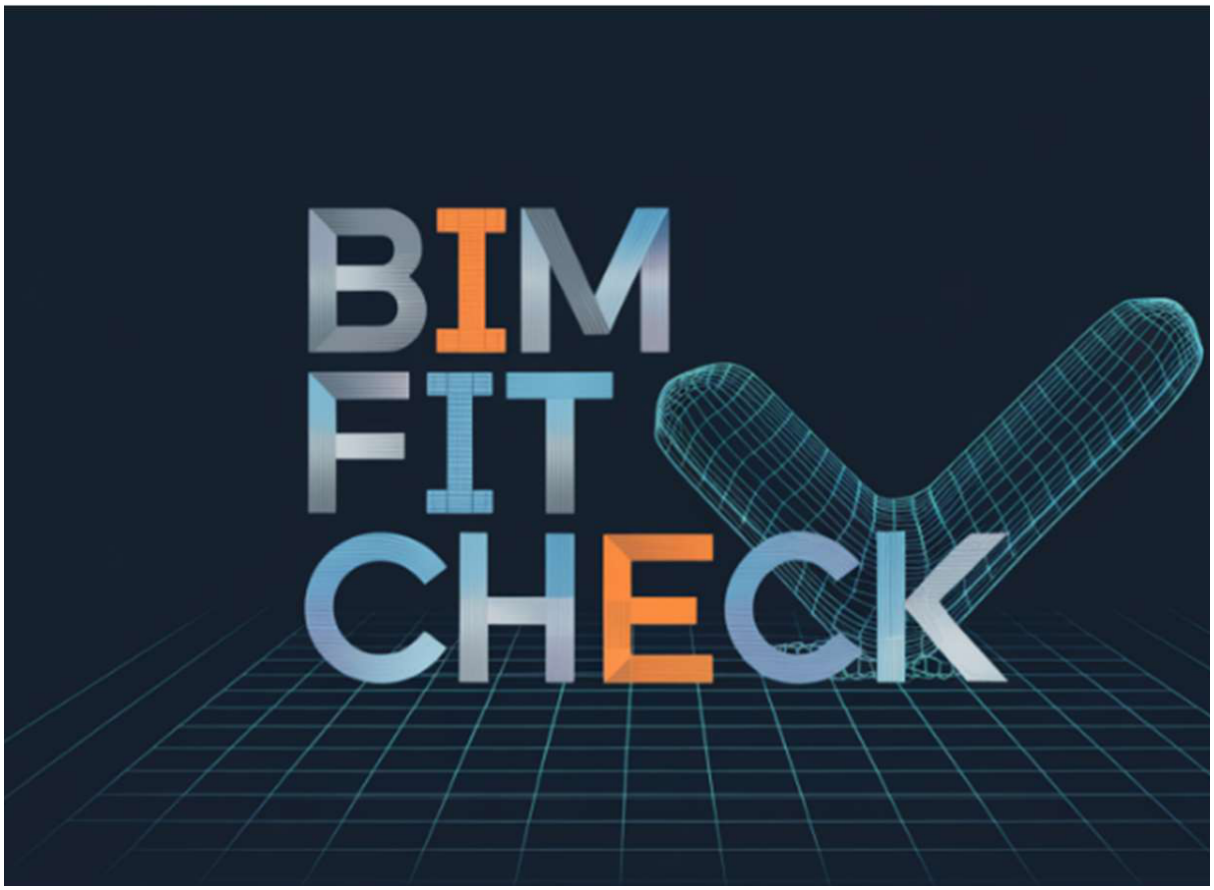
BIM Fit Check — BCF Exchanges

Technical Findings & Recommendations



Authors

Štefan Jaud *(corresponding author)*

Philipp Bonny
Maximilian Bürgi
Miguel Pérez
Rico Steyer
Simon Werz
Dominique Marchand-Fässler

Table of Contents

0 Document Metadata

0.1 Author Data

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

Philipp Bonny, WPM Ingenieure GmbH — Philipp.Bonny@wpm-ingenieure.de

Maximilian Bürgi, AKG Software Consulting GmbH — buergi@akgsoftware.de

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

Simon Werz, WPM Ingenieure GmbH — Simon.Werz@wpm-ingenieure.de

Miguel Pérez, PMG Projektraum Management GmbH — mpe@pmgnet.de

Dominique Marchand-Fässler, PMG Projektraum Management GmbH — dmf@pmgnet.de

0.2 File Version

Version	Date	Description	Author
0.1	2026-04-20	Initial version	Štefan Jaud
0.2	2026-04-28	Added findings 3.10–3.12	Štefan Jaud, Philipp Bonny, Simon Werz
0.3	2026-05-08	Added general remarks	Štefan Jaud, Miguel Pérez, Dominique Marchand-Fässler
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.

This document collects the technical findings that emerged from the BCF round. It follows the structure established in the IFC-round findings documents:

- **Section 3 — Findings:** observed issues, per topic, with element references and possible causes.
- **Section 4 — Improvements:** recommended best practices and resolutions, mapped one-to-one to the findings.

The intended readership is implementors of BCF in software products and members of the bSI BCF working group. Background knowledge of BCF 2.x / 3.0 is assumed.

2 Participants

2.1 Software Vendors

The BCF round took place at the 23rd buildingSMART Anwendertag in Berlin on 13 May 2026. The following software vendors participated.

1. AKG Software Consulting GmbH — VESTRA INFRAVISION B69/B70
2. albert.ing GmbH — Squirrel 3.1
3. ComputerWorks GmbH — Vectorworks 2026
4. DICAD Systeme GmbH — STRAKON 2026
5. PMG Projektraum Management GmbH — PAVE
6. WPM Ingenieure GmbH — ZIS Ing-Bau

2.2 Jury

Georg Dangl, DanglIT GmbH

Marcus Nill, IngenieurGruppe Bauen

Pasi Paasiala, Solibri Inc.

2.3 Lead

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

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

3 Findings

3.1 `relatedTopic` references non-existing topic

Element: `Topic/RelatedTopic/@Guid`

Observed behaviour: A BCF topic contains a `relatedTopic` element whose GUID points to a topic that does not exist in the same BCF file (or container).

Possible causes:

- The referenced topic was deleted after the relation was created.
- The relation was copied across BCF files without bringing the referenced topic along.
- The GUID was manually entered and contains a typo.

Open question: What is the correct best practice for consuming applications?

- (a) Silently delete the broken link.
- (b) Inform the user (warning / validation message) but keep the link.
- (c) Attempt to resolve the GUID against the application's own issue list (cross-file resolution).

3.2 `Header/File` date precedes IFC file creation

Element: `Markup/Header/File/@Date` (BCF 2.x) / equivalent in BCF 3.0

Observed behaviour: A BCF file is exported (e.g. from a review or coordination session) before the referenced IFC file has been formally created or saved. The `Header/File` element is expected to carry the date of the IFC file, but no valid date exists yet.

Possible causes:

- BCF export triggered early in the workflow, before the IFC model is finalised or even created.
- Tooling auto-populates `Date` with the BCF export timestamp, which may predate the IFC file's actual creation date.
- IFC file does not yet exist — header is speculative or forward-referencing.

Open question: What should a conformant application write in `Header/File/@Date` in this scenario?

- (a) Omit the `Date` attribute entirely (if the schema allows it).
- (b) Write the BCF export date as a fallback.
- (c) Write a null / sentinel value.

- (d) Leave the entire `Header/File` element out if no IFC file is linked.

3.3 `Header/File/@Filename` ambiguity: IFC header name vs. OS filename

Element: `Markup/Header/File/@Filename`

Related: 3.2 (both concern `Header/File` content semantics).

Observed behaviour: The BCF specification states that `Filename` should correspond to the filename as recorded in the IFC file's STEP header (`FILE_NAME` in the IFC physical file). In practice, most authoring tools write the operating system filename instead. These two values frequently diverge:

- IFC `FILE_NAME` is set at export time and often reflects an internal project name, version string, or tool-generated identifier.
- The OS filename may have been renamed, versioned, or copied since export.

Additional dimension — serialisation format: The `FILE_NAME` anchor assumes IFC-SPF (STEP Physical File) serialisation. When the referenced model is serialised as ifcXML (`.ifcxml`), ifcJSON (`.ifcjson`), or a future format, there is no STEP header and therefore no `FILE_NAME` equivalent. It is unclear which value should populate `Filename` in these cases — the OS filename is the only candidate, but that re-introduces the instability described above.

Additional dimension — file extension: The specification does not state whether the file extension (e.g. `.ifc`, `.ifczip`, `.ifcxml`) should be included in the `Filename` string or omitted. In the official BCF test cases the value is written without extension (e.g. `BCF-ARK` rather than `BCF-ARK.ifc`). In participant submissions from BIM Fit Check, both conventions were observed. This creates an additional axis of mismatch: a tool writing `building.ifc` will not match a tool writing `building`, even when both refer to the same file.

Impact:

- Applications that attempt to match BCF references to open IFC files using `Filename` may fail to find the correct file.
- No interoperability guarantee: one tool writes the IFC header name, another writes the OS name — recipients cannot reliably resolve which file is meant.
- The "correct" interpretation is underspecified; both readings are defensible.
- For non-STEP serialisations, the spec's implied anchor (`FILE_NAME`) does not exist at all.
- The extension question adds a second independent axis of mismatch, compounding the ambiguity.

3.4 Official examples and Test Cases contain dead external IFC links

Elements: `Markup/Header/Files/File[@IsExternal="true"]/Reference`; `DocumentReference/Url`

Source: Official BCF example files and Test Cases in the bSI GitHub repository (`Test Cases/v3.0/Markup/`).

Observed behaviour: Multiple test case markup files contain `Header/Files/File` entries with `IsExternal="true"` whose `Reference` values point to IFC files that no longer exist. A full audit of all Markup test cases in `Test Cases/v3.0/` produced the following results:

Test Case	Header/File external reference	Status
Document reference internal	2 × bimsync.com revision URL	Dead
Document Reference External	`DocumentReference/Url` → buildingsmart-tech.org (old domain)	Dead
Due Date	2 × bimsync.com revision URL	Dead
Labels	2 × bimsync.com revision URL	Dead
Milestone	2 × bimsync.com revision URL	Dead
Minimum information	`<Files/>` element is empty	Clean
Related topics	2 × bimsync.com revision URL	Dead
User assignment	2 × bimsync.com revision URL	Dead
Orthogonal camera (Visualization)	No `<Reference>` — uses IfcProject GUIDs	Clean
Component selection (Visualization)	No `<Reference>` — uses IfcProject GUIDs	Clean

Six of eight Markup test cases share the same two dead `bimsync.com` revision URLs verbatim:

```
<File IsExternal="true">
 <Filename>BCF-ARK</Filename>
 <Date>2021-01-04T09:37:45.000Z</Date>
 <Reference>https://bimsync.com/project/
   f5fbb3c695274d1890036bf64f77eb71/revisions/
```

```
   007afab57f264d2296aae0a452486ae1</Reference>
</File>
<File IsExternal="true">
 <Filename>BCF-MEP</Filename>
 <Date>2017-08-07T09:51:34.000Z</Date>
 <Reference>https://bimsync.com/project/
   f5fbb3c695274d1890036bf64f77eb71/revisions/
   a21ed391f9e046a2bb2bc879c48f1d48</Reference>
</File>
```

bimsync.com was a BIM collaboration platform that has since been discontinued. Both URLs are permanently dead — no redirect, no archived copy, no way to recover the referenced content. The Document Reference External test case additionally links to `http://www.buildingsmart-tech.org/specifications/bcf-releases/bcfxml-v1/markup.x` an old buildingSMART domain that no longer serves this path.

Compounding factor: the bimsync.com references are not merely stale URLs to files that moved — they are URLs into a third-party proprietary service that no longer exists. The test cases were never self-contained.

Impact:

- Implementors using the official examples as test data get broken references out of the box.
- Automated test suites that fetch the referenced files will fail unconditionally.
- The dependency on a third-party commercial service as a reference anchor was fragile by design.
- Undermines trust in the canonical examples as reliable ground truth.

3.5 Author fields do not reference the `extensions.xml` user list

Elements:

- `Topic/CreationAuthor`
- `Topic/ModifiedAuthor`
- `Comment/Author`
- `Comment/ModifiedAuthor`

Related: `Topic/AssignedTo` (correctly references extensions.xml)

Observed behaviour: The BCF 3.0 specification (`extensions.xsd`) defines a `Users` list in `extensions.xml` alongside `TopicTypes`, `TopicStatuses`, `Priorities`, `Labels`,

`SnippetTypes`, and `Stages`. The `AssignedTo` element in the Topic nodes table correctly states: *"The user to whom this topic is assigned to. Recommended to be in email format (Predefined list in 'extensions.xml')"*.

However, the four author fields that also accept user identifiers from this same list omit any reference to it:

Element	Location	Current description
`CreationAuthor`	Topic nodes	"User who created the topic."
`ModifiedAuthor`	Topic nodes	"User who modified the topic. Exists only when Topic has been modified after creation."
`Author`	Comment nodes	"Comment author"
`ModifiedAuthor`	Comment nodes	"The author who modified the comment"

All four fields accept the same kind of value as `AssignedTo` — a user identifier, recommended to be in email format, drawn from the predefined `Users` list in `extensions.xml`.

Impact:

- Implementors reading the spec for the first time will not know that `CreationAuthor`, `ModifiedAuthor`, and `Author` are constrained by the same extensions list as `AssignedTo`.
- Applications may populate these fields with arbitrary strings instead of validating against the extensions user list.
- Validators and schema-aware tools cannot apply consistent rules across author fields without this documentation signal.
- The inconsistency creates ambiguity: is `AssignedTo` the *only* user field backed by extensions.xml, or are all user fields? The spec does not say.

3.6 `TopicStatus` description duplicates `TopicType` in the specification

Element: `Topic/@TopicStatus`, `Topic/@TopicType`

Source: BCF-XML 3.0 Documentation — Topic attribute table (README.md#topic)

Observed behaviour: The Topic attribute table contains the following two rows:

Attribute	Optional	Description
TopicType	No	Type of the topic (Predefined list in "extensions.xml")
TopicStatus	No	Type of the topic (Predefined list in "extensions.xml")

Both attributes carry the identical description: *"Type of the topic (Predefined list in 'extensions.xml')"*. The description for `TopicStatus` is a copy-paste error — it reuses the `TopicType` text verbatim.

Impact:

- Implementors reading the spec for the first time cannot infer the intended semantics of `TopicStatus` from the documentation alone.
- The error may cause confusion about whether the two attributes are interchangeable or serve the same purpose.
- Automated documentation tooling that validates description uniqueness would flag this as an inconsistency.

3.7 No Normative BCF Profile Equivalent to the IFC General Usage MVD

Element: BCF specification as a whole

Source: BIM Fit Check participant feedback — confusion about which rules are schema requirements and which are BIM Fit Check-specific

Observed behaviour: Throughout both the IFC and BCF rounds of BIM Fit Check, participants repeatedly asked which requirements stemmed from the BCF specification or schema and which were BIM Fit Check-specific additions imposed by the jury. The boundary was unclear, and this confusion affected how participants prioritised their implementation work.

For IFC, this distinction has a formal answer: the IFC schema defines what is technically valid, and the General Usage MVD defines the additional normative constraints that apply to a standard exchange — constraints that are either too complex to express in the schema or represent deliberate policy choices above schema level. A software tool claiming IFC conformance can be assessed against either the schema alone or the schema plus a named MVD.

BCF has no equivalent mechanism. The BCF schema (XSD) enforces what is machine-checkable, but the specification documentation adds constraints that cannot be expressed in XSD and have no formal normative status. A concrete example from BCF 3.0:

The `DocumentReferences` table in the specification (Documentation/README.md#documentreferences-optional) defines that `DocumentGuid` and `Url` are mutually exclusive: exactly one must be present, and which one applies determines whether the referenced document is embedded within the BCF container (`DocumentGuid`) or externally linked (`Url`):

Field present	Meaning
`DocumentGuid` only	Document is stored inside the BCF zip; GUID references the embedded file
`Url` only	Document is an external resource; URL points to it
Both present	Invalid — mutually exclusive
Neither present	Invalid — at least one required

This constraint is semantically clear and practically important — yet it is not encoded in the XSD schema and carries no explicit normative weight. It exists only as prose in a markdown table. During BIM Fit Check, some participants implemented the constraint, others did not, and the jury had no unambiguous specification reference to cite when flagging non-conformant files.

Impact:

- Participants cannot clearly distinguish schema conformance from specification conformance from BIM Fit Check conformance — three potentially different bars.
- Jury decisions on specification-level constraints are perceived as subjective rather than grounded in normative text.
- The absence of a normative BCF profile makes it impossible to define what "BCF conformance" means beyond XSD validity.
- Requirements added by BIM Fit Check that happen to align with documentation guidance cannot be distinguished from arbitrary jury preferences.

3.8 Camera position depends on BIM project origin under LoGeoRef50

Element: `VisualizationInfo/PerspectiveCamera`, `VisualizationInfo/OrthogonalCamera`

Source: BIM Fit Check participant submissions — BCF viewpoints across multiple software vendors

Observed behaviour: BCF viewpoints produced by different tools placed the camera at vastly different positions in world space when the same IFC model was opened, even when the viewpoints

nominally referenced the same area of the model. The root cause was found in the interaction between the BCF camera coordinate system and the LoGeoRef level of the underlying IFC file.

Under **LoGeoRef30** (the only mechanism available in IFC 2x3), georeferencing is carried as a property of the site placement. The camera position in a BCF viewpoint is expressed relative to the local project coordinate system — regardless of where the project origin sits in world space. Consequently, camera positions are consistently interpretable across tools: two tools opening the same file will place the camera at the same point in the model, irrespective of georeferencing.

Under **LoGeoRef50** (introduced with IFC 4 as the semantically preferred mechanism), the `IfcMapConversion` — or `IfcRigidOperation` — is applied to the project coordinate system. When tools honour this transformation and express BCF camera coordinates in the mapped (georeferenced) space, the camera position becomes relative to the project base point as transformed into the CRS. If the project base point is far from the model geometry — as is common when national CRS coordinates are used directly — the camera is displaced by the same offset, and the viewpoint is effectively broken when opened in a tool that interprets coordinates differently.

Impact:

- BCF viewpoints exported from a LoGeoRef50-aware tool cannot reliably be consumed by a tool with a different interpretation of the project base point.
- The issue was not observable in IFC 2x3 workflows (LoGeoRef30), which explains why it went undetected in earlier exchange rounds.
- The problem surfaces specifically in infrastructure and civil engineering projects, where national CRS coordinates introduce large offsets.
- No guidance in the BCF specification addresses the interaction between BCF camera coordinates and IFC georeferencing levels.

3.9 `.bcf` File Extension Introduced Without Normative Force

Element: BCF container file

Source: BCF-XML specification documentation — BCF File Structure

Observed behaviour: BCF files submitted during BIM Fit Check used both `.bcfzip` and `.bcf` extensions interchangeably, including for BCFv2.1 and BCFv3.0 content.

The BCF specification introduced the `.bcf` extension starting with version 2.1. The documentation states: *"A BCF file is a zip containing one folder for each topic with its file extension 'bcfzip' for BCFv1.0 and BCFv2.0. The file extension 'bcf' is introduced since BCFv2.1."* The verb "is introduced" carries no normative weight — it is a descriptive statement, not a requirement. No MUST, SHALL, or SHOULD accompanies the introduction of the new extension.

As a consequence, both `.bcfzip` and `.bcf` remain permissible for BCFv2.1 and BCFv3.0 containers. Tools export BCFv3.0 files with `.bcfzip` extensions without violating any stated rule. The extension alone is no longer a reliable indicator of BCF version, and the intended migration from `.bcfzip` to `.bcf` has not materialised across the ecosystem. The BIM Fit Check jury had no normative basis to flag `.bcfzip` usage as non-conformant for participants submitting BCFv3.0 content.

This finding is distinct from finding 3.3, which concerns the filename string inside the BCF markup; this finding concerns the extension of the BCF container file itself. See Section 4.9 for proposal.

3.10 Documents Registered in `documents.xml` Without Topic Reference

Elements: `documents.xml`; `Markup/DocumentReferences/DocumentReference/DocumentGuid`

Observed behaviour: During BIM Fit Check, one participant placed files in the `Documents/` folder and registered them in `documents.xml`, but no topic's `DocumentReferences` contained a `DocumentGuid` pointing to these files. The documents were present and correctly catalogued at the container level, but unreachable through any topic.

Stated purpose vs. schema reality:

The specification introduces `DocumentReferences` as follows: *"DocumentReferences provides a means to associate documents with topics."* This framing implies that documents are placed in the container *in order to be associated with topics*. However, the schema makes `DocumentReferences` optional at the topic level (`minOccurs="0"`), and places no constraint requiring that every entry in `documents.xml` be referenced by at least one topic. The result is a legal but semantically ambiguous state: a registered document with no topic association.

Impact:

- **Discoverability is effectively zero.** Every BCF tool surfaces documents in the context of a topic — the user opens a topic, sees its `DocumentReferences`, and navigates to the linked file. There is no standard mechanism for browsing the `documents.xml` catalog independently. An unreferenced document is invisible in any normal BCF workflow.
- **Intent is indistinguishable.** A validator or receiving application cannot determine whether an unreferenced document was intentionally added as an exchange-level resource or is the result of a broken or forgotten `DocumentReference` in a topic.
- **Round-trip safety is undefined.** The specification does not prescribe how applications must handle documents that are registered but unreferenced. An application that strips unreferenced `documents.xml` entries on import — treating them as orphans — would not violate any stated rule.

- **No defined use case for exchange-level attachments.** The scenario of adding a document that applies to the whole exchange (not to any individual topic) has no explicit home in the BCF data model. The spec neither provides for it nor prohibits it.

3.11 Point Markers Represented as Degenerate Lines

Element: `VisualizationInfo/Lines/Line`

Source: BIM Fit Check participant feedback

Observed behaviour: The BCF specification defines `Line` elements with a `StartPoint` and an `EndPoint`, both of type `Point` (X, Y, Z coordinates). The specification states: *"Lines that have the same start and end points are to be considered points and may be displayed accordingly."* In practice, this convention is used to mark a specific 3D position in a viewpoint — a point marker — by writing a `Line` whose `StartPoint` and `EndPoint` are identical.

Problem: A `Line` is geometrically and semantically a line segment. Using a degenerate line (zero length) as a point marker is a repurposing of an existing element, not a first-class representation. The workaround is:

- **Semantically ambiguous** — a `Line` with identical start and end points is formally invalid geometry. The spec acknowledges this case with a behavioural note, but a note is not a design.
- **Implementation-fragile** — tools that process `Line` elements for rendering or export may treat zero-length lines as degenerate input and drop them, deduplicate them, or raise errors, depending on the underlying geometry library.
- **Not self-documenting** — an implementor reading the schema sees a `Line` type with two `Point` children and no indication that this doubles as a point marker. The point-marker semantics are buried in a prose note, invisible to schema-driven code generation.
- **No dedicated type** — the `Point` complex type already exists in `visinfo.xsd` (used as the coordinate type for `StartPoint` and `EndPoint`). A dedicated `Points/Point` element parallel to `Lines/Line` would require no new coordinate type — only a new container and element name.

3.12 No Defined Immutability Taxonomy for Fields and Imported Content

Elements: `Topic`, `Comment`, `Viewpoint`; fields including `Guid`, `CreationDate`, `CreationAuthor`, `ModifiedDate`, `ModifiedAuthor`

Source: BIM Fit Check participant feedback

Observed behaviour: BCF exchanges pass through multiple systems — authoring tools, review platforms, coordination hubs — and are routinely imported, modified, and re-exported. The

specification provides no systematic statement of which fields and elements are immutable once created. The single existing immutability rule concerns viewpoints:

> *"Viewpoints are immutable, therefore they should never be changed once created. If new comments on a topic require different visualization, new viewpoints should be added."*

No equivalent rule exists for any other element. In particular:

- `Topic/Guid` and `Comment/Guid` are identity keys used to correlate records across systems. The spec does not state they must never change.
- `Topic/CreationDate`, `Topic/CreationAuthor`, `Comment/Date`, `Comment/Author` record the provenance of a record. The spec does not state they are set once and frozen.
- `ModifiedDate` and `ModifiedAuthor` exist to record changes — but the spec does not define which fields, when changed, must trigger an update to these fields.
- Imported content (topics and comments originating in another system) has no defined immutability status. A receiving system may allow users to edit or delete comments authored externally. The spec neither permits nor forbids this.

Impact:

- **Audit trail integrity is undefined.** If a receiving system modifies `Comment/Author` or `Comment/Date` on an imported comment, the authorship record is silently corrupted. No normative rule prevents this.
- **Round-trip consistency breaks.** When a BCF file is exported, imported, modified, and re-exported, the exporting system cannot know whether `CreationAuthor` or `Guid` values have been altered in transit.
- **Interoperability assumption mismatch.** One system treats imported comments as read-only; another allows free editing. Both behaviours are spec-conformant. The result is conflicting assumptions between systems in the same exchange workflow.
- **No basis for validation.** Without a defined immutability list, validators cannot flag fields that have been improperly modified — there is no normative standard to check against.

4 Improvements

4.1 `relatedTopic` references non-existing topic

Addresses finding 3.1.

Recommendation:

Option **(b) + optional (c)** appears most appropriate:

- **Always inform the user** — silently dropping data (a) is risky; the user may not know the relation existed. A clear warning preserves intent and allows manual recovery.
- **Attempt cross-file resolution (c) as a quality-of-life enhancement** — if the application maintains its own issue list (e.g. a project-wide BCF database), it can offer to re-link the orphaned reference. This should be opt-in, not automatic.
- **Never silently delete** — data loss without user consent is always the wrong default.

Suggested validation severity: Warning (not error) — the BCF file remains structurally valid; the broken link is a data quality issue, not a schema violation.

Spec gap: The BCF standard does not currently prescribe behaviour for broken `relatedTopic` references.

4.2 `Header/File` date precedes IFC file creation

Addresses finding 3.2.

Recommendation:

- **Prefer omission over fabrication** — if the schema allows `Date` to be absent, omitting it is cleaner than writing a date that is semantically wrong. A missing date is honest; a wrong date is misleading.
- **If omission is not schema-valid:** write the BCF export date and document it as "BCF creation date" in a comment or via a dedicated attribute — not as a stand-in for the IFC file date.
- **If no IFC file is linked at all:** the entire `Header/File` element should be omitted rather than populated with placeholder data.

Suggested validation severity: Warning — flag when `Header/File/@Date` predates the topic `CreationDate`, or when a `Date` is present but no corresponding file reference exists.

Spec gap: The BCF specification does not address the case where BCF precedes the IFC file in the workflow. The `Header/File` semantics assume the IFC file already exists. A clarification or an explicit "not yet linked" state would be valuable.

4.3 `Header/File/@Filename` ambiguity

Addresses finding 3.3.

Recommendation:

- **Spec must clarify** which value is authoritative — the current wording is ambiguous in practice even if technically it points to `FILE_NAME`. A normative statement is needed.
- **Preferred resolution:** mandate the IFC `FILE_NAME` header value as the canonical source, since it is stable regardless of how the OS file is later renamed or moved. Document this explicitly with an example.
- **Implementor guidance:** tools should additionally expose the OS filename as a human-readable label (distinct from the `Filename` attribute) so users can identify the file without understanding the IFC STEP header.
- **Fallback rule:** if the IFC `FILE_NAME` is empty or blank (which is valid in IFC), fall back to the OS filename and flag this in validation as an informational notice.

Spec gap (1): The spec does not account for the common case where `FILE_NAME` is empty or meaningless (e.g. auto-generated GUIDs from some exporters). Needs a defined fallback hierarchy.

Spec gap (2): The spec implicitly assumes IFC-SPF serialisation. For ifcXML, ifcJSON, and future serialisations, the `FILE_NAME` anchor does not exist. The spec needs an explicit rule for non-STEP formats — the OS filename is the pragmatic fallback, but this should be normative, not assumed.

Spec gap (3): The spec does not state whether the file extension should be part of the `Filename` string. The official test cases omit it (`BCF-ARK`, not `BCF-ARK.ifc`), but this is not stated as a rule. The spec should explicitly mandate one convention:

- **Recommended: include the extension.** The extension encodes the serialisation format (`.ifc`, `.ifczip`, `.ifcxml`) and is required for unambiguous file resolution — particularly when multiple serialisations of the same model exist in the same project. Matching logic in consuming tools should treat `building.ifc` and `building` as distinct values and not attempt silent normalisation.

4.4 Fix dead external IFC links in official examples and Test Cases

Addresses finding 3.4.

Recommendation: replace all dead external `Reference` URLs in the Test Cases and example files with self-contained alternatives.

Preferred resolution — internal references:

Replace `IsExternal="true"` entries with `IsExternal="false"` and bundle minimal placeholder IFC files (or the actual project IFC files if obtainable and licensed) directly in the test case folder. This makes each test case fully self-contained and immune to link rot:

```
<File IsExternal="false">
 <Filename>BCF-ARK</Filename>
 <Date>2021-01-04T09:37:45.000Z</Date>
 <Reference>BCF-ARK.ifc</Reference>
</File>
<File IsExternal="false">
 <Filename>BCF-MEP</Filename>
 <Date>2017-08-07T09:51:34.000Z</Date>
 <Reference>BCF-MEP.ifc</Reference>
</File>
```

If external references are needed to demonstrate the `IsExternal="true"` feature: replace third-party URLs with stable, persistent URLs under the `github.com/buildingSMART` or `standards.buildingsmart.org` domain, where link persistence can be guaranteed by the bSI organisation.

Broader principle: Test Cases in the BCF-XML repository should never depend on URLs hosted by third-party commercial services. Commercial services change, rebrand, and shut down. All external references in official test material should resolve to resources under bSI's own infrastructure or to archived, content-addressed locations (e.g. a pinned GitHub release asset).

Specific fixes required (from 3.4 audit):

Test Case	Fix required
Document reference internal	Replace 2 × bimsync.com `Reference` with internal IFC files
Document Reference External	Replace `DocumentReference/Url` buildingsmart-tech.org path with a current bSI URL
Due Date	Replace 2 × bimsync.com `Reference` with internal IFC files
Labels	Replace 2 × bimsync.com `Reference` with internal IFC files
Milestone	Replace 2 × bimsync.com `Reference` with internal IFC files

Related topics	Replace 2 × bimsync.com `Reference` with internal IFC files
User assignment	Replace 2 × bimsync.com `Reference` with internal IFC files

4.5 Author fields do not reference the `extensions.xml` user list

Addresses finding 3.5.

Recommendation: add the extensions.xml reference and the email-format recommendation to all four author fields, matching the wording already used for `AssignedTo`.

Proposed corrections — Topic nodes table:

Element	Optional	Current description	Proposed description
CreationAuthor	No	User who created the topic.	User who created the topic. Recommended to be in email format (Predefined list in "extensions.xml").
ModifiedAuthor	Yes	User who modified the topic. Exists only when Topic has been modified after creation.	User who modified the topic. Recommended to be in email format (Predefined list in "extensions.xml"). Exists only when Topic has been modified after creation.

Proposed corrections — Comment nodes table:

Element	Optional	Current description	Proposed description
Author	No	Comment author	Comment author. Recommended to be in email format (Predefined list in "extensions.xml").

ModifiedAuthor	Yes	The author who modified the comment	The author who modified the comment. Recommended to be in email format (Predefined list in "extensions.xml").

Rationale: the `Users` list in `extensions.xml` exists precisely to define the set of valid user identifiers for a project. All fields that accept a user identifier should acknowledge this list. The email-format recommendation is already established by `AssignedTo` and should be applied uniformly.

Suggested validation severity for non-conforming values: informational notice — values outside the extensions user list are not schema-invalid, but a linter may flag them to help projects maintain consistency.

4.6 `TopicStatus` description duplicates `TopicType`

Addresses finding 3.6.

Recommendation: correct the description of `TopicStatus` in the Topic attribute table.

Proposed correction:

Attribute	Optional	Description
TopicType	No	Type of the topic (Predefined list in "extensions.xml")
TopicStatus	No	**Status of the topic (Predefined list in "extensions.xml")**

The word "Type" in the `TopicStatus` row should read "Status". The two attributes serve distinct purposes:

- `TopicType` classifies *what kind* of topic it is (e.g. Issue, Request, Clash, Remark).
- `TopicStatus` describes *the current state* of the topic's lifecycle (e.g. Open, In Progress, Resolved, Closed).

4.7 Define a Normative BCF Profile Equivalent to the IFC General Usage MVD

Addresses finding 3.7.

Recommendation: buildingSMART should define a **BCF General Usage Profile** — a normative document that collects all constraints on BCF content that are stated in specification prose but not encoded in the XSD schema. This profile would serve the same function as the IFC General Usage MVD: it defines what a standard, interoperable BCF exchange looks like, above and beyond bare schema validity.

Scope of the profile: The profile should capture, at minimum:

- Mutual exclusivity constraints such as `DocumentGuid` / `Url` on `DocumentReference`
- Required/forbidden field combinations not expressible in XSD
- Ordering or cardinality rules stated only in prose
- Any constraint currently enforced by BIM Fit Check that derives from specification intent rather than jury policy

Effect on BIM Fit Check: Once a BCF General Usage Profile exists, BIM Fit Check criteria can be stated precisely as one of: 1. **BCF schema conformance** — XSD validity only 2. **BCF General Usage Profile conformance** — schema + normative prose constraints 3. **BIM Fit Check extension** — additional requirements specific to the exchange scenario being tested

This three-tier framing eliminates the ambiguity participants encountered and gives the jury a clear normative anchor for every requirement it enforces.

Good candidates for the profile already exist in the BCF 3.0 documentation. The `DocumentReferences` table cited in finding 3.7 is one example; a systematic audit of the specification prose would surface more.

4.8 Camera position depends on BIM project origin under LoGeoRef50

Addresses finding 3.8.

Recommendation:

The BCF specification should define which coordinate space BCF camera positions are expressed in, and how that choice interacts with IFC georeferencing levels.

Two approaches are viable:

> **Option A — Local project coordinates (LoGeoRef-agnostic):** BCF camera positions are always expressed in the local IFC project coordinate system, before any `IfcMapConversion` or `IfcRigidOperation` is applied. This preserves compatibility with IFC 2x3 / LoGeoRef30 workflows and ensures viewpoints are independent of georeferencing choices. The IFC-to-BCF pipeline must explicitly strip any CRS offset before writing camera coordinates.

Option B — Georeferenced coordinates with explicit declaration: BCF camera positions may be expressed in the mapped CRS space, provided the BCF file declares which LoGeoRef level and which CRS the coordinates refer to. Consuming tools must apply the inverse transformation when the IFC file uses a different level. This approach is more powerful but requires coordinated support from both exporting and importing tools.

Recommendation: Option A is preferable in the near term for maximum interoperability. A declaration mechanism (Option B) should be defined as an optional extension for applications that require georeferenced BCF workflows.

Spec gap: Neither BCF 2.1 nor BCF 3.0 addresses the interaction between BCF viewpoint coordinates and IFC georeferencing. A normative clarification or a new optional attribute (e.g. `CoordinateSpace` on `VisualizationInfo`) should be introduced.

4.9 Strengthen Normative Language for the `.bcf` File Extension

Addresses finding 3.9.

The BCF specification should replace the descriptive statement *"the file extension 'bcf' is introduced since BCFv2.1"* with an explicit normative requirement. A minimal change sufficient to drive ecosystem migration:

> *BCF files conforming to BCFv2.1 or later SHOULD use the `.bcf` file extension. The `.bcfzip` extension is deprecated for these versions and SHOULD NOT be used for new files.*

A stronger formulation using SHALL rather than SHOULD would be appropriate if the intent is to make the extension a conformance criterion rather than a recommendation. The weaker SHOULD is proposed here as a pragmatic first step that does not immediately break existing tooling, but signals unambiguously that `.bcfzip` is no longer the preferred form.

This change requires no schema modification — only a one-sentence update to the specification prose — and would give conformance-checking tools and BIM Fit Check juries a clear normative basis for flagging `.bcfzip` containers in BCFv2.1+ exchanges.

4.10 Clarify the Scope and Discoverability of `documents.xml` Entries

Addresses finding 3.10.

Recommendation:

The BCF specification should resolve the ambiguity by choosing one of two positions and stating it explicitly:

Option A — Require topic association:

Every document registered in `documents.xml` must be referenced by at least one topic via `DocumentGuid`. Unreferenced entries are non-conformant. This aligns with the stated purpose of `DocumentReferences` (*"a means to associate documents with topics"*) and eliminates the orphan state. Tools must validate and warn when a `documents.xml` entry has no corresponding `DocumentGuid` in any topic markup.

Option B — Define an exchange-level document concept:

Explicitly permit documents that are not topic-scoped. Introduce a mechanism — either a flag in `documents.xml` (e.g. `ExchangeLevel="true"`) or a dedicated section in the BCF container (e.g. `exchange-documents.xml`) — for documents that apply to the exchange as a whole rather than to individual topics. Tools must expose these documents through a container-level view, not only through topic detail panels.

Recommendation: Option A is preferable in the near term. It closes the ambiguity without requiring schema changes — only a prose clarification and an optional validation rule. If a genuine use case for exchange-level documents is identified, Option B can be introduced in a subsequent revision with a defined UI contract.

Minimum required change (Option A): add one sentence to the `documents.xml` description:

> *Every document listed in `documents.xml` must be referenced by at least one `DocumentReference/DocumentGuid` element in a topic markup file. Documents with no topic reference are non-conformant.*

4.11 Introduce a Dedicated `Points` Element for Point Markers

Addresses finding 3.11.

Recommendation:

Introduce a `Points` element in `VisualizationInfo`, parallel to the existing `Lines` element, containing one or more `Point` children. Each `Point` carries a 3D coordinate (X, Y, Z) and represents a discrete position marker in the viewpoint.

Proposed schema addition to `visinfo.xsd`:

```
<xs:element name="Points" minOccurs="0">
 <xs:complexType>
   <xs:sequence>
     <xs:element name="Point" type="Point" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
 </xs:complexType>
</xs:element>
```

The `Point` complex type already exists in `visinfo.xsd` — no new type definition is required.

Migration path: The existing degenerate-line convention should be deprecated but retained for backward compatibility. Tools reading BCF files should continue to recognise `Line` entries with identical `StartPoint` and `EndPoint` as point markers. Tools writing BCF files should use the new `Points/Point` element. A transitional note in the spec:

> *Point markers should be expressed using `Points/Point`. The convention of representing a point marker as a `Line` with identical `StartPoint` and `EndPoint` is deprecated and retained only for backward compatibility with files produced by earlier implementations.*

Spec gap: The current specification has no first-class point marker element. The degenerate-line workaround, while documented, is an accidental feature rather than a deliberate design.

4.12 Define an Immutability Taxonomy for BCF Fields and Imported Content

Addresses finding 3.12.

Recommendation:

The BCF specification should define a systematic immutability taxonomy, classifying every field and element in one of three categories:

Category	Definition	Examples
Immutable	Set once at creation; must never be changed thereafter	`Topic/Guid`, `Topic/CreationDate`, `Topic/CreationAuthor`, `Comment/Guid`, `Comment/Date`, `Comment/Author`, all `Viewpoint` content
Mutable (tracked)	May change; every change must update `ModifiedDate` and `ModifiedAuthor`	`Topic/Title`, `Topic/Description`, `Topic/AssignedTo`, `Topic/TopicStatus`, `Comment/Comment`
Mutable (untracked)	May change without modification tracking	`Topic/Index`, display-only metadata

On imported content:

The specification should state that content imported from an external source (i.e., content whose `CreationAuthor` belongs to a different system or user) is treated as immutable by the receiving

system by default. Specifically:

> *Comments imported from an external BCF source must not be edited or deleted by the receiving system. A receiving system may append new comments to a topic but must not alter comments it did not originate.*

This rule preserves the audit trail across system boundaries and aligns with how BCF is used in practice: as a shared, append-only log of coordination activity, not a freely editable shared document.

Minimal required change: extend the immutability statement currently applied only to viewpoints into a general section — *"Field immutability and modification rules"* — covering Topic, Comment, and Viewpoint consistently.

Note: The `ModifiedDate` / `ModifiedAuthor` trigger rules (which field changes must cause a modification record) are a related gap. These should be defined in the same section.

5 General Remarks

5.1 Standard Definition, Best Practices, and Industry Convention

A recurring tension throughout the BIM Fit Check BCF round was the interplay between three distinct reference points: the *standard definition* as specified in the BCF schema and its documentation on GitHub; *best practices* grounded in common sense and process knowledge; and *industry convention* — what the majority of software providers have already implemented and shipped.

These three are frequently misaligned, and the misalignment creates dilemmas where no solution is fully satisfactory. Adopting a best practice, for instance, may conflict with established industry conventions that participants rely on for compatibility. Industry conventions themselves sometimes deviate from the formal standard definition — leaving implementors to choose between strict conformance and compatibility with what others have already done.