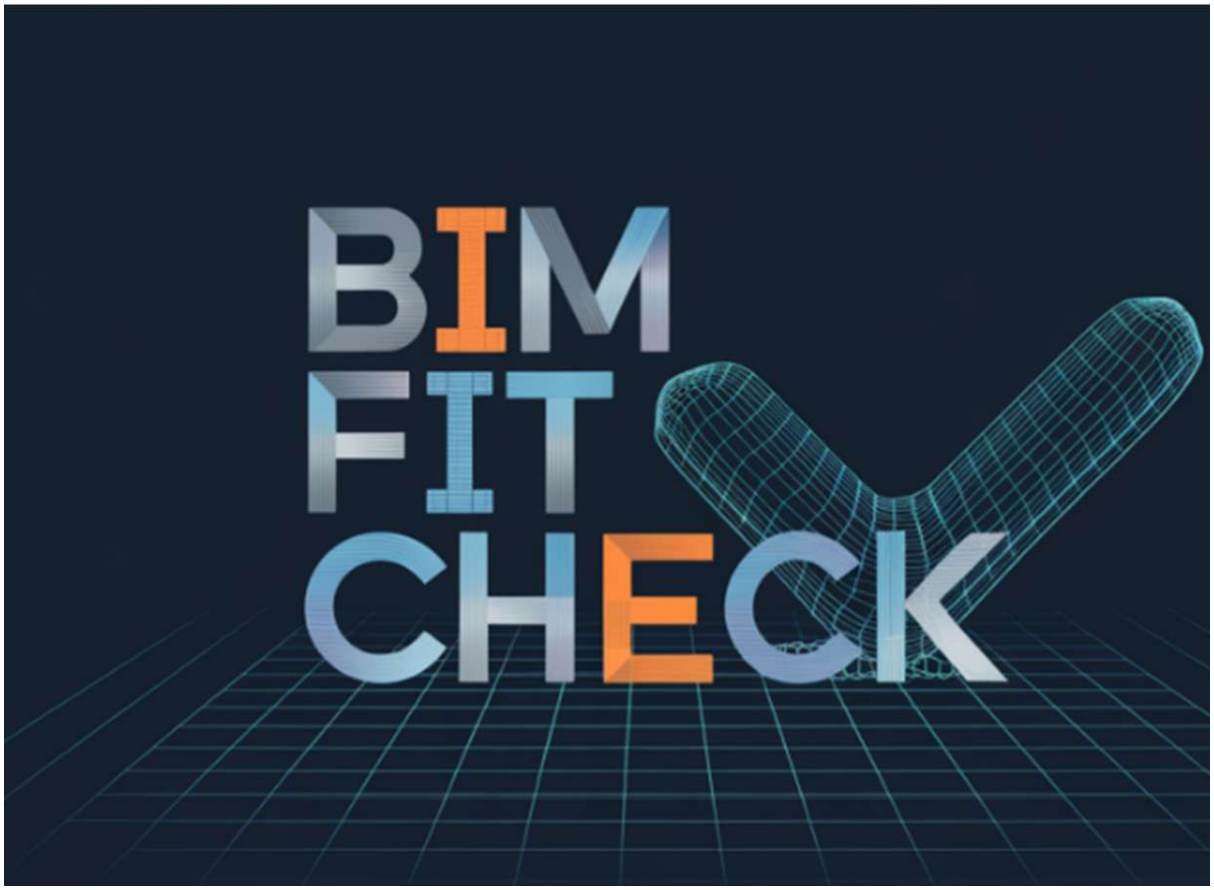
BIM Fit Check — IFC Georeferencing & Alignment Exchanges

Technical Findings & Recommendations



Authors

Štefan Jaud *(corresponding author)*

Anne-Kathrin Birkenbeul
Jörg Braunes
Rick Brice
Christian Clemen
Dean Hintz
Christoph Kautter
Elke Kocher
Christian Leverenz
Andreas Pinzenöhler
Steffen Rabe
Rainer Raacke
Artur Schütz
Rico Steyer
Bernhard Wehrle

Table of Contents

0 Document Metadata

0.1 Author Data

Corresponding author: Štefan Jaud, Jaud IT GmbH — stefan@jaud-it.com

Anne-Kathrin Birkenbeul, con terra GmbH — a.birkenbeul@conterra.de

Jörg Braunes, Thinkproject Deutschland GmbH — Joerg.Braunes@thinkproject.com

Rick Brice, Washington State Department of Transportation — Richard.Brice@wsdot.wa.gov

Christian Clemen, HTW Dresden — christian.clemen@htw-dresden.de

Dean Hintz, Safe Software Inc. — dean.hintz@safe.com

Christoph Kautter, DB InfraGO AG — Christoph.Kautter@deutschebahn.com

Elke Kocher, RIB Software GmbH — Elke.Kocher@rib-software.com

Christian Leverenz, IB&T Software GmbH — christian.leverenz@card-1.com

Andreas Pinzenöhler, IQSoft GmbH — andreas.pinzenoehler@iqsoft.com

Steffen Rabe, RIB Software GmbH — Steffen.Rabe@rib-software.com

Rainer Raacke, buildingSMART Deutschland e.V. — rainer.raacke@buildingsmart.de

Artur Schütz, Thinkproject Deutschland GmbH — Artur.Schuetz@thinkproject.com

Rico Steyer, AKG Software Consulting GmbH — steyer@akgsoftware.de

Bernhard Wehrle, AKG Software Consulting GmbH — wehrle@akgsoftware.de

0.2 File Version

Version	Date	Description	Author
0.1	2026-03-08	First draft	Rick Brice
0.2	2026-03-13	Complete restructure	Štefan Jaud
0.3	2026-03-13	Polishing, resolved last todos	Štefan Jaud, Rick Brice
0.4	2026-03-18	Added section on segment placement	Štefan Jaud
0.5	2026-03-18	Added sections on `IfcAxis2PlacementLinear.Axis` default	Štefan Jaud
0.6	2026-04-10	Added georeferencing findings	Štefan Jaud, Christian Clemen
1.0	2026-05-12	Final polishing	Štefan Jaud

1 Introduction

BIM Fit Check® is a new interactive event format from buildingSMART Germany — a playful challenge for software companies. The target audience are software companies and development teams who want to demonstrate that their products support the open standards and services of buildingSMART for common — carefully chosen and defined — use cases.

Software companies that take up the challenge attend a series of workshops to implement the required functionalities within a set timeframe. Those who feel ready can then demonstrate how their product masters the challenges in front of an audience at a buildingSMART Germany event. If the solution passes the challenge before an independent jury — confirming that the product ensures greater interoperability — it's "Check!" and "Congratulations!" This is a win-win-win for the software company, buildingSMART Germany, and — perhaps most importantly — the software user: the successful software company is awarded a digital badge documenting its achievement, and the joint success is promoted through buildingSMART Germany's communication channels. The silent winner is the software user, who can be confident that their software correctly implements the open standards.

1.1 Intended Readership

This document is written for readers who already know the IFC standard and its specification(s). It skips background explanations — the aim is a practical bucket list for those improving future releases.

2 Participants

2.1 Software Vendors

Two BIM Fit Check events took place: the first in May 2025 (Georeferencing Exchanges) and the second in September 2025 (Alignment Exchanges). The following software vendors participated in one or both events.

1. AKG Software Consulting GmbH — VESTRA INFRAVISION
2. albert.ing GmbH — squirrel 3.0
3. A+S Consult GmbH — KorFin®
4. IB&T Software GmbH — card_1
5. QLX GmbH — smarttrass
6. RIB Software GmbH — RIB Civil
7. ProVI GmbH — ProVI CAD
8. Safe Software Inc. + con terra GmbH — FME Platform
9. Thinkproject Deutschland GmbH — VDC Manager

2.2 Jury

Georeferencing

Andreas Geiger, Karlsruhe Institute of Technology (KIT)

Christoph Kautter, DB InfraGO AG

Christian Clemen, Dresden University of Applied Sciences (HTW Dresden)

Alignment Exchanges

Andreas Pinzenöhler, IQSoft GmbH

Peter Bonsma, RDF Ltd.

Rick Brice, Washington State Department of Transportation

2.3 Lead

Štefan Jaud, Jaud IT GmbH *(Technical Lead)*

Rainer Raacke, buildingSMART Deutschland e.V. *(Organizational Lead)*

3 Findings

The BIM Fit Check event provided a valuable opportunity to validate the IFC 4x3 Add 2 specification in real-world scenarios for georeferencing and alignment applications. Through implementation efforts, discussion, and comparison across multiple independent participants, the group identified a range of insights that highlight areas of the specification where additional clarity or refinement would be beneficial. The following findings summarize the key technical observations and challenges that emerged from this effort, offering guidance for future versions of the IFC specification and its implementations.

3.1 Lack of Authoritative Guidance

There is a lack of authoritative guidance covering the use of alignment semantic and geometric definitions. See Section 4.2 for specific suggestions.

3.2 Lack of Validation Data

Authoritative validation data sets are needed to validate implementations. Test cases should be formalized and transparent. In addition to model-level test data, small unit test-like files are needed that — independent of any agreed model view definitions — each highlight a specific structure, attribute, or attribute value. These focused test files allow implementors to isolate and verify individual aspects of the specification in a controlled way, and must be accompanied by thorough documentation explaining the intent and expected behaviour of each test case. Both the test files and their documentation should be publicly available, ideally included directly in the specification. The testing procedures themselves should also be formalized to ensure consistent, reproducible validation across implementations. BIM Fit Check made extensive use of the IFC4x3 Alignment Geometry Implementation Guide by Richard Brice, PE, and the example data produced by Andreas Pinzenöhler. Both are exactly what the IFC specification itself should provide.

3.3 Predefined Types for `IfcAlignment`

The only predefined types for alignment are USERDEFINED and NOTDEFINED. This is insufficient for practical use. A list of suggested predefined types is provided in Section 4.3.

3.4 Object Nesting Concept Template Excluded from the AbV

The stationing of an alignment is defined by the first `IfcReferent` nested to the alignment. Concept Template 4.1.4.4.3 Object Nesting general usage relates `IfcAlignment` and `IfcReferent` with an `IfcRelNests` relationship. However, the concept template is not part of the Alignment-based View. See Section 4.4 for suggestions regarding the scope of the Alignment-based View.

3.5 Best Practice for Nesting with `IfcAlignment`

`IfcAlignmentHorizontal`, `IfcAlignmentVertical`, and `IfcAlignmentCant` are associated with `IfcAlignment` through an `IfcRelNests` relationship. See Concept Template 4.1.4.4.1 Alignment Layouts. Concept Template 4.1.4.4.3 Object Nesting general usage also relates `IfcReferent` with `IfcAlignment` through an `IfcRelNests` relationship. The following questions emerge:

1. Should the alignment layout entities `IfcAlignmentHorizontal`, `IfcAlignmentVertical`, and `IfcAlignmentCant` be Related Objects in the same or different `IfcRelNests`?
2. `IfcRelNests.RelatedObjects` is an ordered collection. If the alignment layout entities belong in the same `IfcRelNests`, what is the required order? The most logical order is horizontal, vertical, then cant but this is not explicitly documented.
3. Should `IfcReferent` be contained in the same or different `IfcRelNests` as the alignment layout entities? If the same, what is the relative order of an `IfcReferent` compared to the alignment layout entities? The best practice is to have `IfcReferent` in its own `IfcRelNests`, separate from the alignment layout entities. This approach was agreed upon by the participants of BIM Fit Check.

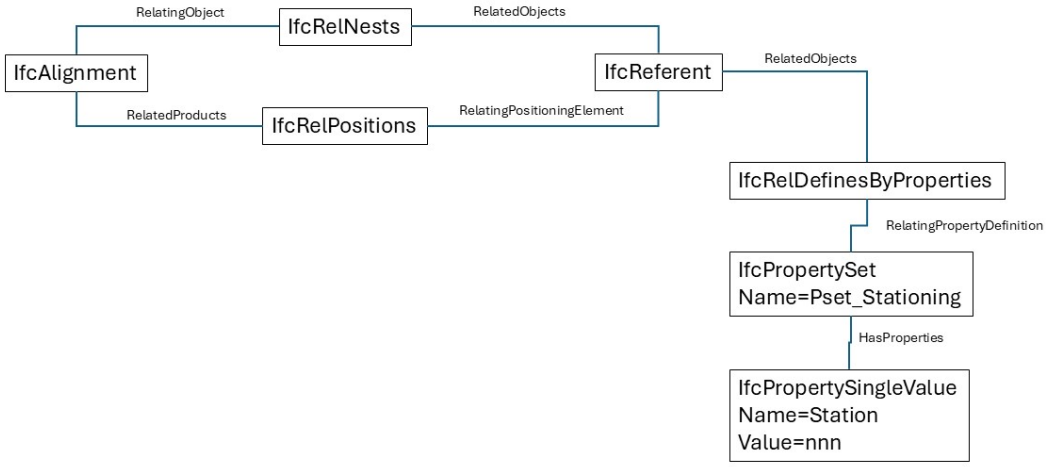
3.6 Alignment Positioning

The IFC specification defines stationing along an alignment through `IfcReferent`. Section 5.4.3.44 `IfcReferent` notes that:

- Referents can be nested to alignments, using `IfcRelNests`, to describe stations along an alignment. Being `IfcRelNests.RelatedObjects` an ordered list, the first nested referent is the starting station of a given alignment.
- The stationing value of any object can be provided, using an `IfcReferent` and the respective Pset_Stationing. The relationship between the given object and the referent indicating its stationing value is an `IfcRelPositions`.

However, it is not specified how the starting station relates back to the alignment itself:

- Should the first nested referent also have an `IfcRelPositions` relationship with the alignment to indicate it positions the start of the alignment, as shown in the sketch below?
- Should `IfcAlignment` position the `IfcReferent` via `IfcRelPositions` even if the `IfcReferent` itself is not placed using linear placement?



3.7 Incomplete Treatment of Alignment Geometry Representations

`IfcOffsetCurveByDistances`, `IfcPolyline`, and `IfcIndexedPolyCurve` are valid representations of `IfcAlignment`. However, concept templates for these geometric representations are absent from the IFC specification. See Section 4.6 for suggestions.

3.8 Alignment Layout with Simplified Geometry

The IFC specification seems to exclude semantic alignment layout with a simplified geometric representation. Concept Template 4.1.4.4.1.1 Alignment Layout - Horizontal, Vertical and Cant in conjunction with 4.1.7.1.1.1 Alignment Geometry - Horizontal, 4.1.7.1.1.2 Alignment Geometry - Horizontal and Vertical and 4.1.7.1.1.3 Alignment Geometry - Horizontal, Vertical and Cant imply that all alignment layouts must have a geometric representation based on `IfcCompositeCurve`, `IfcGradientCurve`, and `IfcSegmentedReferenceCurve`, respectively.

The following requirements are imposed:

1. A **zero-length segment** shall be added, at the end of the list of segments for `IfcAlignmentSegment.DesignParameters`.
2. If the geometry definition is also present, then each of the zero-length segments shall have an `IfcCurveSegment` counterpart - of length zero.

Does the requirement of a zero-length `IfcCurveSegment` in the geometry definition eliminate the possibility of `IfcPolyline` or `IfcIndexedPolyCurve` as geometric representations?

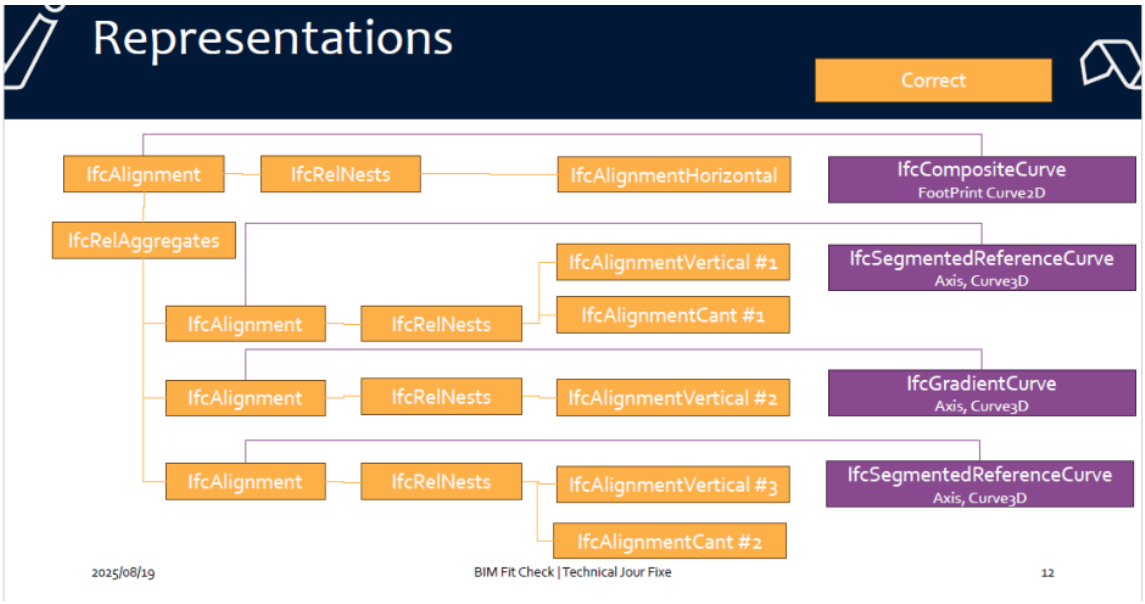
If not, the requirement for a zero-length segment is not applicable to `IfcPolyline` or `IfcIndexedPolyCurve` because it violates the prohibitions on coincidence of consecutive points.

3.9 Proper Geometric Representation When Reusing Horizontal Layout

Concept Template 4.1.4.4.1.2 Alignment Layout - Reusing Horizontal Layout defines the semantic definition of multiple vertical, or vertical and cant, layouts reusing a common horizontal layout. A corresponding concept template for the geometric representation is absent from the IFC specification. The following questions emerge:

- Should the representations all be contained within `IfcAlignment.Representation.Representations` of the parent `IfcAlignment`?
- Should each representation be contained within the `IfcAlignment.Representation.Representations` of the children `IfcAlignment`?
- Should the `IfcCompositeCurve` representation of the horizontal layout be a representation on the parent `IfcAlignment` or on the `IfcAlignmentHorizontal`?
- Is the presence of one geometry a requirement for all geometries to be present?
- Which combinations of supported shape representations for alignment are valid? For example, can the horizontal alignment have an `IfcCompositeCurve` representation with one vertical as an `IfcGradientCurve` and the other vertical as an `IfcOffsetCurveByDistances`?

A possible agreement for the representation is shown in the figure below.



3.10 Proper Treatment of Horizontal, Vertical, and Cant Layouts of Different Length

8.7.3.4 `IfcAlignmentVerticalSegment` and 8.7.3.1 `IfcAlignmentCantSegment` both have a StartDistAlong attribute of type `IfcLengthMeasure`. This allows for the first `IfcAlignmentSegment` within `IfcAlignmentVertical` and `IfcAlignmentCant` to begin before or after the start of the `IfcAlignmentHorizontal`.

`IfcGradientCurve` and `IfcSegmentedReferenceCurve` could be longer or shorter than the BaseCurve (`IfcCompositeCurve`).

- What is the intended treatment strategy for evaluating the 3D location of a point where the curves do not overlap?
- Is the composition of `IfcCompositeCurve`, `IfcGradientCurve`, `IfcSegmentedReferenceCurve` limited to the region where all the curves overlap?
- Should the shorter curves be projected from their start and end points? What is the strategy for projecting the curves?

3.11 Non-Continuous Semantic Segment Definition

In a semantic alignment definition, each segment is defined by a start point and a length. While `IfcAlignmentHorizontal` clearly states that layout segments are connected end-to-start, `IfcAlignmentVertical` and `IfcAlignmentCant` do not define a like requirement. If a segment is too short to reach the start point of the subsequent segment, a gap arises in the alignment definition. The IFC specification does not address this situation:

- Is a non-continuous semantic segment definition considered invalid data?
- If not, what is the intended strategy for evaluating geometry in the gap between segments?

3.12 Limitations on `IfcAlignment.ObjectPlacement`

Alignments and alignment segments are generally placed in the global coordinate system for a project. A best practice recommendation for `IfcAlignment.ObjectPlacement` and `IfcAlignmentSegment.ObjectPlacement` is `IfcLocalPlacement` with:

1. `IfcLocalPlacement.RelativePlacement.Location` = `IfcCartesianPoint`((0.,0.,0.))
2. `IfcLocalPlacement.RelativePlacement.Axis` = null or `IfcDirection`((0.,0.,1.))
3. `IfcLocalPlacement.RelativePlacement.RefDirection` = null or `IfcDirection`((1.,0.,0.))

Neither `IfcAlignment` nor `IfcAlignmentSegment` shall be placed using linear placement (`IfcLinearPlacement`) or grid placement (`IfcGridPlacement`). See Section 4.8 for proposed formal propositions. Note that `IfcAlignmentSegment` may be placed in 2D, in which case the above constraints apply accordingly, omitting the Z component where not applicable.

3.13 Alignment Concepts for Dual Carriageway and Multi-Track Configurations

In highway and railway civil engineering works, it is common to have a central horizontal alignment defining separate road or railway elements on both sides independently. Each side carries its own vertical profile and, in the case of rail, its own cant, while stationing is carried along the common central alignment. For rail geometry, this configuration is sometimes referred to as the 7-line geometry. These are common constructions and should have explicit semantic and geometric definitions in the IFC specification.

3.14 Placement Along Alignment in Distorted Geometric Context

Consider a sleeper whose geometry is defined in a geometric context that is 1:1 to reality using mapped geometry, but whose placement is along an alignment in a geometric context distorted according to the projected CRS. The IFC specification does not provide explicit guidance or a mechanism for achieving this.

3.15 Ambiguous Distance Measure When Using `IfcOffsetCurveByDistances` as BaseCurve for `IfcLinearPlacement`

When `IfcOffsetCurveByDistances` is used as the BaseCurve of `IfcLinearPlacement`, the `IfcParameterValue` used to position objects along the curve is ambiguous. `IfcOffsetCurveByDistances` is an interpolated curve formed by discrete offsets from its BasisCurve; its geometry depends on the method of interpolation and the sampling frequency. As a result, the distance along the offset curve — and therefore the position of any object placed using `IfcLinearPlacement` — cannot be determined precisely. This ambiguity is compounded in the presence of spiral segments: no curve is mathematically equidistant from a spiral at all points, leaving the behaviour of `IfcParameterValue` along such an offset undefined.

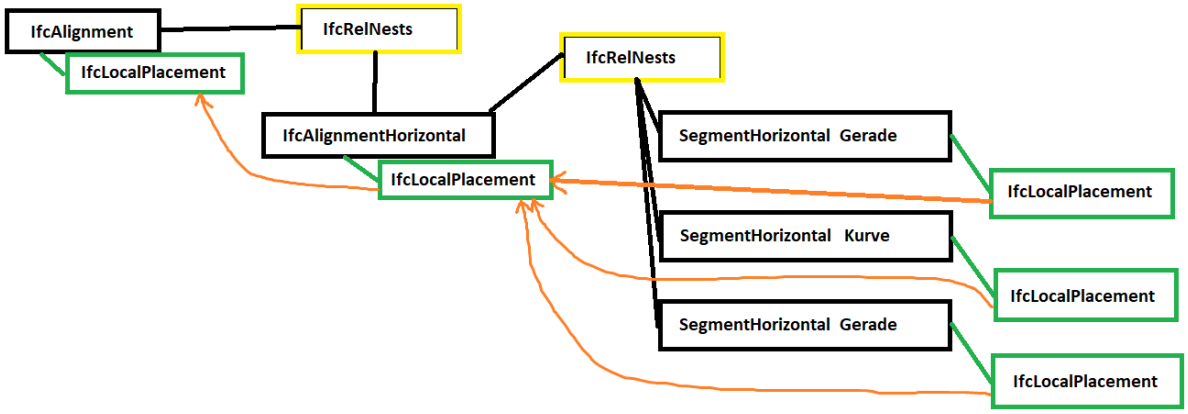
3.16 Partial Concept Templates for Curve Segment Geometry

The subsections of 4.2.2.1 Curve Segment Geometry provide partial concept templates that aim to detail requirements for the various curve segment geometry types. The information provided is insufficient for consistent implementations, lacking the specificity needed for software vendors to independently and correctly implement the required mappings.

3.17 Coordinate Reference for `IfcAlignmentSegment` Parameters

`IfcAlignmentSegment` inherits an ObjectPlacement attribute, but the specification does not state whether providing it is required.

Without ObjectPlacement, the coordinate reference for segment design parameters — such as `IfcAlignmentHorizontalSegment.StartPoint` — is undefined. `IfcAlignmentHorizontalSegment` says the segment placement corresponds to `IfcCurveSegment.StartPlacement`, but stating a correspondence is not the same as defining a coordinate system. See Section 4.11.



The files added in PR #507 of the buildingSMART ifc-gherkin-rules repository illustrate this problem.

```
+ #137=IFCCARTESIANPOINT((3660.44140161384,2050.74103009789));
+ #138=IFCALIGNMENTHORIZONTALSEGMENT($,$,#137,1.34433278507712,0.,0.,1886.9,$,.LINE.);
+ #139=IFCALIGNMENTSEGMENT('3oWeK2ZbX8svZBt6w1JO20',$,$,$,$,$,$,#138);
```

3.18 Undefined Default for `IfcAxis2PlacementLinear.Axis`

`IfcAxis2PlacementLinear` defines the local coordinate system used by `IfcLinearPlacement` to orient objects along an alignment. Its `Axis` attribute — the local Z direction — has no defined default value. By contrast, `IfcAxis2Placement3D` defaults Axis to `(0,0,1)` when omitted. The corresponding default for `IfcAxis2PlacementLinear` is not defined, and its correct value depends on the geometry of the underlying curve:

- For `IfcGradientCurve` and `IfcSegmentedReferenceCurve`, which carry elevation data, the Axis should be perpendicular to the RefDirection within the "Distance Along – Elevation" plane.
- For `IfcCompositeCurve`, which has no elevation component, `(0,0,1)` is the natural choice.

Since `RefDirection` already defaults to the curve tangent — itself varying by curve type and position — the omission of a normative Axis default leaves implementations unable to consistently

orient objects placed along alignments. See Section 4.12 and buildingSMART/IFC4.3.x-development#732.

3.19 Compound CRS Without a Dedicated EPSG Code

When a compound coordinate reference system — combining a horizontal CRS with a vertical CRS — carries its own EPSG code, encoding it in `IfcCoordinateReferenceSystem.Name` is straightforward. However, compound CRS without a dedicated EPSG code are common in practice, particularly in national and project-specific configurations. In these cases the horizontal CRS and the vertical CRS each have their own EPSG codes, but no single code identifies the combination.

`IfcProjectedCRS` provides two attributes that could carry component identifiers: `GeodeticDatum` and `VerticalDatum`. Both are untyped strings, and the specification does not clarify whether they are intended to hold datum identifiers (e.g. EPSG:6258 — the ETRS89 datum) or CRS identifiers (e.g. EPSG:4258 — the ETRS89 geographic CRS). No guidance is given on what to write in `Name` when the compound CRS has no code of its own, nor on how software should reconstruct the full compound CRS from its parts.

A concrete example illustrates the problem. `EPSG:9934` (DB_REF2016 zone 4) is an active compound CRS — a combination of projected CRS `EPSG:5684` and vertical CRS `EPSG:9927` — and its `Name` encoding is unambiguous. By contrast, combining `EPSG:31468` (DHDN / 3-degree Gauss-Krüger zone 4) with the same vertical CRS `EPSG:9927` yields a valid compound CRS in practice, but no dedicated EPSG code exists for this combination. The only encoding option is to decompose the compound into its parts:

`IfcProjectedCRS` attribute	**EPSG:9934**	**EPSG:31468 + EPSG:9927**	**EPSG:25832 + EPSG:5783**
`Name`	`EPSG:9934`	no compound code exists	no compound code exists
`GeodeticDatum`	`EPSG:1081` (Deutsche Bahn Reference System)	`EPSG:4314` (DHDN)	`EPSG:6258` (ETRS89)
`VerticalDatum`	`EPSG:1318` (GNTRANS2016)	`EPSG:1318` (GNTRANS2016)	`EPSG:5182` (DHHN92)
Projected CRS	`EPSG:5684` (derivable from compound)	`EPSG:31468` — no attribute available	`EPSG:25832` — no attribute available

Vertical CRS	`EPSG:9927` (derivable from compound)	`EPSG:9927` — no attribute available	`EPSG:5783` — no attribute available

See Section 4.13 for proposed options.

3.20 No Mechanism to Validate Coordinate Transformation

The IFC specification defines `IfcMapConversion` to encode the mathematical transformation from a local engineering coordinate system to a georeferenced CRS. This transformation can be checked for internal consistency, but the specification provides no mechanism to encode independent validation data — that is, point pairs with known coordinates in both the local engineering coordinate system and the referenced CRS — that would allow software or field surveyors to verify the transformation empirically.

Such control points are standard practice in geodetic and surveying workflows. Without a normative way to encode them in IFC, there is no structured basis for validating that a model's georeferencing is correct beyond inspecting the transformation parameters themselves. See Section 4.14 for a proposed extension to address this gap.

3.21 Insufficient Guidance on `IfcMapConversion` vs `IfcRigidOperation` when used with `IfcProjectedCRS`

The IFC 4.3 specification provides two coordinate operation subtypes for positioning a project on Earth: `IfcMapConversion`, used in the "Project Global Positioning" concept template, and `IfcRigidOperation`, used in the "Project Global Positioning Mapped" concept template. The specification does not clearly define which georeferencing scenario calls for which.

`IfcMapConversion` provides the transformation parameters to convert a local engineering coordinate system into a georeferenced projected (combined) CRS. It carries a full set of transformation parameters: translation (Eastings, Northings, OrthogonalHeight), rotation (XAxisAbscissa, XAxisOrdinate), and a unit scale factor. It is intended for cases where the project coordinate system is an arbitrary local system with no simple derivation or relation to a CRS.

`IfcRigidOperation` applies a pure translation offset without scale or rotation. It is intended for cases where the project is already working within a recognised CRS but with truncated or shifted coordinates — a common practice to avoid numerically large coordinate values. The operation simply restores the removed offset to recover the full CRS coordinates. As such, the model remains in length units of the projection.

In practice, both operations can encode what might appear to be the same data — a translation with scale 1 and no rotation. Without clear guidance, implementors either apply the wrong operation or are unaware that `IfcRigidOperation` exists for the already-in-CRS scenario. During BIM Fit

Check, inconsistent use of these two operations was observed across participant submissions. See Section 4.15 for proposal.

3.22 Undefined Defaults for `IfcMapConversion.XAxisAbscissa` and `IfcMapConversion.XAxisOrdinate`

`IfcMapConversion.XAxisAbscissa` and `IfcMapConversion.XAxisOrdinate` are both declared `OPTIONAL` in the IFC 4.3 schema. Together they encode the rotation of the local engineering coordinate system relative to the map CRS — specifically, the direction of the local X axis expressed in the map CRS horizontal plane. When both are absent, the intended interpretation is that no rotation is applied: the local X axis aligns with map East, corresponding to values XAxisAbscissa = 1 and XAxisOrdinate = 0.

The specification does not state this default anywhere. Unlike `IfcMapConversion.Scale`, which carries an explicit default of 1.0 when omitted (clarified in IFC4.3.x-development issue #43), no equivalent statement exists for the rotation attributes. Implementors who omit these attributes must either assume the intended default by convention or populate them defensively to avoid ambiguity. Consumers who encounter a null pair have no normative basis for their interpretation.

This gap was confirmed as unaddressed upstream: a search of `buildingSMART/IFC4.3.x-development` issues and pull requests found no open or closed item specifically covering `XAxisAbscissa`/`XAxisOrdinate` defaults. The gap is directly analogous to finding 3.18 (`IfcAxis2PlacementLinear.Axis` missing default), which is tracked as GitHub issue #732 and marked before-NWI. See Section 4.17 for proposal.

3.23 Official Example Files Have Redacted `FILE_DESCRIPTION` Headers

The STEP physical files distributed as official examples in the IFC 4.3 specification carry redacted metadata in their ISO 10303-21 `FILE_DESCRIPTION` and `FILE_NAME` records. Fields that would normally identify the authoring application, its version, and the originating organisation are replaced with placeholder strings such as `'redacted'` or left blank. While this may have been done to avoid endorsing specific software tools, it removes information that is essential for diagnosing problems in the files.

Example files distributed with a specification are not required to be perfect — they are exemplary, illustrating a particular concept or use case. In practice, example files sometimes contain minor errors, non-normative workarounds, or application-specific dialects that reflect the tool used to produce them. When such issues are encountered, the natural diagnostic step is to identify the authoring application and version: different exporters have known quirks, and understanding which tool produced a file allows reviewers to distinguish a specification gap from a tool-specific behaviour.

With redacted headers, this diagnostic path is closed. It is impossible to determine:

- Which software produced the file and whether that software is known to have relevant bugs or idiosyncrasies
- Which version of the authoring application was used — relevant when a bug was fixed in a subsequent release
- Whether an anomaly in the file reflects the author's intent, a tool limitation, or a misunderstanding of the specification

During BIM Fit Check, participants and the jury encountered example files from the IFC specification whose behaviour could not be fully explained without knowledge of their provenance. The redacted headers prevented reconstruction of the production context and left open questions that could not be resolved.

3.24 Backwards Compatibility Preserves Legacy Georeferencing Constructs

The IFC specification maintains full backwards compatibility across versions. As a consequence, the LoGeoRef30 georeferencing approach — encoding the project base point through `IfcSite.ObjectPlacement`, without any coordinate reference system declaration — remains equally valid in IFC 4.3 alongside the semantically superior LoGeoRef50 approach based on `IfcMapConversion` and `IfcProjectedCRS`.

During BIM Fit Check, participants questioned why backwards compatibility requires preserving LoGeoRef30 as a first-class option in a current IFC 4.3 exchange. From an implementor's perspective, supporting both approaches simultaneously introduces unnecessary complexity: software must handle the full range of LoGeoRef levels to be considered conformant, even when the target exchange has no legacy requirement. The presence of LoGeoRef30 as a valid option also means that a conforming IFC 4.3 file may carry no machine-readable CRS declaration at all — an outcome that is semantically regressive relative to what the standard is capable of expressing.

The root tension is that the IFC specification is a single artefact covering all use cases and all eras. Backwards compatibility is a valid design constraint for the schema, but it does not follow that every exchange must accept the full historical range of constructs. See Section 4.19 for a proposal to address this through MVD-level constraints.

3.25 `IfcProduct.ObjectPlacement` Has No Explicit Link to a Geometric Representation Context

Georeferencing an IFC model requires a chain of three elements: (1) the product's effective world-space coordinates, derived by resolving the `IfcProduct.ObjectPlacement` hierarchy; (2) the `IfcGeometricRepresentationContext` that defines the local engineering coordinate system (ECS) those coordinates are expressed in; and (3) the `IfcMapConversion` that transforms from the ECS to the map CRS. Steps (1) and (3) are both well-supported by the schema. Step (2) — linking a product placement to its governing context — is not.

`IfcRepresentation.ContextOfItems` provides an explicit, traversable reference from a product's geometry representations to the `IfcGeometricRepresentationContext` that governs them. No equivalent attribute exists on `IfcObjectPlacement` or any of its subtypes (`IfcLocalPlacement`, `IfcLinearPlacement`, `IfcGridPlacement`). The governing context for a placement is inferred by convention: a null `IfcLocalPlacement.PlacementRelTo` is understood to anchor the placement to the world coordinate origin of *some* `IfcGeometricRepresentationContext`, but the schema does not say which one.

This ambiguity is latent in single-context files, where the convention holds by default. It becomes concrete in files with multiple contexts — for example, a "Model" context at full scale carrying `IfcMapConversion`, and a "Plan" context for 2D drawings at drawing scale. A product's placement and its geometry representations may reference different contexts, and there is no normative basis for resolving which context governs the placement coordinates when applying `IfcMapConversion`.

`IfcLinearPlacement` (introduced in IFC 4.1) adds a further dimension: a linearly placed product is positioned relative to a curve geometry. The curve's own coordinates are in the ECS of their representation context, which is again not explicitly linked to the placement. Correctly georeferencing a linearly placed product requires evaluating the curve geometry at the station, applying lateral and vertical offsets, and then transforming to the map CRS — but the applicable `IfcMapConversion` cannot be identified from the placement structure alone.

No correction to the current specification is proposed here. The gap reflects a structural asymmetry in how IFC separates geometric context from spatial placement, and addressing it would require a schema change with broad impact on the placement model.

3.26 Single Model Context Cannot Represent Ground-Dimension Geometry and Map-Dimension Alignment Simultaneously

In railway and infrastructure projects, geometry exists in two distinct coordinate domains: *true to reality* (physical distances as measured on the terrain surface, corrected for map projection distortion and elevation) and *true to the underlying CRS* (coordinates as defined in the projected CRS, where distances are distorted by the map projection and elevation reduction). IFC 4.3 provides two coordinate operation types that correspond to these domains. `IfcMapConversion` — used in the "Project Global Positioning" concept — applies a translation, a rotation, and a scale factor to convert a local engineering coordinate system into a projected map CRS; the scale factor accounts for the difference between the units of the CRS and the units of the model. `IfcRigidOperation` — used in the "Project Global Positioning Mapped" concept — applies a pure translation to restore a large CRS offset that was removed for numerical convenience; no scale is involved, meaning the file coordinates are true to the underlying CRS.

A concrete exchange scenario that IFC cannot currently represent without approximation: a railway file contains an alignment whose curve geometry is true to the underlying CRS (surveyed in a CRS and stored with an applied `IfcRigidOperation`). Individual structural elements — for example sleepers — are represented as mapped items using a shared geometry library, where the geometry is true to reality, capturing the element's true physical dimensions. The sleeper is placed along the alignment via `IfcLinearPlacement`. Because the alignment curve is true to the CRS and the sleeper geometry is true to reality, the two representations operate at different scales. Where the combined scale factor of the map projection and elevation correction deviates from 1.0 (as it does in all real-world projects), applying the sleeper geometry in the alignment's coordinate space introduces a systematic dimensional error.

The correct solution would require two separate 3D `IfcGeometricRepresentationContext` instances in the same file: one with `IfcMapConversion` (with scale) governing geometry that is true to reality, and one with `IfcRigidOperation` (no scale) governing geometry that is true to the underlying CRS, such as the alignment. IFC's informal proposition that a project shall contain at most one 3D model representation context prevents this. The constraint exists for good reasons — multiple 3D contexts create ambiguity about which context governs a given product's placement, as described in finding 3.25 — but its effect is that true-to-reality and true-to-CRS content cannot coexist in a single well-formed IFC file.

It is worth noting that the two-context solution would also imply two distinct `IfcCoordinateOperation` instances in the same file — one per `IfcGeometricRepresentationContext` of type "Model". The types of those operations need not be the same: in the scenario above, one context would carry an `IfcMapConversion` (true to reality) and the other an `IfcRigidOperation` (true to the underlying CRS). More generally, any combination of `IfcCoordinateOperation` subtypes is conceivable depending on the nature of the two coordinate domains being mixed. The current schema does not prohibit multiple `IfcCoordinateOperation` instances per se, but the one-context constraint renders the question moot in practice.

No correction to the current specification is proposed here. The finding points to a structural limitation of the single-context model whose resolution would require coordinated changes across the context, placement, and georeferencing parts of the schema.

3.27 IFC 2x3 Lacks Native Entities for CRS-Based Georeferencing — Convention-Based Property Set Workaround Required

IFC 2x3 does not contain `IfcMapConversion` or `IfcProjectedCRS`. These entities — which together constitute the LoGeoRef50 georeferencing level — were introduced in IFC 4. Files conforming to IFC 2x3 can therefore only express georeferencing through LoGeoRef20 (latitude/longitude on `IfcSite`) or LoGeoRef30 (`IfcSite.ObjectPlacement` as project base point). Neither level encodes a full coordinate transformation into a projected CRS with named

datum, projection, zone, and scale.

Jaud et al. (2025) propose a convention-based workaround to backport the full range of IFC 4 and IFC 4.3 georeferencing constructs into IFC 2x3 files using five property sets, all attached to `IfcProject`:

- **`ePset_MapConversion`** — carries the transformation parameters: `Eastings`, `Northings`, `OrthogonalHeight`, `XAxisAbscissa`, `XAxisOrdinate`, `Scale`. Mirrors `IfcMapConversion` (IFC 4). For projects with an arbitrary local engineering coordinate system transformed into a projected CRS.
- **`ePset_MapConversionScaled`** — extends `ePset_MapConversion` with a combined scale factor that accounts separately for the map projection scale and the elevation scale correction. Mirrors the additional precision requirements beyond what the basic `Scale` attribute of `IfcMapConversion` provides.
- **`ePset_RigidOperation`** — carries a pure translation offset: `Eastings`, `Northings`, `OrthogonalHeight`. No rotation or scale. Mirrors `IfcRigidOperation` (IFC 4.3). For projects whose coordinates are already within a recognised CRS but with a large offset subtracted for numerical convenience.
- **`ePset_ProjectedCRS`** — carries the projected CRS description: `Name`, `Description`, `GeodeticDatum`, `VerticalDatum`, `MapProjection`, `MapZone`, `MapUnit`. Mirrors `IfcProjectedCRS` (IFC 4).
- **`ePset_GeographicCRS`** — carries a geographic (unprojected) CRS description for cases where the target CRS is expressed in angular units rather than projected metres. Mirrors the distinction between geographic and projected CRS that is implicit in the IFC 4 schema but not represented by a dedicated entity.

The workaround allows IFC 2x3 files to carry the same georeferencing information as IFC 4 and IFC 4.3 files, enabling processing pipelines to handle all schema versions through a unified read path. Where native entities exist (IFC 4 or IFC 4.3 file), they take precedence; for IFC 2x3 files, the property sets serve as the georeferencing source.

The fundamental limitation of this approach is that it is a community convention, not a normative part of the IFC 2x3 schema. A conformant IFC 2x3 file may carry these property sets without any validator being able to check their presence, completeness, or correctness. Consumers that are unaware of the convention will silently ignore the sets and fall back to the lower LoGeoRef levels. This gap was discussed during BIM Fit Check by participants, the jury, and the technical leadership as a practical concern for organisations maintaining IFC 2x3 workflows alongside newer schema versions.

Reference: Jaud, Š.; Max, P.-C.; Pullmann, T.; Geiger, A. (2025). Comprehensive Georeferencing Support for all Official Versions of the IFC Standard. In: Tagungsband des 36. Forum Bauinformatik,

Aachen, 24.–26. September 2025. RWTH Aachen University. KITopen: 1000182635.

4 IFC Specification Improvement Suggestions

BIM Fit Check revealed specific shortcomings in the IFC specification through direct implementation experience. The following suggestions address those shortcomings. Where the specification is silent, ambiguous, or inconsistent, concrete changes are proposed.

4.1 Documentation Must Be Accompanied by Examples and Validation

Experience from BIM Fit Check has shown that providing documentation alone and expecting implementors to follow it does not work in practice. The approach of "give us documentation, we implement" has proven insufficient. Documentation must be accompanied by examples, validation data, and test cases.

Examples must be provided in consistent, agreed data formats. Critically, hand-crafted examples should themselves pass the buildingSMART Validation Service — a file that is intended as a reference but fails automated validation is worse than no example, as it creates ambiguity about which behaviour is correct. Verification should not depend on manual inspection alone; automated tools that can reproducibly confirm conformance are essential. A stable, versioned state of the Validation Service must be established and agreed upon before any future BIM Fit Check event, so that participants and the jury share a common, objective reference.

4.2 Comprehensive Example Coverage for Alignment Concepts

A comprehensive set of examples should be developed that covers the two alignment layout concepts (Sections 3.8 and 3.9), four alignment geometry concepts (Section 3.7), and five supported shape representations (Sections 4.7 and 4.9). A complete index mapping which example covers which concept should be made available.

4.3 Predefined Types for `IfcAlignment`

The following predefined types are suggested. For the procedure governing schema changes such as extending this list, see Section 5.2.

Predefined Type	Description
ROADWAY	A road designed for motor vehicle traffic, typically forming part of a highway or arterial network.
STREET	A road within a built-up area, bordered by buildings and serving both vehicles and pedestrians.

PATH	A route for non-motorized users such as pedestrians or cyclists.
RAILWAY	A track for rail vehicles guided by fixed rails, used for passenger or freight transport.
REFERENCEAXIS	A geometric reference line used to define stationing or to position other alignments, without representing a physical infrastructure element.
CHANNEL	An open waterway, natural or artificial, used to convey water.
CANAL	An artificial waterway constructed for navigation or irrigation.
SEWER	An underground conduit for conveying wastewater or stormwater.
DRAINAGE	A channel or conduit designed to remove surface or subsurface water from an area.
PIPE	A closed conduit used to convey fluids, gases, or other substances under pressure or gravity.
MAGLEV	A guideway for magnetic levitation vehicles, which travel without physical contact using electromagnetic forces.
TRAM	A rail vehicle running on tracks laid in a public road, used for urban passenger transport.
TROLLEY	A rubber-tyred urban transit vehicle powered by overhead electric wires, also known as a trolleybus.
WATERWAY	A navigable body of water such as a river or canal, used for the movement of vessels.
LEVEE	An embankment constructed alongside a river or coast to prevent flooding.
DIKE	A barrier built to hold back water or to reclaim land from the sea.

EMBANKMENT	A raised earthwork used to carry a road, railway, or other infrastructure above the surrounding terrain.
AUXILIARYAXIS	A secondary geometric axis supporting the definition of other alignments, such as an offset or transition reference, without representing a primary infrastructure element.

4.4 Alignment-Based View Scope

The scope of the Alignment-based View should be extended to include the Object Nesting concept template (Concept Template 4.1.4.4.3), as the stationing of an alignment — defined by the first `IfcReferent` nested to the alignment — depends on this concept template. See Section 3.4.

4.5 Interpretation of Insertion Point Coordinates of Layout Placements

The ObjectPlacement of `IfcAlignmentHorizontal`, `IfcAlignmentVertical`, and `IfcAlignmentCant` uses `IfcLocalPlacement`. The ObjectPlacement.PlacementRelTo points to the `IfcAlignment.ObjectPlacement` that nests this layout. The (x, y) coordinates of the insertion point are interpreted differently for each layout type, relative to the coordinate system of the parent `IfcAlignment`:

- **`IfcAlignmentHorizontal`:** The (x, y) coordinates are interpreted as the (x, y) coordinates in the coordinate system of the `IfcAlignment`.
- **`IfcAlignmentVertical`:** The (x, y) coordinates are interpreted as (s, z), where s is the distance along the horizontal projection and z is the elevation in the coordinate system of the `IfcAlignment`.
- **`IfcAlignmentCant`:** The (x, y) coordinates are interpreted as (s, d), where s is the distance along the horizontal projection and d is the deviation of the centreline — for example, half the cant when the cant rotates on a rail, and zero when the cant rotates around the centreline.

The RefDirection and Axis attributes of the `IfcLocalPlacement` are interpreted as follows, with the alternative that both attributes are omitted:

- **`IfcAlignmentHorizontal`:** RefDirection and Axis shall be equal to the RefDirection and Axis of the parent `IfcAlignment`'s ObjectPlacement, respectively.
- **`IfcAlignmentVertical`:** RefDirection and Axis shall be equal to the RefDirection and Axis of the parent `IfcAlignment`'s ObjectPlacement, respectively.
- **`IfcAlignmentCant`:** RefDirection and Axis shall be equal to the RefDirection and Axis of the parent `IfcAlignment`'s ObjectPlacement, respectively.

4.6 Alignment Geometry Concept Templates

Concept templates 4.1.7.1.1.1 Alignment Geometry – Horizontal, 4.1.7.1.1.2 Alignment Geometry – Horizontal and Vertical, and 4.1.7.1.1.3 Alignment Geometry – Horizontal, Vertical and Cant cover only the `IfcCompositeCurve`, `IfcGradientCurve`, and `IfcSegmentedReferenceCurve` representations. However, `IfcOffsetCurveByDistances`, `IfcPolyline`, and `IfcIndexedPolyCurve` are valid geometric representations of alignment layouts and are not covered by any concept template. These templates should be enhanced, or new concept templates developed, to cover these representations, including the applicable RepresentationIdentifier and RepresentationType values for each.

The concept templates should also clarify which geometric forms are permitted in conjunction with the zero-length segment requirement imposed on `IfcAlignmentSegment.DesignParameters`. In particular, it should be made explicit whether the zero-length segment requirement applies to `IfcPolyline` and `IfcIndexedPolyCurve` representations, or whether it is limited to `IfcCompositeCurve`-based representations.

4.7 New `IfcShapeRepresentation` Identifiers: Axis-FallBack and EndOfLine

The table of RepresentationIdentifier values for `IfcShapeRepresentation` should be extended with the following two entries:

Identifier	Description
Axis-FallBack	2D or 3D simplified representation of an alignment axis, e.g. as a polyline approximation of a higher-order parametric curve representation. Follows the pattern of Body-FallBack.
EndOfLine	Point or zero-length segment identifying the end of a curve or alignment representation. This value can be used for validation purposes, analogous to CoG.

4.8 Limitations on `IfcAlignment.ObjectPlacement`

The best practice described in Section 3.12 should be elevated to at minimum an informal proposition, and preferably a formal proposition, on `IfcAlignment` in the IFC specification:

IP1: The ObjectPlacement of `IfcAlignment` shall be of type `IfcLocalPlacement`. The placement shall not be of type `IfcLinearPlacement` or `IfcGridPlacement`.

IP2: The RelativePlacement of the `IfcLocalPlacement` shall have its Location at the origin (0., 0., 0.), its Axis equal to (0., 0., 1.) or omitted, and its RefDirection equal to (1., 0., 0.) or omitted.

IP3: The `IfcLocalPlacement` of `IfcAlignment` shall not be placed relative to another placement, unless that placement results in an identity transformation — that is, `IfcLocalPlacement.PlacementRelTo` shall be null or shall reference a placement that is itself an identity transformation.

A corresponding formal proposition (WHERE rule) would enforce these constraints at the schema level.

4.9 Restrict or Define Distance Measurement for `IfcOffsetCurveByDistances` as BaseCurve

The use of `IfcOffsetCurveByDistances` as the BaseCurve of `IfcLinearPlacement` should either be prohibited or be accompanied by a normative definition of the distance measurement method. Without such a definition, implementations cannot produce consistent results. Two options are proposed:

- **Option A:** Prohibit `IfcOffsetCurveByDistances` as the BaseCurve of `IfcLinearPlacement`.
- **Option B:** Define a normative method for computing distance along `IfcOffsetCurveByDistances` when used as a BaseCurve.

4.10 Expand Partial Concept Templates for Curve Segment Geometry

The partial concept templates in the subsections of 4.2.2.1 Curve Segment Geometry should be expanded to provide a complete and accurate mapping between the design parameters of a semantic alignment segment and the corresponding parent curve geometry. Currently, the information provided is incomplete and in some cases inaccurate. A reference implementation of the required mappings is available in the IFC4x3 Alignment Geometry Implementation Guide by Richard Brice, PE.

4.11 Require `IfcAlignmentSegment.ObjectPlacement`

`IfcAlignmentSegment.ObjectPlacement` shall be required. Its coordinate reference is the coordinate system of the constituent layout — the `IfcLocalPlacement` of whichever `IfcAlignmentHorizontal`, `IfcAlignmentVertical`, or `IfcAlignmentCant` the segment belongs to. See the figure in Section 3.17.

The relationship between `IfcAlignmentSegment.ObjectPlacement` and `IfcCurveSegment.StartPlacement` needs a formal definition. Saying the two "correspond" is not enough.

4.12 Define Default Axis for `IfcAxis2PlacementLinear`

A normative default for `IfcAxis2PlacementLinear.Axis` should be defined. Two approaches are proposed:

Option A — Geometry-dependent default: The default Axis depends on the type of the underlying BaseCurve. For `IfcGradientCurve` and `IfcSegmentedReferenceCurve`, the Axis is derived as the vector perpendicular to the RefDirection within the "Distance Along – Elevation" plane. For `IfcCompositeCurve`, the Axis defaults to `(0,0,1)`.

Option B — Simplified global-up orthogonalization: The Axis is always derived by orthogonalizing the global-up vector `(0,0,1)` against the RefDirection (the curve tangent). This mirrors — and reverses — the logic of `IfcAxis2Placement3D`, where Axis is primary and RefDirection is adjusted for orthogonality. This option has the advantage of a single consistent rule across curve types.

In either case, the specification should clarify the semantics when Axis and RefDirection are provided explicitly versus when they are omitted. Consideration should also be given to introducing derived attributes on `IfcAxis2PlacementLinear` that expose the local build axes, analogous to `IfcBuildAxes`, to make the computed coordinate frame explicit and verifiable.

4.13 Encode Compound CRS Without a Dedicated EPSG Code

The specification should define normative encoding for compound CRS that lack a dedicated EPSG code. Three options are proposed:

Option A — Horizontal CRS in `Name`, vertical in `VerticalDatum`: Populate `Name` with the EPSG code of the horizontal CRS (e.g. `EPSG:25832`) and `VerticalDatum` with the EPSG code of the vertical CRS (e.g. `EPSG:5783`). `GeodeticDatum` is omitted as it is implied by the horizontal CRS already expressed in `Name`. This reuses existing attributes without schema changes, but repurposes `VerticalDatum` in a way its current definition does not explicitly support.

Option B — Composite identifier in `Name`: Define a canonical string convention for `Name` when no single EPSG code is available, for example `EPSG:25832+EPSG:5783`. Software parses the `+` delimiter to extract the horizontal and vertical component codes. This requires a normative string format to be specified and agreed upon.

Option C — Schema extension: Introduce an `IfcCompoundCRS` entity or extend `IfcProjectedCRS` with typed attributes for an explicit horizontal CRS reference and an explicit vertical CRS reference, replacing the current free-text `GeodeticDatum` and `VerticalDatum` strings with structured references. This is the most expressive option but requires a schema change.

4.14 LoGeoRef60: Ground Control Points for Georeferencing Validation

Görne & Clemen (2019) have established the **LoGeoRef** (Levels of Georeferencing Reference) framework to describe increasing levels of georeferencing fidelity in IFC models. The currently supported levels in the IFC schema reach up to LoGeoRef50, which encodes a full coordinate transformation via `IfcMapConversion`. However, LoGeoRef50 provides no means of validating whether the stated transformation is empirically correct.

This report calls for adoption of **LoGeoRef60** as proposed by Görne & Clemen (2019). This level encodes surveyed ground control points — pairs of known coordinates in both the local engineering coordinate system and the referenced CRS. Ground control points serve two complementary purposes:

1. **Validation:** Software and surveyors can independently verify the `IfcMapConversion` parameters by forward-transforming each local point and comparing the result to the stated CRS coordinates. Discrepancies that exceed a stated tolerance indicate an erroneous or outdated transformation.
2. **Construction site connection:** Ground control points directly link BIM geometries to their physical realisation on the construction site. Surveyors use the listed CRS coordinates to set out positions from a total station or GNSS receiver. This closes the loop between the digital model and the physical works.

To support LoGeoRef60, a new entity (e.g., `IfcGroundControlPoint`) is proposed. Each instance represents one surveyed point and shall carry the following attributes:

Attribute	Type	Description
`PointNumber`	`IfcLabel`	Unique identifier of the control point, matching the survey documentation — if the class is derived from `IfcRoot`, its attribute `Name` could be reused
`PredefinedType`	`IfcGroundControlPointType`	Classification of the survey monument (e.g. nail, bolt, concrete pillar, benchmark plate)
`LocalCoordinates`	`IfcCartesianPoint`	Coordinates in the local engineering coordinate system as used in the IFC model
`CRSCoordinates`	`IfcCartesianPoint`	Coordinates in the referenced CRS (easting / northing / height, or latitude / longitude / height)

`AccuracyClass`	`IfcLabel`	OPTIONAL. Accuracy class or tolerance of the survey measurement
`Description`	`IfcText`	OPTIONAL. Additional remarks

The survey monument type is a standard attribute in geodetic practice; its inclusion ensures that the point can be physically located on site and re-observed if the model is revised.

`IfcGroundControlPoint` instances should be associated with the `IfcMapConversion` they validate. A minimum of three non-collinear control points is recommended to verify the full planar transformation (translation, rotation, scale); a fourth point provides redundancy and enables residual analysis. The choice of realisation should be determined through the schema change procedure described in Section 5.2.

4.15 Document Typical Scenarios for `IfcMapConversion` and `IfcRigidOperation`

The specification should add normative guidance clarifying the intended use of each operation, linked explicitly to the corresponding concept template. The following two scenarios cover the cases addressed by the specification:

Scenario	Recommended operation	Concept template
Project uses a local arbitrary engineering coordinate system with no direct relation to a CRS	`IfcMapConversion`	Project Global Positioning
Project coordinates are already within a recognised CRS but with a large offset subtracted for numerical convenience (truncated coordinates)	`IfcRigidOperation`	Project Global Positioning Mapped

The guidance should also clarify whether `IfcMapConversion` and `IfcRigidOperation` can be chained, and if so, in what order and with what semantics.

4.16 Define Default Values for `IfcMapConversion.XAxisAbscissa` and `IfcMapConversion.XAxisOrdinate`

Addresses finding 3.22.

The specification should add an explicit statement of the default values that apply when `XAxisAbscissa` and `XAxisOrdinate` are omitted:

Attribute	Default when omitted	Meaning
`XAxisAbscissa`	1	Local X axis points in the direction of map East
`XAxisOrdinate`	0	No rotation relative to the map CRS

The wording should mirror the treatment of `Scale`, which states: *"If omitted, the value of 1.0 is assumed."* An equivalent normative statement — *"If omitted, XAxisAbscissa defaults to 1 and XAxisOrdinate defaults to 0, indicating no rotation of the local coordinate system relative to the map coordinate system."* — resolves the ambiguity without any schema change.

Upstream action: No issue exists in `buildingSMART/IFC4.3.x-development` for this gap. A new issue should be filed, referencing the parallel treatment of `Scale` (issue #43) and `IfcAxis2PlacementLinear.Axis` (issue #732) as precedents. Given that #732 is marked before-NWI, this issue warrants the same priority.

4.18 Restore Authoring Metadata in Official Example Files

Addresses finding 3.23.

The `FILE_DESCRIPTION` and `FILE_NAME` records of official IFC example files should carry honest, complete metadata identifying the authoring application, version, and — where applicable — the producing organisation. Redacting this information does not protect the specification from the appearance of tool endorsement; it merely removes diagnostic value without benefit.

The redaction was introduced during the transition from IFC 4 to IFC 4.3 — the original IFC 4 example files carried complete headers. The contrast is visible when comparing the same example file across both spec generations. The file `basin-tessellation.ifc` as published in the IFC 4 ADD2 TC1 specification (Annex E) reads:

```
FILE_DESCRIPTION(('ViewDefinition [ReferenceView_V1]'),'2;1');
FILE_NAME(
 /* name */              '',
 /* time_stamp */        '2016-02-04T08:47:55',
 /* author */            ('Jon'),
 /* organization */      ('Unknown'),
 /* preprocessor_version */ 'GeomGymIFC by Geometry Gym Pty Ltd',
 /* originating_system */ 'Unknown Application',
 /* authorization */     'None');
```

The same file as published in the IFC 4.3 specification (Annex E) reads:

```
FILE_DESCRIPTION(('ViewDefinition [ReferenceView]'),'2;1');
FILE_NAME(
 /* name */              'basin-tessellation.ifc',
 /* time_stamp */        '2016-02-04T08:47:55',
 /* author */            ('redacted'),
 /* organization */      ('redacted'),
 /* preprocessor_version */ 'redacted',
 /* originating_system */ 'redacted - redacted - 3.14159',
 /* authorization */     'None');
```

Both files share the identical timestamp (`2016-02-04T08:47:55`), confirming they are the same source file. The IFC 4.3 version replaces author, organisation, preprocessor version, and originating system with the literal string `'redacted'` — including a placeholder version string (`3.14159`), which confirms the redaction was intentional rather than incidental.

The fix is correspondingly simple: identify the commit in the `buildingSMART/IFC4.3.x-development` repository that introduced the redaction and revert it for the affected example files. No new content needs to be authored; the original metadata already exists in the version history.

4.19 Distinguish Modern from Legacy MVDs to Enforce Current Georeferencing Constructs

Addresses finding 3.24.

The IFC schema must remain backwards compatible — that constraint is unlikely to change and is not itself the problem. The proposal is to resolve the tension at the MVD level rather than the schema level.

Proposed distinction: Modern MVDs vs. Legacy MVDs

MVD category	Georeferencing scope	Intended use
Modern MVD	Requires LoGeoRef50 or higher (`IfcMapConversion` / `IfcRigidOperation` + `IfcProjectedCRS`)	New projects, current workflows, IFC 4.x and later
Legacy MVD	Permits LoGeoRef30 (`IfcSite` geographic attributes)	Migration of existing IFC 2x3 data, archival exchange

A Modern MVD explicitly prohibits the use of LoGeoRef30 constructs — not by changing the schema, but by adding a normative constraint within the MVD itself. Software claiming conformance to a Modern MVD must reject or warn on LoGeoRef30 georeferencing. This gives implementors a clear, enforceable target without breaking conformance for legacy toolchains.

Broader applicability: The modern/legacy MVD distinction is not limited to georeferencing. The same mechanism can be applied wherever the IFC schema retains deprecated or superseded constructs for compatibility reasons. MVDs become the instrument through which the community enforces current best practice, while the schema retains its historical breadth. This approach decouples schema governance (slow, consensus-driven) from exchange governance (faster, use-case-driven).

Prerequisite: This proposal requires that the MVD governance procedure be established and that a nominated body has the authority to publish and maintain Modern MVDs. See Section 5.3.

5 Governance

5.1 Validation Service

The buildingSMART Validation Service proved disruptive during BIM Fit Check. Its requirements lack transparency, as checks are not clearly linked to the specific sections of the IFC specification being validated. Participants found the service difficult to work with and ultimately stopped relying on it.

Each check performed by the validation service should reference the specific section of the IFC specification it validates. A stable, versioned state of the validation service should be established and agreed upon before the start of any future BIM Fit Check event. If these improvements are made, the validation service could be a shared, objective reference for conformance checking — useful to both participants and the jury.

The gherkin rules that feed the validation service may themselves reflect unresolved specification issues. When the specification is silent or ambiguous, files get flagged as non-conformant — not because they are wrong, but because no requirement exists yet. See Section 4.11.

5.2 Schema Change Procedure

The IFC schema lacks a clear, public procedure for proposing, reviewing, and approving schema changes. This includes changes as seemingly minor as extending a predefined type list. Concrete examples identified during BIM Fit Check include the proposed extension of the `IfcAlignment` predefined type list (Section 4.3), the addition of new concept templates for alignment geometry representations (Section 4.6), and the introduction of new `IfcShapeRepresentation` identifiers (Section 4.7). The absence of such a procedure creates uncertainty for implementors and other stakeholders who wish to contribute to the evolution of the standard, and undermines confidence in the governance of the specification. The technical leadership of buildingSMART International is urged to establish and publish a formal schema change procedure — and to execute it.

5.3 MVD Governance and Amendment Procedure

No documented procedure exists for producing or governing a Model View Definition. It is unclear who has the authority to initiate, review, approve, or publish an MVD, and through what process. During BIM Fit Check, a nucleus of an Alignment Design Transfer View was produced. However, without a formal governance framework, there is no defined path for turning this nucleus into a ratified MVD, nor for proposing and approving amendments to it or any other established MVD.

The appropriate scope of an MVD for alignment is also unresolved. It is unclear whether a standalone Alignment-only MVD is the right vehicle, or whether alignment requirements should be structured as a base MVD with optional add-on MVDs covering specific use cases or extensions. This structural question needs to be settled before further investment in MVD development makes sense.

The IFC specification offers multiple ways to implement the same concept, without guidance on which to prefer. The example of georeferencing — where both LoGeoRef30 and LoGeoRef50 are valid but incompatible approaches — illustrates a broader problem: without MVDs that explicitly constrain implementation choices, interoperability cannot be guaranteed even between conforming implementations.

A related confusion arose during BIM Fit Check regarding which requirements stemmed from the IFC General Usage MVD and which were BIM Fit Check-specific additions. The General Usage MVD concept exists in the IFC ecosystem, but its documentation was insufficiently clear to serve as a shared reference during the event. Because these were among the first structured BIM Fit Check rounds, the distinction had not yet been worked out in advance: jury and participants alike were navigating the boundary between general-usage IFC conformance and exchange-specific BIM Fit Check requirements in real time. Future rounds should establish this distinction explicitly before the fit phase begins — clearly separating what is required by the General Usage MVD from what is a BIM Fit Check extension.

6 References

6.1 Concept Templates

- 4.1.4.4.1 Alignment Layouts
- 4.1.4.4.1.1 Alignment Layout – Horizontal, Vertical and Cant
- 4.1.4.4.1.2 Alignment Layout – Reusing Horizontal Layout
- 4.1.4.4.3 Object Nesting
- 4.1.7.1.1.1 Alignment Geometry – Horizontal
- 4.1.7.1.1.2 Alignment Geometry – Horizontal and Vertical
- 4.1.7.1.1.3 Alignment Geometry – Horizontal, Vertical and Cant
- 4.1.9.4 Project Global Positioning
- 4.1.9.6 Project Global Positioning Mapped

6.2 Entities

- `IfcAlignment`
- `IfcAlignmentCant`
- `IfcAlignmentCantSegment`
- `IfcAlignmentHorizontal`
- `IfcAlignmentHorizontalSegment`
- `IfcAlignmentSegment`
- `IfcAlignmentVertical`
- `IfcAlignmentVerticalSegment`
- `IfcAxis2PlacementLinear`
- `IfcCompositeCurve`
- `IfcCurveSegment`
- `IfcGradientCurve`
- `IfcGridPlacement`
- `IfcIndexedPolyCurve`
- `IfcLinearPlacement`
- `IfcLocalPlacement`
- `IfcMapConversion`

- IfcOffsetCurveByDistances
- IfcPolyline
- IfcProjectedCRS
- IfcReferent
- IfcRelNests
- IfcRelPositions
- IfcRigidOperation
- IfcSegmentedReferenceCurve
- IfcShapeRepresentation

6.3 Property Sets

- Pset_Stationing