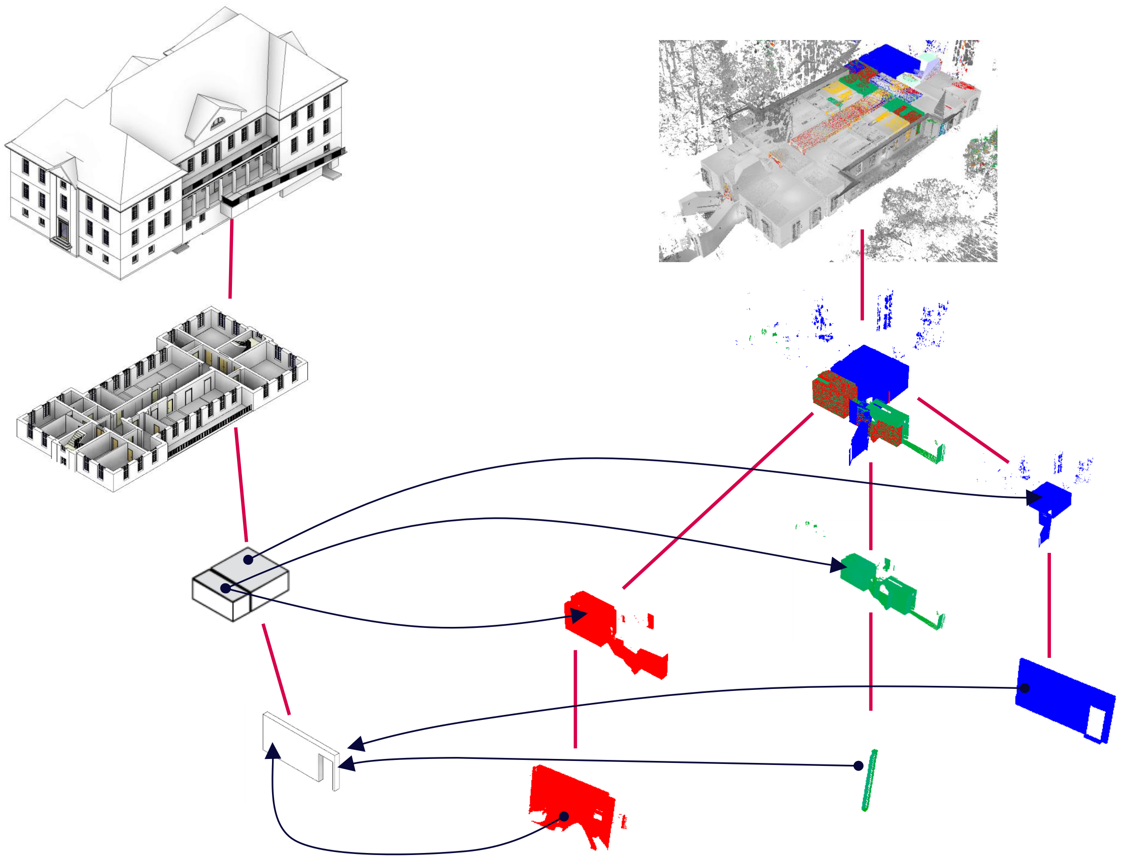


Figure 2: Illustration capturing the decomposition structure and association relationships of a building and aggregated point cloud scan

1.4 Benefits for point cloud compression

With the increasing resolutions of 3D acquisition devices, researchers have investigated possibilities to efficiently compress point clouds. Ideal point cloud compression techniques

would enable *a*) high compression ratios without sacrificing decompression speed *b*) the ability to decompress localized segments without decompressing the entire file *c*) streaming levels of detail. Universal entropy-based methods, like [25], that perform well on data such as text, are less effective for point cloud compression as they do not detect and exploit hidden underlying geometric structures in the unstructured point cloud. Therefore, several methods [5, 15, 20] use an Octree to approximate the actual surface that is underlying the sampled point cloud.
Conversely, the schema extension introduced in this deliverable assumes a point cloud is paired with an IFC file with geometrical representations for its building elements[1]. As such, the proposal in this paper follows a shape-proxy-based compression method, for example as outlined in [11, 13, 21]. The key to an efficient compression is to find shape proxies that explain a large amount of measured points in order to store the points not as triplets of $\{x, y, z\}$ coordinates, but instead projected onto the shape proxies. Furthermore, when discretized into a height field, the dimensionality of a point is reduced to only a single component that specifies deviation from the surface at a specific grid point. The assumption is that representations of IFC entities will yield better localized surfaces than structures that are estimated from the point cloud data itself, hence yielding higher compression ratios for the points that conform to building element surfaces. More traditional point cloud compression and surface fitting techniques can be employed for the points that do not conform.

1.5 Addressed Use Cases

1.5.1 Use cases defined in the scope of the DURAARK project

In the DURAARK deliverable D2.1, use cases related to *core long-term preservation* tasks have been defined as:

- UC1: Deposit 3D architectural objects
- UC2: Search and retrieve archived objects

[1]Harmonizing the two data formats is the primary objective of this deliverable, but in case an "as-designed" IFC model is not available for a point cloud model, within DURAARK deliverable D5.3 a tool is described that can extract elementary surfaces from point cloud models and serialize them as semantically rich IFC building components. As such, users can benefit from the compression and enrichment methodologies described in this deliverable also when considering a point cloud model in isolation, as the IFC file can be generated from it.

- UC3: Maintain Semantic Digital Archive

In addition, use cases related to *production and consumption-oriented* tasks are identified to be:

- UC4: Detect differences between planning state and as-built state
- UC5: Monitor the evolution of a structure over time
- UC6: Identify similar objects within a point-cloud scan
- UC7: Plan, document and verify retrofitting/energy renovations
- UC8: Exploit contextual information for urban planning
- UC9: Enrich BIM/IFC model with metadata from a repository

1.5.2 Use cases addressed

The schema extension and binary serialization format affect and facilitate several of the use cases sketched above.

1. The `deposit of architectural objects (UC1)` is accommodated by an additional option that is introduced to deposit a single unified container that contains both the IFC building model as designed, and one or more point cloud scans that capture the edifice as-built.

2. The `retrieval of archived objects (UC2)` is facilitated as a schema extension is proposed that segments localized subsets of point clouds pertaining to individual building elements. Furthermore, a serialization format is introduced that greatly enhances the extraction of a subset of an IFC graph that spans a request for information from a end-user querying the archival system.

3. The `difference detection (UC4)` use case and the case that addresses the `evolution of a structure over time (UC5)` are accommodated by a unified file format that explicitly documents the semantic relation between building elements and point cloud subsets and facilitates the procedures that are described in more detail in DURAARK deliverable D4.2.

1.6 Integration strategy

This deliverable contributes two new additions to the domain of architectural engineering, archival and facility management. This section outlines how these two additions, an extension to the IFC schema and an efficient binary serialization format for IFC instance models, fit into the scope of the DURAARK project, the WorkbenchUI and the Digital Preservation Service.
The schema extension proposed in this deliverable offers purely an addition to the IFC4 Addendum 1 schema as released by buildingSMART[2]. As such, tools that are part of the DURAARK system and workflows, described in DURAARK deliverable D2.5, can operate on the extended schema without impacting the acceptance and validity of instance files serialized according to tools that have no knowledge of this extended schema.
In addition, the binary serialization format of IFC instance models according to ISO 10303-26, as developed for the software prototype accompanying this deliverable, will be fully understood by the DURAARK components, such as the relevant metadata extraction tool[3]. This future work will greatly reduce the time needed for the metadata extraction process to complete on large instance models, as the task of extracting instance attributes identified by fixed paths (DURAARK deliverable D3.3 Section 4.4) is exactly something that is greatly facilitated by the hierarchical nature of HDF5, outlined in Section 4.2.
Many components of the workbench are left unaffected by this new introduction, as for example the file identification within the SIP generator, which is based on the PRONOM registry, as described in D6.1, already has access to signatures for HDF5 by means of the DROID signature files[4].

[2] http://www.buildingsmart-tech.org/specifications/ifc-releases/ifc4-add1-release
[3] https://github.com/DURAARK/pyIfcExtract
[4] https://www.nationalarchives.gov.uk/aboutapps/pronom/droid-signature-files.htm

2 Prototypical implementation and documentation

For this deliverable a prototypical implementation is presented that produces and consumes IFC building models enriched with point cloud data. The module is available on-line at: `https://github.com/DURAARK/IFCPointCloud` and consists of four main components.

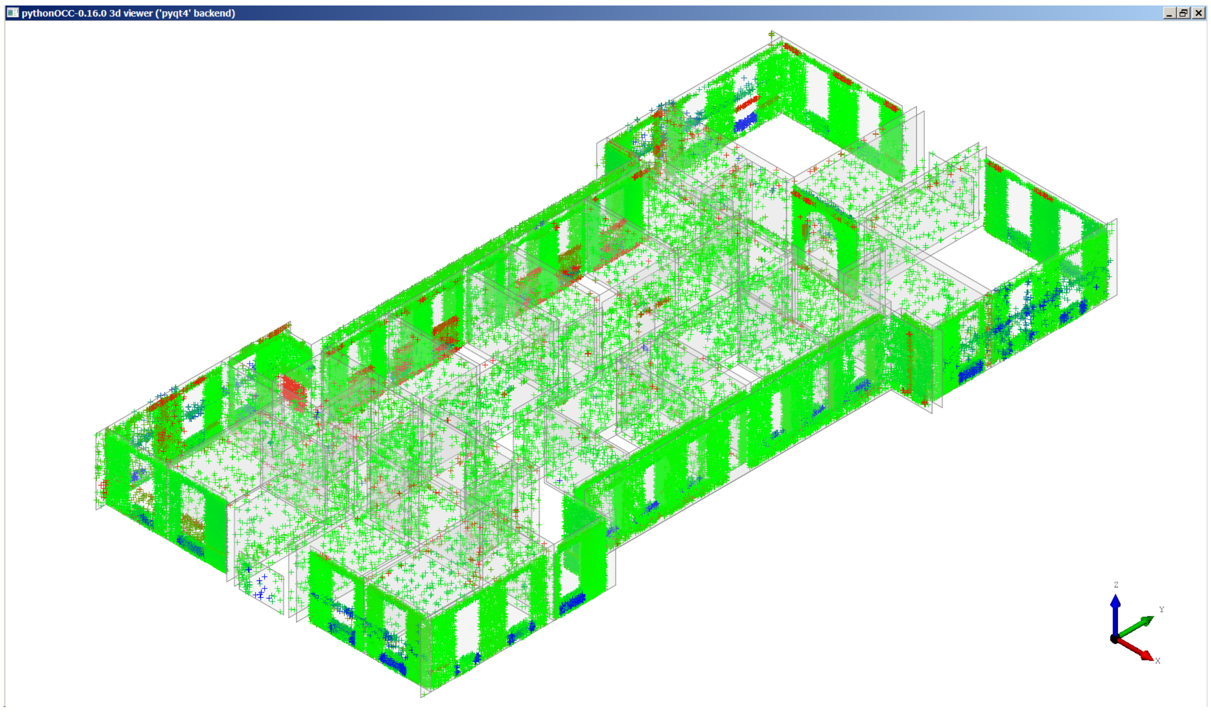


Figure 3: A screen-shot of the viewer application, part of the prototype described below. In this particular case, the point cloud visualization is limited to walls contained in a certain building storey. Furthermore, a different subsampling is applied based on the type of the wall the point cloud subset is associated to.

1. `Point cloud association` (point_association.py)

   Input:

   IFC file SPF-serialized IFC instance model that describes the building element data.

   Point cloud scans A filename pattern that points to a set of gzip-compressed ASCII-encoded .pcd files.

Alignment matrix An alignment matrix as emitted by the tool described in DURAARK D4.2 Section 3.

Output:

Cached BRep representations For every `IfcProduct` descendant in the IFC file, an Open Cascade BRep file is generated that represents the "Body" representation of the product in global model coordinates.

Associated points A binary sequence of the Cartesian coordinates (as transformed by the alignment matrix) for every point in the .pcd input files. If the point is close to an IFC building element surface, the Cartesian coordinates are followed by *a*) a reference to the product *b*) an index that points to a face in the Cached BRep representations *c*) the parametric coordinates and deviation of the point as projected onto the surface. The points are subdivided over several passes so that a contiguous subset of the points yields a randomly subsampled subset.

2. `IFC SPF generation` (write_spf_file.py)

Input:

IFC file SPF-serialized IFC instance model that describes the building element data.

Cached BRep representations A reference to the geometry cache generated earlier.

Associated points The sequence of associated points as generated above.

Output:

IFC file An SPF-serialized IFC instance model that, in addition to the original data, contains point cloud data and association relations conforming to the schema extension as outlined in Section 3.

3. `HDF5 serialization` (hdf5_serialization.py)

Input:

IFC file SPF-serialized IFC instance model that describes the building element data.

IFC schema The EXPRESS (ISO 10303-11) rendition of the complete IFC model.

Output:

IFC file An HDF5-serialized version of the input file, generated according to Section 4.

4. `IFC and point cloud viewer` (viewer.py)

Input:

IFC file An SPF- or HDF- serialized IFC file with or without point cloud data. Configuration of the viewer, i.e what to visualize and in what level of detail happens in-code.

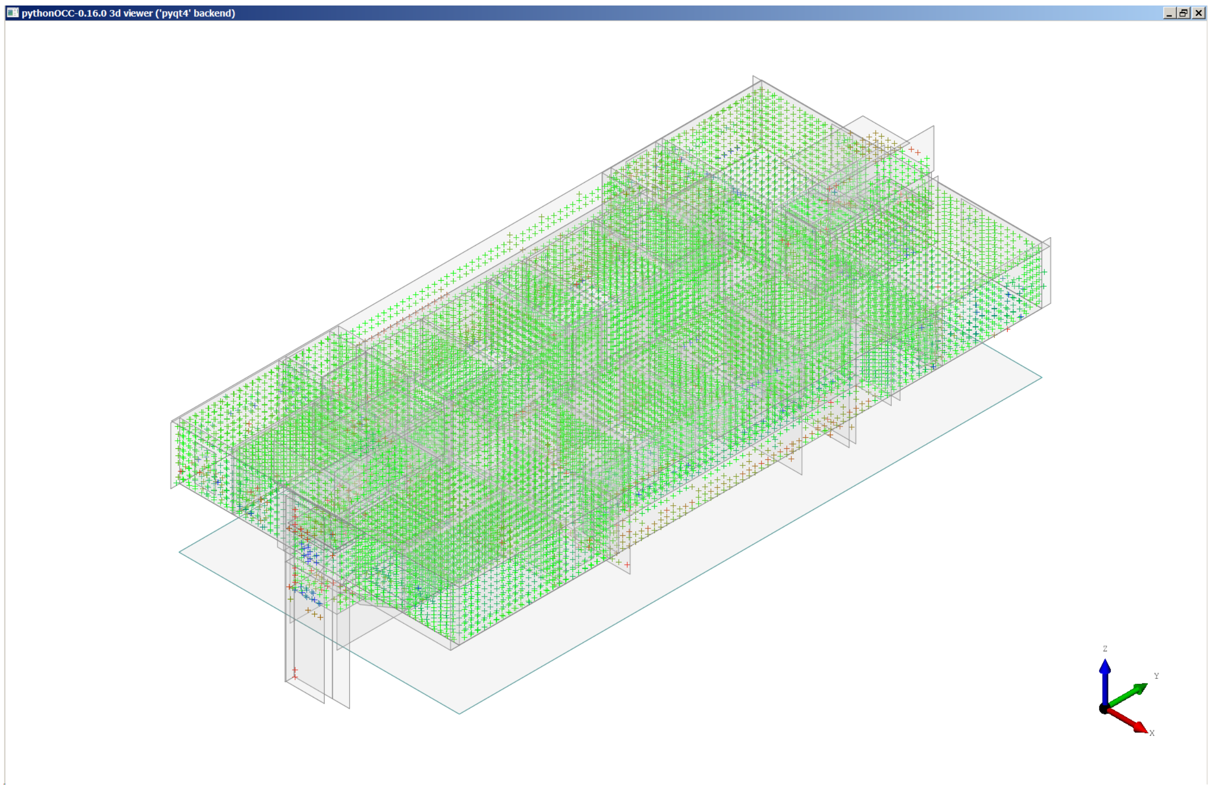


Figure 4: A screen-shot of the viewer application, part of the prototype described above. In this particular case a very coarse grid-discretized rendition of the point cloud is selected and shown on top of the parametrization surfaces.

3 IFC Schema extension

3.1 Existing data structures

In this section a brief overview of the existing means to represent objects in an IFC instance model is provided. The current data types and information structures to capture point cloud information are introduced and their shortcomings with regard to performance and storage space are discussed.

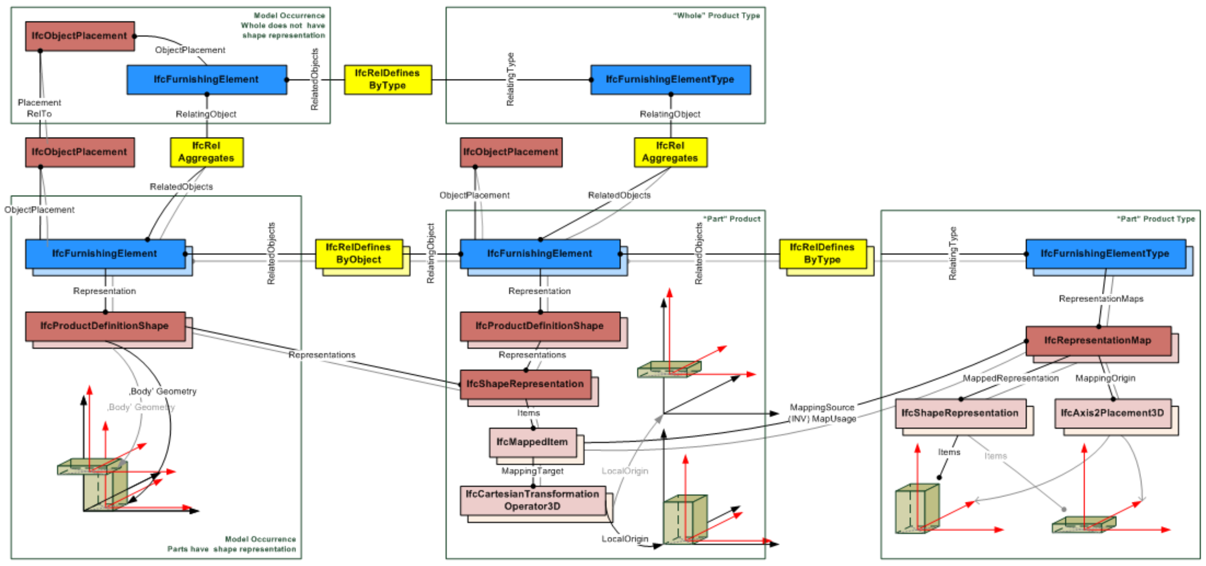


Figure 5: Modeling constructs to represent an `IfcProduct` with geometric or topological items (source: IFC documentation)

Geometric representations in IFC

In general, the approach to include geometric representations of objects (or more specifically `IfcProducts`) has been adapted from ISO/IS 10303-42:1994, and the improved definitions of the second edition, ISO/DIS 10303-42:1999 "Integrated generic resources: Geometric and topological representations". The main compartements of the IFC schema are `IfcTopologyResource` and `IfcGeometryResource` containing the ENTITY and TYPE definitions for geometry (curves, surfaces, placement etc.) and topology (vertex, edge, face, shell etc.) as well as the `IfcRepresentationResource` modeling the associations between the objects and their (2D/3D) representations. The principle mechanisms of these relations have been illustrated in Figure 5.

Depending on the use of parametric or explicit geometry types, the entities defined in these three resources often account for up to 90% of all instances found in Part 21 population STEP Physical Files (SPF)[5].

In order to address the rising needs to capture implicit information in building models coming from bulk measurements such as laser scans in the form of point clouds, the recent version 4 of the IFC schema introduced the new entity `IfcCartesianPointList3D`, see Listing 1, to *"provide a compact representation of larger list of points, such as in point clouds, and in indexable representation of points used as vertices in tessellated items or poly curves."*

```
ENTITY IfcCartesianPointList3D
SUBTYPE OF (IfcCartesianPointList);
   CoordList : LIST [1:?] OF LIST [3:3] OF IfcLengthMeasure;
END_ENTITY;
```

Listing 1: Schema fragment of the IfcCartesionPointList3D entity introduced in IFC 4

The point list is modeled as a subtype of `IfcGeometricRepresentationItem` and allows the use of simple two-dimensional, unbounded arrays (list of list) of x,y,z coordinates wrapping floating point numbers as a `IfcLengthMeasure` values. The compactness mentioned in the description provided by buildingSMART refers to the preexisting method for storing lists of Cartesian points in which every point needs to wrapped in an individual entity instance.

With respect to practical needs for handling point clouds a number of short comings can be identified:

- **Quantity of data and encoding**. Point cloud data sets usually amount to many hundreds of thousands or even millions of individual points. Storing them in clear text encoded SPF that have no way of indexing, would require long processing times. Alternative ways to capture the position of individual point in space, like the height-field approaches introduced in this proposal (3.2.2) can cut down data quantities significantly. Specialized packaging algorithms can additionally be employed to save space and processing time.

[5]The higher the use of parametric shapes vs. explicit long lists of `IfcCartesianPoint`s, polygonal loops, face boundaries and faces in meshes and (faceted) BReps, the smaller the files.

- **Data types**. Beyond the raw x,y,z coordinates point cloud data aquired from lasers scans etc. contain other important information such as reflection normals, colour etc. that is valuable to preserve. Specialized dedicated data formats like E57 account for this through the provision of data structures allowing to capture these accordingly.
- **Decomposition** Measuring complete buildings through laser scanning and other technologies usually requires many individual scans that are then aligned in specialized applications. A single building element such as a wall is often covered by a number of different scans from various locations (i.e. within the same room). On the other hand, a single scan covers a number of different building elements. This make the use of decomposition techniques necessary, which cannot be done with `IfcGeometricRepresentationItem`s alone. This enables the possibility to record both the affiliation with a scanner position and association to a building element side by side in the model.

3.2 Alternative approaches to capturing points

In this section a number of alternative approaches to integrate IFC and point cloud representations are discussed. Plain lists of cartesian points like the `IfcCartesionPointList3D` entity are just one – and often not the most space efficient – way to capture point locations aquired from laser scans. Depending on their precision, they require three complete floating point values for each individual point. Depending on the units set for the respective `IfcLengthMeasure` being used, the required number of bits can be significantly large (to e.g. capture decimal places).
This can be cut down considerably using a characteristic of the overwhelming majority of buildings: The main spatial elements of buildings - walls, floors and ceilings - but also coverings of opening elements such as doors and windows usually are planar objects that are often configured in orthogonal ways.
Augmenting the `IfcCartesionPointList3D` approach, two new data structures harnessing these characteristics of building are introduced in the proposed schema in order to allow efficient and compact storage of large quantities of data.

3.2.1 Point coordinates in parametric UVW space

In this first optimized approach, boundary geometry of existing building product representations can be used as a reference surface. For this purpose, each point with its

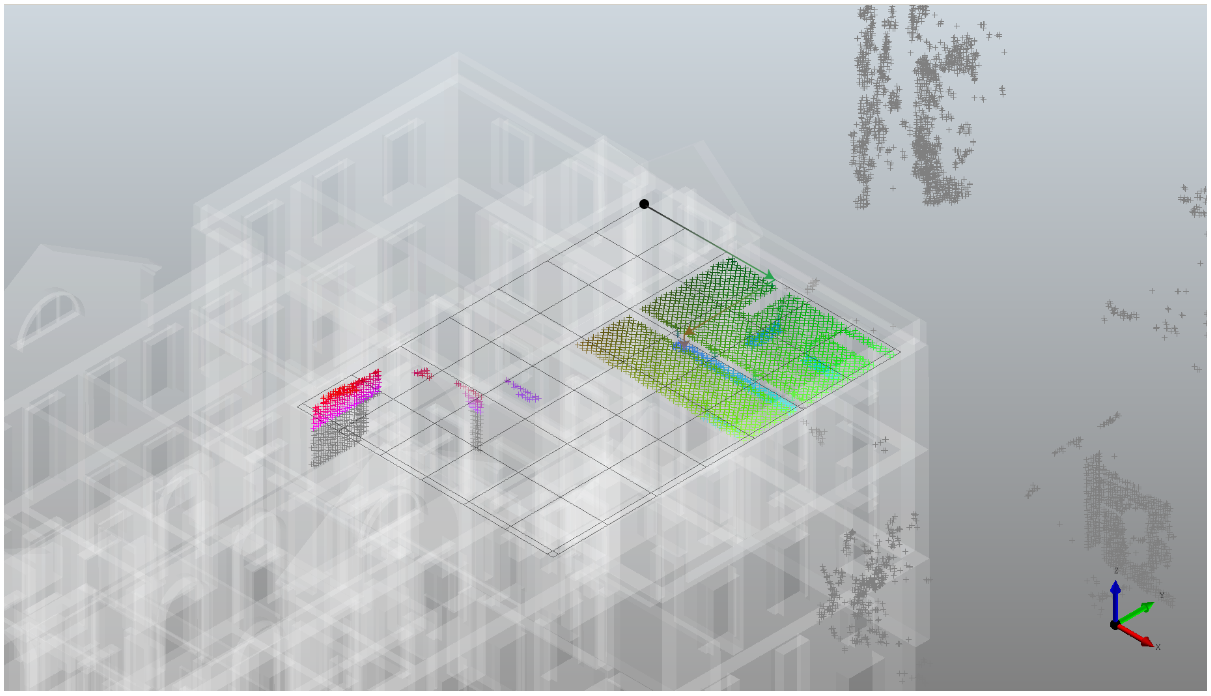


Figure 6: Illustration of mapping points from Cartesian space into parametric U,V,W space. Colors represent the respective UV coordinates in the parametric space represented by the superimposed grid structure

Cartesian coordinates $\{x,\ y,\ z\}$ is projected onto the parametric space of the reference surface consisting of $\{u,\ v\}$ for the parametric coordinates and w for the deviation from the surface along the surface normal evaluated at $\{u,\ v\}$. In contrast to global or local coordinates in geometric/Cartesian space, points in parametric space can be stored with less bits due to the smaller and known domain ranges that have to be described[6]. This is illustrated in Figure 6 by showing how a reference plane coplanar with the ceiling of the Haus 30 dataset can be used as a parametric coordinate system to capture points from different individual scans.

Note that the proposed approach does not only work on planar faces, but can also describe the parametrization of points along curved and double curved surfaces. In ideal cases these surfaces are modeled as parametric surfaces from which particular sub-domain ranges e.g. in a single room can be derived. Faceted approximations of curved surfaces can also be

[6]Note however that such an advantage mainly applies to binary serializations of IFC data, as the EXPRESS modeling language offers no mechanism to denote the precision of underlying data structures to store REAL-valued attributes.

used but are less efficient due to the implied loss of precision and overhead in individual data structures.
Apart from the potential reduction in storage space stemming from the decomposition of the Cartesian coordinate space in bounded parametric spaces, this approach of storing point clouds offers an important semantic advantage over the former approach, as the deviation from building surfaces is explicitly recorded as one of the coordinate components. As such, cardinal metrics pertaining to the association of point cloud and building product data, for example an average indication of the overall deviation between the two representations, can efficiently be computed.
The technical implementation of the respective data structure into the IFC data model in the proposed schema is detailed in section 3.3.4.

3.2.2 Heightmaps using discreet spaces[7]

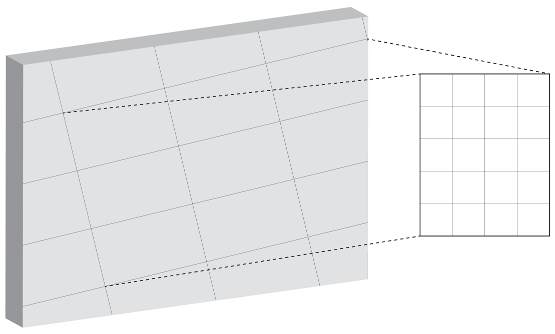


Figure 7: Principal illustration of a discreet grid projected onto a building element in an IFC file to be used for capturing point cloud data as heightmaps

A second approach for the storage of points as an alternative to Cartesian points makes use of a discreet space similar to pixel in bitmap images. Points irregularly scattered on a reference surface are clustered together and averaged for a chosen cell size using algorithms left to the implementing application which yield different losses in precision with respect to the original scanner raw data. Using mechanisms to define and describe these grids on the targeted surface by reusing the `IfcGrid` data structure the cells / pixels are can be defined in a two dimensional array of grid axis intersections. Sampling points

[7]This allows only for a lossy storage of points

from the continuous space into discreet space necessarily comes with a loss of precision. Depending on the grid cell size and sampling methods however, the requirements of many use cases in digital preservation scenarios would still be satisfiable. For example, grids do not have to be regularly spaced or rectangular, but can be radial grids aligned to match the specifications of the laser scanner. Therefore, depending on the detail such methods can be classified as pseudo lossless. A more detailed evaluation of precision in relation to various use cases will be part of the D7.4 deliverable. The advantage of using such heightmap techniques is the applicability of sophisticated compression algorithms developed for bitmap images such a JPEG2000 that allow the significant reduction in file sizes. This of course is only possible where a suitable serialization such as the HDF5 mapping of STEP files described later in the document. A bitmap representation of such a heightmap is provided in Figure 8. Figure 7 illustrates the basic principle of superimposing a grid structure on an arbitrary surface as a part of a building element representation within an IFC file.

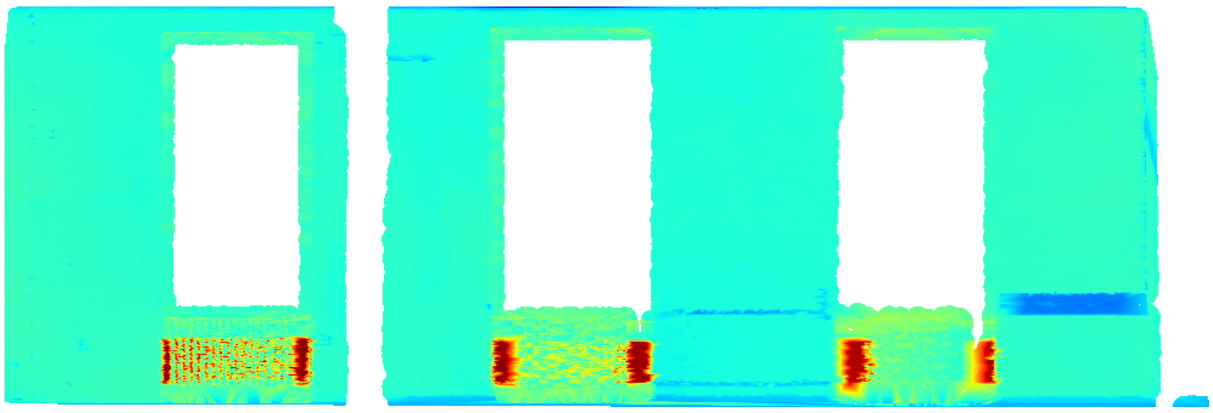


Figure 8: A color coded 2d presentation of point cloud converted into a heightmap project onto a building element surface. The red parts of the image correspond to areas where points extend out of the building element surface, blue parts inwards.

3.2.3 Discretization of floating-point numerals

Current predominant indoor and outdoor laser scanners can only capture points up to limited precision. DURAARK deliverable D7.1 states the average point distance error to be 2.6mm. Yet, the precision offered by a 32bit or 64bit IEEE 754 floating point number offers roughly 6 respectively 12 significant digits, yielding an implied precision way beyond what can be captured by the scanning device. The EXPRESS modeling language itself does not offer means to specify the precision or width of numbers, but typical p21 serializations of IFC files use up to 15 significant digits for the clear text encoding of REAL valued attributes[8][9].

As such, a precision would be implied that is not realized by the scanning device and space is unnecessarily occupied by meaningless trailing digits. Therefore, to exploit the knowledge on scanning precision[10] and knowledge on the bounded domain of parametrized coordinate components, an integer numerator approach is presented that allows to capture bounded numbers using fewer bytes in clear text and as well as binary serialized files, without sacrificing readability.

A standard floating point number with arbitrary range is represented as $(-s) \times d \times \beta^e$. In IEEE 754 double precision this requires 23 bits for the significand d 8 bits for the exponent e and 1 signbit s. With a known and bounded range, the same number can be represented as $d/(2^n - 1) \times (v_{max} - v_{min}) + v_{min}$ with $d < 2^n$ and n the number of bits for the integer numeral. n, v_{max} and v_{min} are specified only once for a complete range of values, i.e typically a localized subset of the point cloud. For efficient binary storage the number of bits n can conform to typical widths of unsigned integer numerals, such as 8 for an `UNSIGNED BYTE` and 16 for an `UNSIGNED SHORT`[11]. An example with an indicative compression ratio for clear text serialization is provided in Listing 2.

```
          1         2         3         4         5         6         7
0123456789012345678901234567890123456789012345678901234567890123456789012345678

((0.8482145585275755,0.4089384818729891,0.8027061702482456,0.11449717768247669))

((55587,26799,52605,7503),65536)
```

Listing 2: An example of floating point discretization yielding a compression ratio of $(79 - 31)/79 = 61\%$ with the denominator included.

The parametric components `{u, v}` are bounded by the `[U1, U2]` and `[V1, V2]` ranges defined in the `IfcRectangularTrimmedSurface` which defines the association region in parametric range of the underlying building element surface. The `w` component is bounded

[8]Naturally, in a clear text encoding, this depends on the nature of the numbers, modular architectural length measures or components of orthogonal directions can be expressed succinctly as for example `120.` or `0.`

[9]The `IfcGeometricRepresentationContext` to limit the tolerance of geometric operations, but this is typically not reflected in the length of the number representations.

[10]Work still remains on capturing scanner meta-data in predefined `IfcPropertySet` instances.

[11]or `H5T_STD_U8LE` and `H5T_STD_U16LE` for little-endian equivalents in HDF5 datasets

by the range specified on the `IfcParameterValueList` and `IfcGridOffsetList` respectively, as can be seen in Sections 3.3.4 and 3.3.5 respectively.

3.3 Proposed schema extension

In this section the proposed extension to the schema is introduced an its use is illustrate with examples
As illustrated in Figure 9 the schema extension only needs to add a small number of new `ENTITY` and `TYPE` definition in order to allow the efficient description of large and complex building models including their point cloud representations. A complete listing of the proposed schema extending the current version of the IFC specification IFC4_ADD1 is provided in Appendix A.

3.3.1 Schema overview

Two main entity definitions build the structural backbone of the schema: `IfcPointCloud` and `IfcPointCloudElement`.

3.3.2 `IfcPointCloud`

```
ENTITY IfcPointCloud
 SUBTYPE OF (IfcGeometricRepresentationItem);
   LevelOfDetail : OPTIONAL IfcLabel;
   Coordinates   : IfcPointCloudCoordinateSelect;
   Attributes    : LIST [0:?] OF IfcPointCloudAttribute;
END_ENTITY;
```

Listing 3: Schema fragement showing the definition of the `IfcPointCloud` entity definition

The entity `IfcPointCloud`, modeled as a subtype of `IfcGeometricRepresentationItem` can be attached to any `IfcProduct` using the regular representation paths illustrated in Figure 5. It contains the following attributes:

`LevelOfDetail`: In order to allow the construction of increasing Levels of Detail (LOD), that let users toggle between various resolutions based on their purpose, the attribute `LevelOfDetail` allows tagging the point cloud with a user defined string. Examples could be "coarse", "LOD 1", "subsampled 0.1" etc.

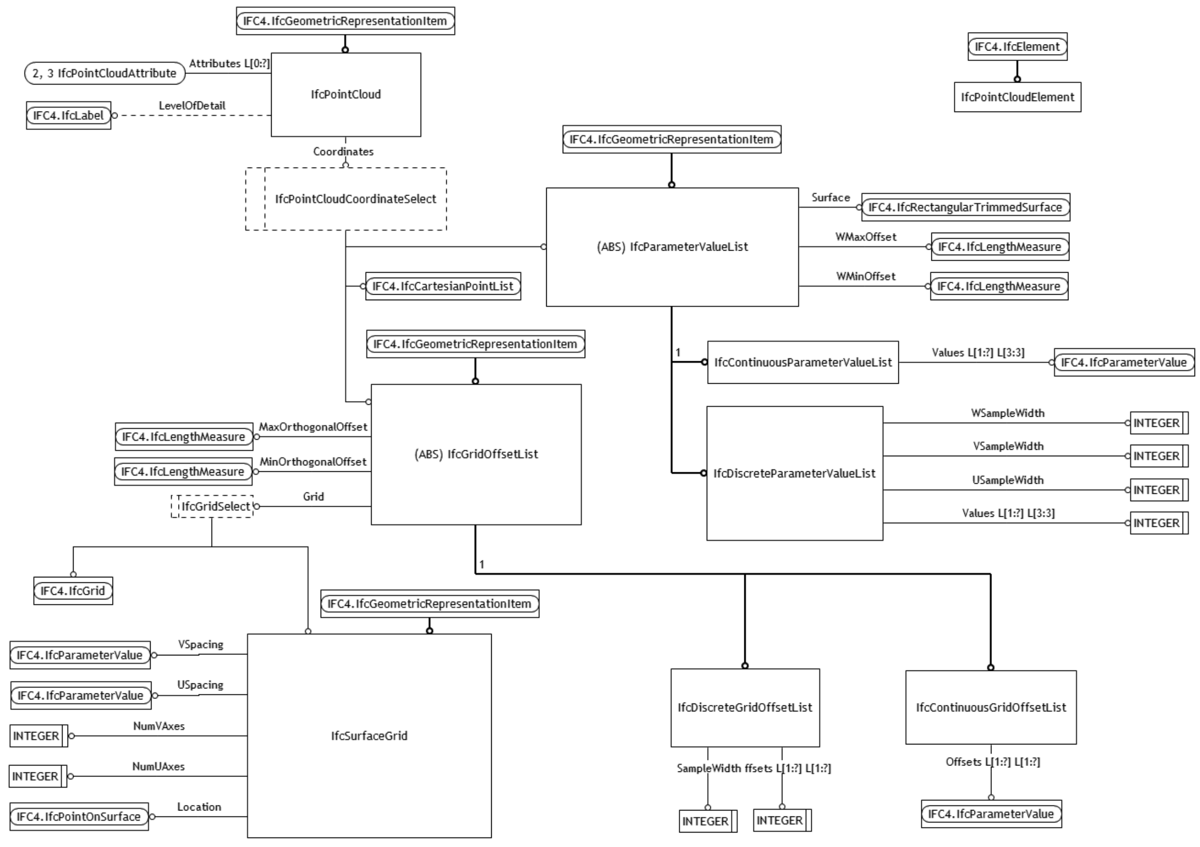


Figure 9: EXPRESS-G diagram of the proposed schema extension

`Coordinates` of type `IfcPointCloudCoordinateSelect` allows the description of point clouds using one of the proposed modeling mechanisms described in Section 3.2 and detailed later in this section.

`Attributes` allows the association of colors and reflection normals to the individual points, where every element in the list of coordinates corresponds to an element in an individual attribute. The variety of attributes that can be expressed, i.e. colors and reflection normals, are represented visually in Figure 10.

3.3.3 `IfcPointCloudElement`

An `IfcPointCloudElement`, see Listing 4, is an entity that can contain subsets of point clouds that cannot be otherwise associated to building elements or spatial subdivision

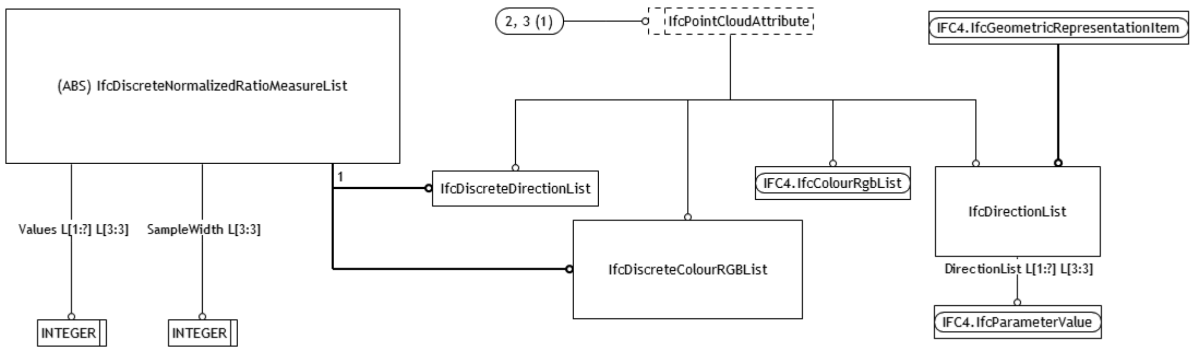


Figure 10: EXPRESS-G diagram of the proposed schema extension outlining the optional the attribute vector elements

structures. This can include, for example, partially covered external foliage and reflections from windows. The `IfcPointCloudElement` can also be used to present a decomposition relation for example to record how a complete point cloud scan is aggregated over scans at individual scanner positions, as is illustrated in Figure 2.

```
ENTITY IfcPointCloudELement
    SUBTYPE OF (IfcElement);
END_ENTITY;
```

Listing 4: Schema fragement showing the definition of the `IfcPointCloudElement` entity definition

3.3.4 IfcParameterValueList

The `IfcParameterValueList`, provided in Listing 5, allows storing point cloud coordinates in the parametric space of building element surfaces, as described in Section 3.2.1. The construct comes in two variants, modeled as subtypes, that represent the coordinate components either as a REAL-valued aggregate or using the integer discretization approached outlined in Section 3.2.3. `WMinOffset` and `WMaxOffset` bound the parametrized range of the `w` component of the supplied `Values` in order for the discretization to be possible. In addition, the range provides an indication to client applications that want to selectively load point cloud subsets based on their deviation without iterating over the entire list of points.

```
ENTITY IfcParameterValueList
 ABSTRACT SUPERTYPE OF (ONEOF(IfcContinuousParameterValueList,
     IfcDiscreteParameterValueList))
 SUBTYPE OF (IfcGeometricRepresentationItem);
    WMinOffset   : IfcLengthMeasure;
    WMaxOffset   : IfcLengthMeasure;
    Surface      : IfcRectangularTrimmedSurface;
END_ENTITY;

ENTITY IfcContinuousParameterValueList
 SUBTYPE OF (IfcParameterValueList);
    Values       : LIST [1:?] OF LIST [3:3] OF IfcParameterValue;
END_ENTITY;

ENTITY IfcDiscreteParameterValueList
 SUBTYPE OF (IfcParameterValueList);
    Values       : LIST [1:?] OF LIST [3:3] OF INTEGER;
    USampleWidth : INTEGER;
    VSampleWidth : INTEGER;
    WSampleWidth : INTEGER;
END_ENTITY;
```

Listing 5: Schema fragement showing the definition of the `IfcParameterValueList` entity definition

3.3.5 IfcGridOffsetList

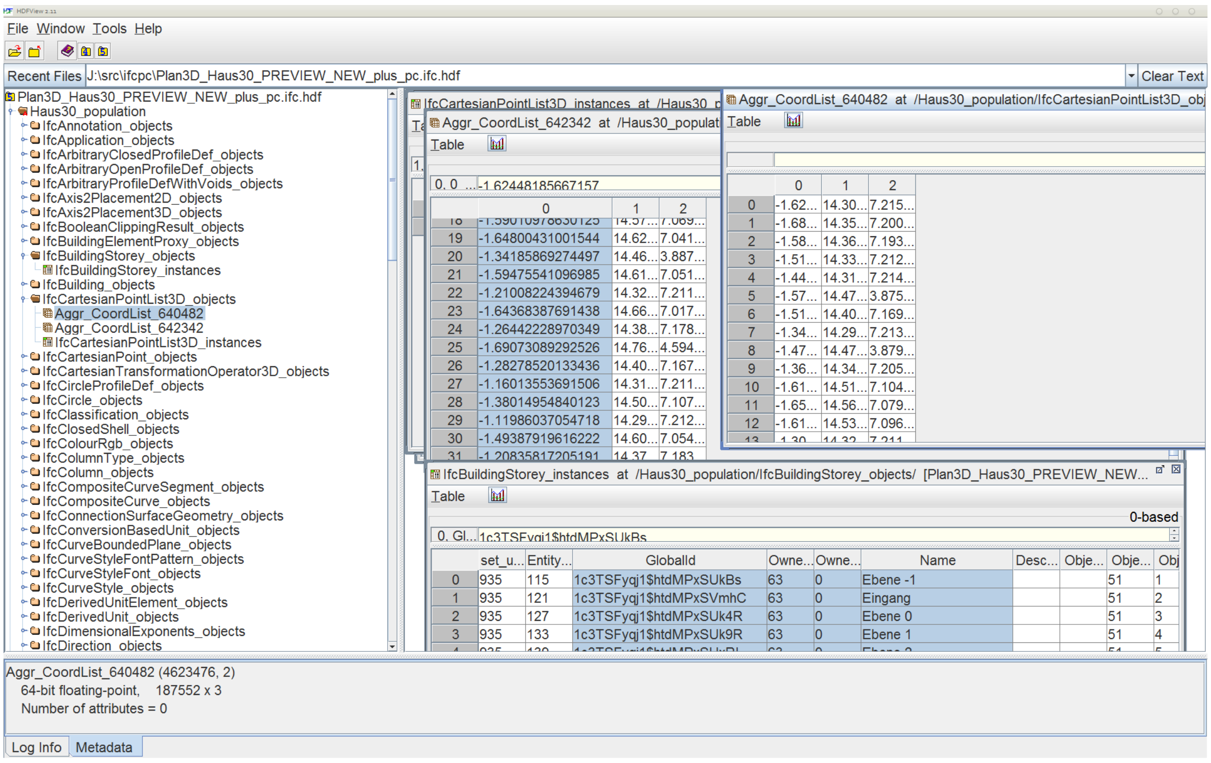
The `IfcGridOffsetList` is a data structure to capture point cloud coordinates as heightmaps projected onto building element surfaces, as described in Section 3.2.2. Its EXPRESS definition is provided in Listing 6. The concept of parametrization is analogous to the construct described above, but only a single `w` component is stored per-point, rather than the `{u, v, w}` components for the latter. The `Grid` attribute refers to `IfcGridSelect` which allows the modeling application to choose from a legacy `IfcGrid`, which for this purpose defines grid axes as `IfcPCurve`s on the association surface, or a newly introduced `IfcSurfaceGrid`, which removes the burden and overhead of defining grid axes individually and provides a uniformly spaced rectangular grid in parametrized surface coordinates. The entire schema extension is provided for reference in Appendix A.

```
ENTITY IfcGridOffsetList
 ABSTRACT SUPERTYPE OF (ONEOF (IfcContinuousGridOffsetList,
     IfcDiscreetGridOffsetList))
 SUBTYPE OF (IfcGeometricRepresentationItem);
    Grid               : IfcGridSelect;
    MinOrthogonalOffset : IfcLengthMeasure;
    MaxOrthogonalOffset : IfcLengthMeasure;
END_ENTITY;

ENTITY IfcContinuousGridOffsetList
 SUBTYPE OF (IfcGridOffsetList);
    Offsets             : LIST [1:?] OF LIST [1:?] OF IfcParameterValue;
END_ENTITY;

ENTITY IfcDiscreteGridOffsetList
 SUBTYPE OF (IfcGridOffsetList);
    Offsets             : LIST [1:?] OF LIST [1:?] OF INTEGER;
    SampleWidth         : INTEGER;
END_ENTITY;
```

Listing 6: Schema fragement showing the definition of the `IfcGridOffsetList` entity definition

4 HDF 5 serialization

Embedding point cloud data into IFC files necessarily brings about large model sizes and large aggregate valued entity instances. In this light, this section highlights some of the shortcomings of the most widely used way to serialize STEP instance models, in the form of the clear-text STEP Physical File Format (SPF), formalized as Part 21 of the ISO 10303 standard. Many of these limitations are also manifested independently from unified point cloud datasets, for example when lazily extracting subsets from IFC instance models for progressive streaming partial visualization.
Therefore, as an alternative, the use of ISO 10303 Part 26 is suggested and compared. ISO 10303-26 is currently not one of the mandated serialization formats of IFC, but offers benefits as instance models grow in size, a natural side effect of embedding point cloud data.



J:\src\ifcpc\Plan3D_Haus30_PREVIEW_NEW_plus_pc.ifc.hdf

```
Plan3D_Haus30_PREVIEW_NEW_plus_pc.ifc.hdf
  Haus30_population
    IfcAnnotation_objects
    IfcApplication_objects
    IfcArbitraryClosedProfileDef_objects
    IfcArbitraryOpenProfileDef_objects
    IfcArbitraryProfileDefWithVoids_objects
    IfcAxis2Placement2D_objects
    IfcAxis2Placement3D_objects
    IfcBooleanClippingResult_objects
    IfcBuildingElementProxy_objects
    IfcBuildingStorey_objects
      IfcBuildingStorey_instances
    IfcBuilding_objects
    IfcCartesianPointList3D_objects
      Aggr_CoordList_640482
      Aggr_CoordList_642342
      IfcCartesianPointList3D_instances
    IfcCartesianPoint_objects
    IfcCartesianTransformationOperator3D_objects
    IfcCircleProfileDef_objects
    IfcCircle_objects
    IfcClassification_objects
    IfcClosedShell_objects
    IfcColourRgb_objects
    IfcColumnType_objects
    IfcColumn_objects
    IfcCompositeCurveSegment_objects
    IfcCompositeCurve_objects
    IfcConnectionSurfaceGeometry_objects
    IfcConversionBasedUnit_objects
    IfcCurveBoundedPlane_objects
    IfcCurveStyleFontPattern_objects
    IfcCurveStyleFont_objects
    IfcCurveStyle_objects
    IfcDerivedUnitElement_objects
    IfcDerivedUnit_objects
    IfcDimensionalExponents_objects
    IfcDirection_objects
```

IfcCartesianPointList3D_instances at /Haus30_p

Aggr_CoordList_642342 at /Haus30_populat

0, 0 ... -1.62448185667157

	0	1	2
19	-1.64800431001544	14.62...	7.041...
20	-1.34185869274497	14.46...	3.887...
21	-1.59475541096985	14.61...	7.051...
22	-1.21008224394679	14.32...	7.211...
23	-1.64368387691438	14.66...	7.017...
24	-1.26442228970349	14.38...	7.178...
25	-1.69073089292526	14.76...	4.594...
26	-1.28278520133436	14.40...	7.167...
27	-1.16013553691506	14.31...	7.211...
28	-1.38014954840123	14.50...	7.107...
29	-1.11986037054718	14.29...	7.212...
30	-1.49387919616222	14.60...	7.054...
31	-1.20835817205191	14.37	7.183

Aggr_CoordList_640482 at /Haus30_population/IfcCartesianPointList3D_obj

	0	1	2
0	-1.62...	14.30...	7.215...
1	-1.68...	14.35...	7.200...
2	-1.58...	14.36...	7.193...
3	-1.51...	14.33...	7.212...
4	-1.44...	14.31...	7.214...
5	-1.57...	14.47...	3.875...
6	-1.51...	14.40...	7.169...
7	-1.34...	14.29...	7.213...
8	-1.47...	14.47...	3.879...
9	-1.36...	14.34...	7.205...
10	-1.61...	14.51...	7.104...
11	-1.65...	14.56...	7.079...
12	-1.61...	14.53...	7.096...

IfcBuildingStorey_instances at /Haus30_population/IfcBuildingStorey_objects/ [Plan3D_Haus30_PREVIEW_NEW...

0, Gl... 1c3TSFyqj1$htdMPxSUkBs

	set_u...	Entity...	GlobalId	Owne...	Owne...	Name	Desc...	Obje...	Obje...	Obj
0	935	115	1c3TSFyqj1$htdMPxSUkBs	63	0	Ebene -1			51	1
1	935	121	1c3TSFyqj1$htdMPxSVmhC	63	0	Eingang			51	2
2	935	127	1c3TSFyqj1$htdMPxSUk4R	63	0	Ebene 0			51	3
3	935	133	1c3TSFyqj1$htdMPxSUk9R	63	0	Ebene 1			51	4

```
Aggr_CoordList_640482 (4623476, 2)
   64-bit floating-point,    187552 x 3
   Number of attributes = 0
```

Figure 11: Screen-shot of the general-purpose and freely available[13] HDFView application for browsing HDF5 files, demonstrating the conversion of the "Haus 30" dataset, consisting of an IFC model with explicit geometry and registered point clouds, into the efficient hierarchical structure described in the ISO 10303-26 standard.

4.1 Shortcomings of the clear-text SPF encoding

The following shortcomings are identified. Most notably these impact the `retrieval of archived objects (UC2)`, as described in 1.5.2, because they hinder the efficient streaming retrieval of localized or subsampled subsets of a model.

Parsing complexity of the graph structure An IFC file represents a directed graph with edges for forward and inverse attributes. The serialization of this graph as an unordered set of entity instance statements necessitates the entire file to be parsed in order to reconstruct all edges to a particular node, as instances that have inverse attributes towards that particular node can be defined anywhere in the file. Furthermore, seeking to arbitrary positions in the file is non-trivial due to not being a regular language.

Parsing complexity of attribute values The plain text serialization of floating point numerals are inefficient to restore as native floating point numerals that can be efficiently operated on. Furthermore, a subrange of large aggregate values cannot easily be obtained, as the length of the aggregate, if not dictated by the schema, can only be determined by iterating over the entire list.

File size The SPF file format contains many inefficiencies in terms of file size, for example in terms of the many repeated entity names such as `IFCCARTESIANPOINT` and potentially lengthy byte sequences for entity instance references and numeric attribute values.

4.2 The HDF5 file format

As standardized by ISO 10303-26 a binary serialization format for EXPRESS instance models exists that uses the HDF5 hierarchical data format. HDF5 is robust binary format backed by large institutional users that deal with scientific data sets. The use of HDF5 has also gained notable interest in the digital preservation community [3, 14] and interoperability use cases [9, 12]. Some aspects that make HDF5 a desirable candidate for IFC instance model serializations are listed below.

Hierarchical nature HDF5 is decomposed into several datasets, which are individually indexed by a separate tree structure.

[13] `https://www.hdfgroup.org/products/java/hdfview/`

Self-documenting complex type definitions HDF5 allows data type definitions for compound structures with named attributes.

Transparent specific-purpose compression (Parts of) individual HDF5 datasets can be individually compressed and retrieved without decompressing the entire file. User defined compression algorithms can be supplied, tailored to the structure of the data at hand.

Open standard HDF5 is an open standard, led by a non-profit consortium, with implementations in major programming languages. This and other considerations make HDF5 a suitable component in a long-term digital preservation context. As a file format, it is incorporated in the PRONOM database[14].

4.3 An HDF5 serialization in the light of IFC and point clouds

The nature of the HDF5 file format serialization of IFC according to ISO 10303-26 is a viable approach to address the shortcomings of the prevalent SPF encoding, especially in the light of the proposed schema extension to store point cloud data in IFC that potentially induces large aggregate valued attributes and large data volumes. In particular, the hierarchical nature of HDF5 allows to navigate to the dataset in question in constant time regardless of total file size. If a user is only interested in a subset of the IFC model, contrary to the SPF serialization, in a HDF5 serialized file, only the data of interest has to be read. In addition, the binary nature of HDF5 allows for efficient parsing of the numeric attributes and aggregates, which is of cardinal importance when operating on large volumes of point cloud data. Furthermore, the specific-purpose compression algorithms that can be applied to isolated aggregate-valued attributes enable to reduce storage space requirements below industry standard formats dedicated to point cloud storage, as is described in the following section.

[14] https://www.nationalarchives.gov.uk/aboutapps/pronom/droid-signature-files.htm

5 Results

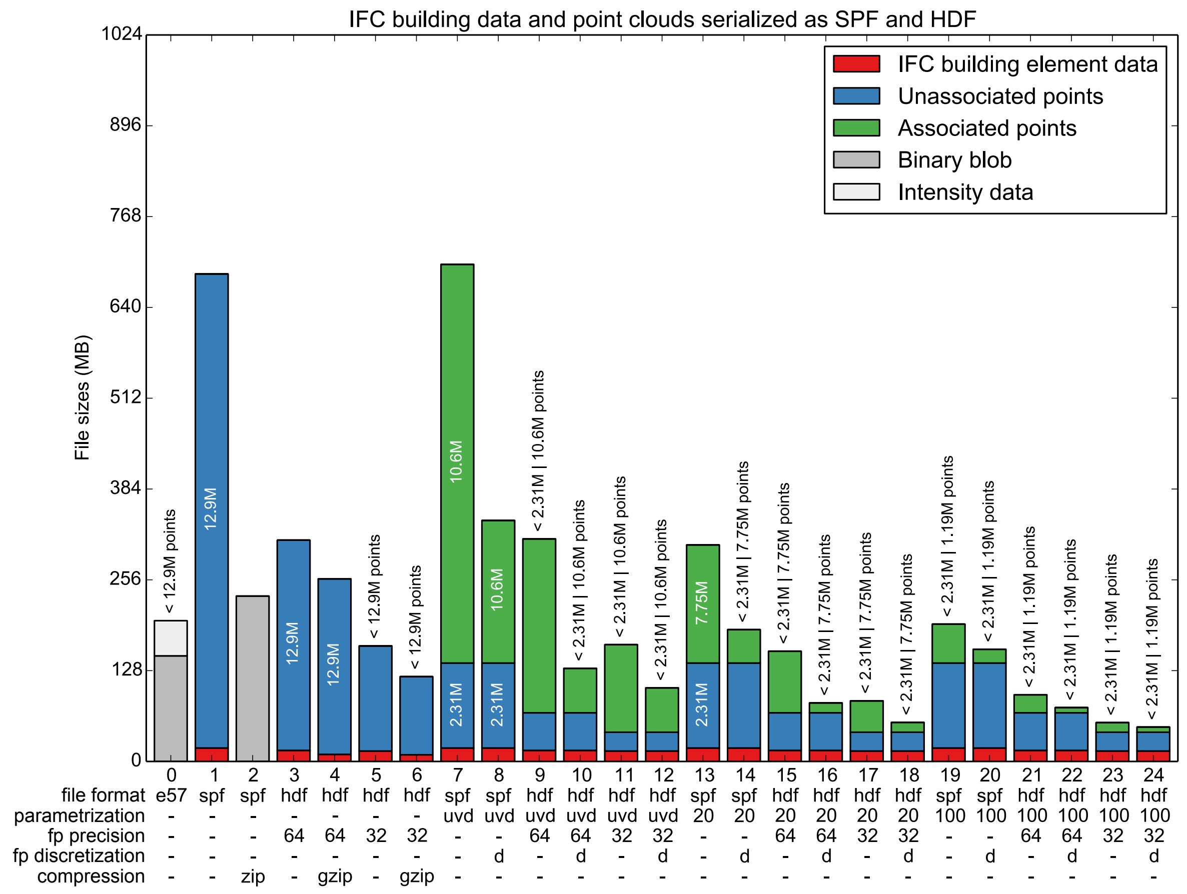


Figure 12: Influence of various serialization options on resulting file sizes and number of points. On the most left there is the e57 file that served as the original input for the point cloud scan. The per-point intensity data has not been incorporated into the conversion process. The total number of points is printed in text for every column. The filesize is represented on the vertical axes, segmented per original IFC building data, unassociated points and associated points (either according to 3.2.1 or 3.2.2).

From the comparison of file size, presented in Figure 12, several observations can be made. Firstly, note that the e57 file (column 0), which has been the source document for the point clouds, is comparable in size to the 32bit uncompressed HDF5 serialization (5) when the non-Cartesian point information is deducted. This is to be expected as both carry the same number of points in the same 32bit floating point format.
In addition, the compressed SPF serialization (2) in byte count, does not deviate much

from the binary HDF file (4). In fact, SPF is not very inefficient with regard to byte count. Three causes that contribute to this observation are identified: *a*) an SPF entity instance reference like "#123" can take more space in HDF as it is described as a compound structure of two unsigned integers, according to ISO 10303-26 6.10.4 *b*) SELECT and OPTIONAL EXPRESS attributes in HDF need to be initialized for all possible concrete implementations, no matter the actual attribute value, as HDF does not allow missing data, as described in ISO 10303-26 6.9.3.4 *c*) on the other hand, the zip compression quite effectively minimizes the otherwise massive overhead of repeated IFC entity names throughout the SPF file, such as the thousands of occurrences of "IFCCARTESIANPOINT". However, note that the compressed SPF file provides an undesirable solution as the file needs to be decompressed in its entirety and offers no mechanisms to retrieve subsets. This however is a feature built-into HDF by means of its hierarchical nature and block-level compression and decompression.
As a third observation, the introduction of associations between point cloud segments and building elements comes with a minimal cost, as association entities need to be populated. This can be seen in cases (3) and (9). However, as discussed in Section 3.2.3, the parametrization onto bounded surfaces allows the use of an integer discretized notation for the parametrized components. Combined with a precision loss under typical scanner's resolution, this yields a tremendous file size reduction in both SPF (7-8) and HDF (9-10). At this point the HDF file (10) is smaller than the original e57 file, carrying the same number of points, enriched with the complete IFC building model instantiation.
An additional layer of compression can be gained from the grid discretization as described in Section 3.2.2. A grid size of 20mm and 100mm is shown in (13-18) and (19-24) respectively. When using the discretized notation for the floating point numbers, even in SPF a low resolution associated point cloud can be embedded in the IFC file of smaller file size than the building data itself, as can be seen in the red and green parts of (20). Such a low resolution (100mm) point cloud totaling to roughly 1.2 million points is probably not useful to assess exact surface properties of the building elements, but might be sufficient for facility management purposes to assess the exact location, rotation and existence of building elements. As mentioned earlier, a more detailed assessment and recommendation in terms of precision in the light of various use cases will be presented in the D7.4 deliverable, which will unify many of the evaluation efforts within the DURAARK project.

6 Technical decisions and risks

6.1 Technical decisions

Schema extension In this deliverable a unification of semantically rich IFC building element data and unstructured point cloud data is presented. This unification is realized by extending the IFC schema with additional entities to represent registered and unregistered points associated to individual building element surfaces. The risks that stem from such an approach, from the perspective of usability, adoption by the buildingSMART consortium and technical feasibility are outlined below.

A schema extension has proven to be the only feasible alternative to realize the desired level of semantic expressiveness in the association of point clouds and building elements. One of the arguments that speak for such unification of file formats is that the serialization of EXPRESS instance models does not allow to arbitrarily point to entity instances from outside of the file[15][16], rendering the option to link to the point cloud by means of an external intermediate file unfeasible.

Binary serialization In addition to the schema extension, a binary serialization format for IFC instance models is proposed, following ISO 10303-26. The binary, hierarchical and self documenting format, described in 4.2, shows clear advantages over the prevalent text-based serialization techniques and guarantees efficient parsing even if the size of models or aggregate-valued attributes grows beyond what is feasible with clear text SPF-encoded models.

Layered compression The layered compression mechanisms, offered by the schema extension and serialization format, provide producing applications the toolset to tailor the precision and resolution of embedded point clouds to the use case at hand. The schema extension follows state of the art point cloud compression techniques (see Section 1.4) and therefore manages to store point cloud models along with explicit building element geometry with file sizes below that of industry standard point cloud formats, as has been presented in Section 5.

[15]For example, the ENTITY INSTANCE NAMES, as defined in ISO 10303-21 §6.3.4, are only to remain unique and unambiguous within a single serialization of the model, any implementation is free to reorder and renumber entity instances.

[16]Subtypes of the ENTITY `IfcRoot` do have a universally unique identifier in the form of their `GlobalId` attribute. However, this only allows to link to entities in the model on a product level, which does not provide the granularity required to point to individual surfaces of a building element

6.2 Risks

Risk Description The schema extension does not cover all use cases from external parties in terms of building management, operation, design and archival.

Risk Assessment

Impact Medium

Probability Medium

Description The schema extension has been conceived from the viewpoint of a semantically rich association of point clouds and building elements. As such, it covers use cases related to digital preservation, facility management, enrichment and the assessment of structural integrity. New scanner hardware might have the possibility to capture new attributes, for which it might not be possible to represent them accurately in the schema extension or other ways of segmenting and association point clouds might be preferred by end-users.

Contingency Solution The attribute vector as part of the `IfcPointCloud` entity allows possible attribute values to be easily extended in subsequent releases of the schema in a backwards compatible manner. The association and decomposition relationships are permissively structured and not mandatory.

Risk Description The schema extension does not cover all use cases from external parties in terms of efficient point cloud visualization.

Risk Assessment

Impact Low

Probability High

Description From the perspective of computer graphics, the proposal offers means to write state-of-the-art point cloud compression algorithms in a structured and standardized format. Yet, for further algorithmic developments entities to represent them might be lacking.

Contingency Solution Additional structures can be implemented in the schema that model more traditional spatial acceleration mechanisms to further the efficient localized retrieval and visualization of point clouds. In the first place, the schema extension is conceived with the building sector in mind, hence the use of the IFC file format.

Risk Description The implementation effort or computational complexity is too high for other software vendors to adopt the extension or serialization format.

Risk Assessment

Impact Low

Probability High

Description There are many parts of the IFC schema currently unimplemented by a vast majority of IFC implementations, for example textures. It is not unlikely that it will take time for a critical mass of IFC implementations to provide support for point clouds according to the proposed extension.

As has been noted, the schema extension proposed in this deliverable offers purely an addition to the IFC4 Addendum 1 schema. As such, tools can operate on the extended schema without impacting the acceptance and validity of instance files serialized according to tools that have no knowledge of this extended schema.

For the binary serialization format an existing ISO standard is followed, which makes its implementation effort feasible and unambiguous.

Contingency Solution The source code for the prototype introduced along with this deliverable is published and documented. If support among other vendors remains lacking, contributions can be made to open source IFC implementations, such as IfcOpenShell[17] and the BIMserver[18].

If support for the binary serialization format proves to be hurdle, a conversion utility that mediates between the various forms of serialization formats for IFC would be trivial to implement and circumvents the need for consuming applications to understand every possible serialization format.

[17] http://ifcopenshell.org
[18] http://bimserver.org

Risk Description The preservation community is reluctant to embrace the schema extension or additional serialization formats for IFC instance models.

Risk Assessment

Impact Medium

Probability Medium

Description Preservation experts might have concerns regarding a more diffuse IFC landscape, with different fragmented standards, which impacts interoperability. However, the schema extension proposed in this deliverable offers purely an addition to the current IFC schema. As such, tools that operate on the extended schema can do so without impacting the acceptance and validity of instance files without this extension. Further steps will be taken in collaboration with WP8 to communicate with the relevant standardization agency, buildingSMART, to embrace the extension and serialization format.

Contingency Solution Stakeholders that are not convinced solely by the performance and scalability improvements obtained from the binary serialization can confide in the large institutional users that have embraced HDF5[19] and other preservation projects [3, 14]. A conversion utility can be introduced that mediates between the various forms of serialization formats and can be part of Preservation Planning workflows of an archival system.

The DURAARK consortium and interested stakeholders will have to urge for standardization of the schema extension by buildingSMART and provide viable reference implementations to make a strong case.

[19] [...] HDF5 1.6 is an approved standard recommended for use in NASA Earth Science Data Systems [...] https://earthdata.nasa.gov/standards/hdf5

7 Software licenses

The following table gives an overview of the software licenses generated and used for the web for the software prototype and schema extension. Permissive licenses are chosen specifically in order not to hinder adoption by the industry, while at the same time, encouraging valuable additions to be contributed back into the product.

IPR Type	**IP used or generated**	**Software name**	**License**	**Information**
software	used	IfcOpenShell	LGPL	http://ifcopenshell.org
software	used	pythonOCC	LGPL	http://www.pythonocc.org/
software	used	PyQt4	GPL	http://www.riverbankcomputing.co.uk/software/pyqt/intro
software	used	h5py	BSD-like[a]	http://www.h5py.org/
software	used	numpy	BSD	http://www.numpy.org/
software	generated	Software prototype	LGPL	https://github.com/DURAARK/IFCPointCloud
standard	generated	Schema extension	CC BY 4.0	https://github.com/DURAARK/IFCPointCloud

[a]Similar to a 3-clause BSD. With additional clauses that require to state modifications to the source and the request to acknowledge the HDF Group and NCSA.

8 Conclusions and impact

In this deliverable a schema extension is introduced that enables to model point cloud subsets as native IFC entities. The layered compression approach of orthogonal components, being *a*) floating point numeral discretization as integer numerators (§3.2.3) *b*) discretization of projected point clouds into heightmaps (§3.2.2) *c*) specific-purpose compression algorithms on a file format level (§4.2), results in file sizes below industry standard point cloud formats, while introducing random access reading, self-documenting datatypes and efficient binary serialization, as opposed to prevalent IFC serialization mechanisms.

The work outlined in this deliverable impacts several concurrent development efforts within the DURAARK consortium and complements the workflows as they are identified within the project. In particular, workflows related to deposit and retrieval of archival objects are helped by a unified serialization format that allows efficient binary access to subsets of the model, irregardless of the model's file size. Therefore, the binary serialization format for IFC instance models, following an existing ISO standard, introduced in this deliverable, will become a core part of the WorkbenchUI, described in DURAARK deliverable D2.5, in addition to the currently documented workflows centered around IFC-SPF and e57. This will realize an improved efficiency in terms of deposit and streaming partial retrieval. Secondly, workflows pertaining to geometrical enrichment and difference detection are helped by a schema extension that allows to explicitly model semantic relationships between point cloud subsets and building element decompositions. More specifically, the schema extension has the potential to provide a formalized, standardized and more efficient alternative to the currently implemented RDF-based segmented linking format described in D5.1 and D2.3 Listings 1 and 2.

Additional further integration work is to be completed in collaboration with WP5 to expose their developed point cloud compression algorithms as a transparent block level compression option within the HDF5 serialization, as it is believed to perform significantly better than the general purpose gzip compression offered by default by the HDF5 library. In the prototype, shipped together with this deliverable, only physical building element surfaces are used as shape proxies to parametrize points. Therefore, points that have remained unassociated to building elements can not be parametrized and are therefore left uncompressed. A deeper integration into the toolchain developed within WP5 will enable to fit surfaces through unassociated points. These surfaces can be serialized into the IFC file unattached to the product scene graph, but still form the base for the efficient point cloud serialization techniques outlined in this deliverable. As such, the proposed binary serialization format offers a standardized and self-documenting platform to emit point clouds according to highly effective compression algorithms developed within WP5, while maintaining reliability, interoperability and suitability for digital preservation offered natively by the HDF5 file format.

A more complete evaluation of the proposed schema extension and serialization will be part of the ongoing evaluation work harmonized in the, to be released, D7.4 deliverable. It will include a more detailed assessment that stems from requirements identified by different stake holder groups and their use cases, as these varying perspectives will have implications on the required spatial resolution, precision, desired per-point attributes and concerns regarding efficient reading and writing. As a result of this evaluation, recommendation can be formulated, for example in terms of required precision, coexisting different levels of details within a model and degrees of decomposition.

Lastly, in conjunction with WP8, more detailed standardization activities will have to unfold. More specifically, for the buildingSMART International Standards Summit in October 2015 in Singapore[20], a compelling use case description, detailed standard specification and documentation will have to be submitted in order for the consortium to contemplate embracing the proposed schema extension and serialization format.

[20] http://www.buildingsmart.org/event/standards-summit-singapore-2/

References

[1] ASTM. E2807 - 11 standard specification for 3d imaging data exchange.

[2] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. Commun. ACM, 18(9):509–517, September 1975.

[3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In Proceedings of the 12th International Society for Music Information Retrieval Conference, pages 591–596, Miami (Florida), USA, October 24-28 2011. http://ismir2011.ismir.net/papers/OS6-1.pdf.

[4] Ankit Bhatla, Soo Young Choe, Oscar Fierro, and Fernanda Leite. Evaluation of accuracy of as-built 3d modeling from photos taken by handheld digital cameras. Automation in Construction, 28:116 – 127, 2012.

[5] Mario Botsch, Andreas Wiratanaya, and Leif Kobbelt. Efficient high quality rendering of point sampled geometry. In Proceedings of the 13th Eurographics Workshop on Rendering, EGRW '02, pages 53–64, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association.

[6] Alexander Braun, Sebastian Tuttas, Andre Borrmann, and Uwe Stilla. A concept for automated construction progress monitoring using bim-based geometric constraints and photogrammetric point clouds. In ITcon Vol. 20, Special Issue ECPPM 2014 - 10th European Conference on Product and Process Modelling, pages 68 – 79, 2015.

[7] Ioannis Brilakis, Manolis Lourakis, Rafael Sacks, Silvio Savarese, Symeon Christodoulou, Jochen Teizer, and Atefe Makhmalbaf. Toward automated generation of parametric {BIMs} based on hybrid video and laser scanning data. Advanced Engineering Informatics, 24(4):456 – 465, 2010. Construction Informatics.

[8] M. Bügler, G. Ongunmakin, J. Teizer, P. A. Vela, and A. Borrmann. A comprehensive methodology for vision-based progress and activity estimation of excavation processes for productivity assessment. In Proc. of the EG-ICE Workshop on Intelligent Computing in Engineering, Cardiff, Wales, 2014.

[9] Vailin Choi and Mike Folk. Investigations into using hdf5 as an alternative to step for finite element modeling data. In Proceedings of the 9th NASA-ESA Workshop on Product Data Exchange, National Aeronautics and Space Administration, Santa Barbara, CA, 2007.

[10] Stefan Dietze, Jakob Beetz, Ujwal Gadiraju, Georgios Katsimpras, Raoul Wessel, and René Berndt. Towards preservation of semantically enriched architectural knowledge. In 3rd International Workshop on Semantic Digital Archives (SDA 2013), September 2013.

[11] Julie Digne, Raphaëlle Chaine, and Sébastien Valette. Self-similarity for Accurate Compression of Point Sampled Surfaces. Computer Graphics Forum, 2014.

[12] Matthew T. Dougherty, Michael J. Folk, Erez Zadok, Herbert J. Bernstein, Frances C. Bernstein, Kevin W. Eliceiri, Werner Benger, and Christoph Best. Unifying biological image formats with hdf5. Commun. ACM, 52(10):42–47, 2009.

[13] Tim Golla, Christopher Schwartz, and Reinhard Klein. Towards Efficient Online Compression of Incrementally Acquired Point Clouds. In Jan Bender, Arjan Kuijper, Tatiana von Landesberger, Holger Theisel, and Philipp Urban, editors, Vision, Modeling & Visualization. The Eurographics Association, 2014.

[14] Jim Gray, David T. Liu, Maria Nieto-Santisteban, Alex Szalay, David J. DeWitt, and Gerd Heber. Scientific data management in the coming decade. SIGMOD Rec., 34(4):34–41, December 2005.

[15] Yan Huang, Jingliang Peng, C. C. J. Kuo, and M. Gopi. A Generic Scheme for Progressive Point Cloud Coding. Visualization and Computer Graphics, IEEE Transactions on, 14(2):440–453, March 2008.

[16] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. Sensors, 12(2):1437, 2012.

[17] Rensselaer Polytechnic Institute. Image Processing Laboratory and D.J.R. Meagher. Octree Encoding: a New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer. 1980.

[18] Shi Pu and George Vosselman. Knowledge based reconstruction of building models from terrestrial laser scanning data. {ISPRS} Journal of Photogrammetry and Remote Sensing, 64(6):575 – 584, 2009.

[19] R.B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In Robotics and Automation (ICRA), 2011 IEEE International Conference on, pages 1–4, May 2011.

[20] Ruwen Schnabel and Reinhard Klein. Octree-based point-cloud compression. In M. Botsch and B. Chen, editors, Symposium on Point-Based Graphics 2006. Eurographics, July 2006.

[21] Ruwen Schnabel, Sebastian Möser, and Reinhard Klein. Fast vector quantization for efficient rendering of compressed point-clouds. Computers and Graphics, 32(2):246–259, April 2008.

[22] Pingbo Tang, Daniel Huber, Burcu Akinci, Robert Lipman, and Alan Lytle. Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. Automation in Construction, 19(7):829 – 843, 2010.

[23] Jochen Teizer. Status quo and open challenges in vision-based sensing and tracking of temporary resources on infrastructure construction sites. Advanced Engineering Informatics, 29(2):225 – 238, 2015. Infrastructure Computer Vision.

[24] Jun Yang, Man-Woo Park, Patricio A. Vela, and Mani Golparvar-Fard. Construction performance monitoring via still images, time-lapse photos, and video streams: Now, tomorrow, and the future. Advanced Engineering Informatics, 29(2):211 – 224, 2015. Infrastructure Computer Vision.

[25] Jacob Ziv and Abraham Lempel. Compression of individual sequences via variable-rate coding, 1978.

A IFC Schema Extension EXPRESS

```
SCHEMA IFC4POINTCLOUD;

USE FROM IFC4 (
        IfcCartesianPointList,
        IfcColourSpecification,
        IfcElement,
        IfcGeometricRepresentationItem,
        IfcGrid,
        IfcLabel,
        IfcLengthMeasure,
        IfcNormalisedRatioMeasure,
        IfcParameterValue,
        IfcRectangularTrimmedSurface,
        IfcColourRGBList,
        IfcDirectionList
);

TYPE IfcPointCloudCoordinateSelect = SELECT
    (IfcCartesianPointList
    ,IfcParameterValueList
    ,IfcGridOffsetList);
END_TYPE;

ENTITY IfcNormalizedRatioMeasureList
 ABSTRACT SUPERTYPE OF (ONEOF(IfcColourRGBList, IfcDirectionList));
    Values : LIST [1:?] OF LIST [3:3] OF IfcNormalisedRatioMeasure;
END_ENTITY;

ENTITY IfcDiscreteNormalizedRatioMeasureList
 ABSTRACT SUPERTYPE OF (ONEOF(IfcDiscreteColourRGBList, IfcDiscreteDirectionList));
    Values      : LIST [1:?] OF LIST [3:3] OF INTEGER;
    SampleWidth : LIST [3:3] OF INTEGER;
END_ENTITY;

ENTITY IfcDiscreteColourRGBList
 SUBTYPE OF (IfcDiscreteNormalizedRatioMeasureList);
END_ENTITY;

ENTITY IfcDiscreteDirectionList
 SUBTYPE OF (IfcDiscreteNormalizedRatioMeasureList);
END_ENTITY;

ENTITY IfcParameterValueList
 ABSTRACT SUPERTYPE OF (ONEOF(IfcContinuousParameterValueList,
     IfcDiscreteParameterValueList))
 SUBTYPE OF (IfcGeometricRepresentationItem);
    WMinOffset : IfcLengthMeasure;
    WMaxOffset : IfcLengthMeasure;
    Surface    : IfcRectangularTrimmedSurface;
END_ENTITY;
```

```
ENTITY IfcContinuousParameterValueList
 SUBTYPE OF (IfcParameterValueList);
    Values : LIST [1:?] OF LIST [3:3] OF IfcParameterValue;
END_ENTITY;

ENTITY IfcDiscreteParameterValueList
 SUBTYPE OF (IfcParameterValueList);
    Values       : LIST [1:?] OF LIST [3:3] OF INTEGER;
    USampleWidth : INTEGER;
    VSampleWidth : INTEGER;
    WSampleWidth : INTEGER;
END_ENTITY;

TYPE IfcPointCloudAttribute = SELECT
    (IfcColourRGBList,
     IfcDiscreteColourRGBList,
     IfcDirectionList,
     IfcDiscreteDirectionList);
END_TYPE;

ENTITY IfcPointCloud
 SUBTYPE OF (IfcGeometricRepresentationItem);
    LevelOfDetail : OPTIONAL IfcLabel;
    Coordinates   : IfcPointCloudCoordinateSelect;
    Attributes    : LIST [0:?] OF IfcPointCloudAttribute;
END_ENTITY;

TYPE IfcGridSelect = SELECT
    (IfcSurfaceGrid,
     IfcGrid);
END_TYPE;

ENTITY IfcSurfaceGrid
  SUBTYPE OF (IfcGeometricRepresentationItem);
    Location: IfcPointOnSurface;
    USpacing: IfcParameterValue;
    VSpacing: IfcParameterValue;
    NumUAxes: INTEGER;
    NumVAxes: INTEGER;
END_ENTITY;

ENTITY IfcGridOffsetList
 ABSTRACT SUPERTYPE OF (ONEOF (IfcContinuousGridOffsetList,
     IfcDiscreetGridOffsetList))
 SUBTYPE OF (IfcGeometricRepresentationItem);
    Grid                : IfcGridSelect;
    MinOrthogonalOffset : IfcLengthMeasure;
    MaxOrthogonalOffset : IfcLengthMeasure;
END_ENTITY;

ENTITY IfcContinuousGridOffsetList
 SUBTYPE OF (IfcGridOffsetList);
```

```
        Offsets             : LIST [1:?] OF LIST [1:?] OF IfcParameterValue;
    END_ENTITY;

    ENTITY IfcDiscreteGridOffsetList
     SUBTYPE OF (IfcGridOffsetList);
        Offsets             : LIST [1:?] OF LIST [1:?] OF INTEGER;
        SampleWidth         : INTEGER;
    END_ENTITY;

    ENTITY IfcPointCloudElement
     SUBTYPE OF (IfcElement);
    END_ENTITY;

    END_SCHEMA;
```

Listing 7: ISO 10303 part 11 EXPRESS schema listing of the proposed IFC Point Cloud extension.