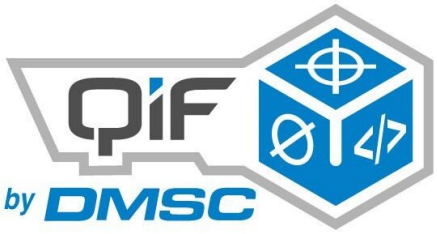



# Recommended Practices

## for

# Cross-Domain Exchange for Downstream Uses

***Release 1.0***

January 29, 2026

**CAx-IF**

**Jochen Boy**
PROSTEP AG
jochen.boy@prostep.com

**Pascal Hubert**
AFNeT
pascal.hubert@afnet-services.fr

**Jamie Yedinak**
The Boeing Company
jamie.a.yedinak@boeing.com

**Technical**

**Asa Trainer**
Trainer Engineering Associates
agtrainer@comcast.net

**Tom Thurman**
TR Thurman Consulting
thomas.r.thurman@imonmail.com

**Larry Maggiano**
Mitutoyo America Corp
Larry.Maggiano@mitutoyo.com

**Tom Kramer**
Thomas Kramer Consulting LLC
tomrkramer2@gmail.com

# Table of Contents



# List of Figures



# List of Tables

## Document History

This document is a new CAx-IF Recommended Practice and adds new constructs.

| Release | Date | Change |
|---|---|---|
| 0.1 | 2025-07-19 | Initial Author's DRAFT |
| 0.2 | 2025-07-21 | Second Author's DRAFT |
| 0.3 | 2025-08-19 | Third Author's DRAFT |
| 0.4 | 2025-08-27 | 4th DRAFT, INTERNAL, planned for SME reviews (ASAP); new Figures 1 and 2; cleaned up QIF examples in Section 5.4; Table 1 needs work. |
| 0.5 | 2025-08-28 | 5th DRAFT, INTERNAL, fixed QIF structures, incorporated Larry's comments; added Peter Rau to the acknowledgements; added new Table 1; scope clarifications added. |
| 0.6 | 2025-09-30 | 6th DRAFT, final edits before SME and DMSC reviews; comments resolved, added inline comments in QIF Example (Section 5.4), revised Figures 1 and 2, added Appendix C. |
| 0.7 | 2025-10-04 | 7th DRAFT, additional edits; |
| 0.8 | 2025-10-21 | 8th Draft; updated TOC, LOF, etc. Updated Section 3, Updated Table 1, Corrected Section 5.4, Added Section 5.5 and Figure 3; Updated Appendix B and revised its links. |
| 0.9 | 2025-10-28 | 9th Draft; updated QIF Example (section 5.4) to eliminate entity usage violation, changed comments to Description entities, removed unnecessary Attributes |
| 0.95 | 2025-12-17 | Pre-Approval DRAFT; updated to include reviewer comments and editorial corrections; Also updated reference example data files and the URL locations for those data files. Released to DMSC for Board approval. |
| 1.0 | 2026-01-29 | DMSC Board technical review complete and RP approved; renumbered previous draft to 0.95; Added DMSC co-branding logos and copyrights; Added DMSC Approval Section. Additional minor editorial changes. |

## DMSC Approval

The DMSC Executive Board and QIF SMEs have reviewed and approved this release of the Recommended Practice for Cross Domain Exchange, v1.0, to facilitate interoperability between partners and systems using STEP AP242 Edition 4 and QIF version 3.0 for integrated Model Based Enterprise. Approval applies only to this version of QIF; subsequent versions will require review and approval by the DMSC Executive Board.

## Acknowledgements

The authors wish to thank Bob Stone, Origin International, Ben Urick, nVariate, Jan de Nijs and Sam Gambrell, Lockheed Martin, and Peter Rau, Mitutoyo, for their subject matter expertise and technical review of this Recommended Practices document. The authors also wish to thank Jochen Boy, ProSTEP, for his editorial review and recommendations.

The authors also wish to thank Daniel Campbell of rubypoint for his technical review on behalf of the DMSC Board and Jennifer Herron of Action Engineering for her liaison efforts between PDES Inc and DMSC on this project.

# 1 Introduction

Research into mechanisms for maintaining traceability of engineering product data during exchange has concluded that the introduction and tracking of persistent IDs in that product data are feasible. Achieving traceability of individual data elements during state changes in product design and product development process can provide immediate practical benefits for several use cases. In this Recommended Practices document, we will focus on a particular use case – cross-domain exchange between design functions publishing MBD data via STEP AP242 Edition 4 with persistent IDs and downstream systems consuming that published STEP data and maintaining traceability via those persistent IDs. This initial release will focus on those downstream consuming systems (e.g. metrology systems) that produce Quality Information Framework (QIF) output (QIF Plans, QIF Results and QIF Statistics) containing persistent IDs.

## *1.1 Downstream Exchange*

Passing of design data to downstream systems has similar needs to maintain associative references between, for example, manufacturing tool paths and the model entities they are derived from or dimensional characteristics as planned and measured in metrology systems and those same dimensional characteristics as defined in the original design systems where the models were created. In addition to the challenges of controlling and managing change in downstream systems, based on model state change in designs, there is a second and perhaps more burdensome requirement for downstream consumption of model data, and that is traceability. Traceability is necessary from the original system of creation of data through all its uses and throughout its lifecycle to follow potential issues or faults to their source - either design, manufacture, material, or process. Keywords to note are – as designed, as required, as planned, as measured, as reported – and establish critical contextual information for the collaborator.

The necessity for traceability is two-fold. The first reason for traceability is to provide a mechanism for linking related elements: for example, requirements, characteristics, physical or virtual artifacts or features; or analyses of those artifacts or features within or between product development documents or databases. This traceability is needed over the entire lifecycle of the product being developed, once delivered into service, and ultimately, when the product reaches its end of life and is disposed of. Traceability enables rapid accessibility to chains of related product content and can, for example, support identification of critical characteristics, ensuring that requirements are met, and allowing feedback to stakeholders.

The second reason for traceability is as a necessary and required forensic tool to be used in legal proceedings and technical investigations where complex product or process failures may have resulted in injury or death. In particular, the manufacturing and metrology communities have long had procedures in place to tag and maintain identification of products and their components, constituent entities, and attributes, to maintain this traceability and ensure their ability to isolate and identify any suspect system characteristics or entities in the event of failure. The ability to rapidly trace through suspect systems can be critical to quickly rectifying dangerous problems in product designs or product development processes.

### *1.2 Maintenance of this Document*

This document shall be maintained collaboratively by the CAx-IF covering Part 21 based implementations with models containing persistent IDs in STEP and by the DMSC covering the equivalent within QIF based models containing persistent IDs[1]. In the current version, the document will focus on the metrology use case. A future version is expected to expand to address the manufacturing use case.

## 2 Scope

**The following are within the scope of this document:**

- The generation and use of Universally Unique Identifiers (UUIDs) (see Section 4) for maintaining associativity of entity instances within and between design and downstream systems.
- Advanced_brep_shape_representation is the STEP topological/geometric model supported by this document as a definitional product topological/geometric model.
- This document is intended to support exchange between STEP AP242 Edition 4 and QIF version 3.0.
- The approach initially used for testing was to copy all the STEP mapping data and both FeatureNominal and CharacteristicNominal data from the Plan file into the Results file. This method is illustrated in the Sections of this document and example data is provided (see Appendix B and links in B-1). Though this approach is valid and traceability was successfully demonstrated, it is inefficient. A more efficient method is to have external references in the Results file that point to entities in the Plan file (i.e. two separate files, the second of which is dependent on (has references to) entities in the first file). In the long (and medium) run for real-world applications, using references from Results to Plans rather than duplicating data is by far the better solution. Though this second approach is not documented in the body of the Recommended Practice, example data for this second approach is also provided (see Appendix B and links in B-2).

[1] The original draft STEP AP242 Edition 4 schema as well as the original CAx-IF recommended practices document for persistent IDs in STEP referred to them as Globally Unique Identifiers (UUIDs). The current version of the QIF standard (v3.0) refers to persistent IDs as QPids. The CAx-IF and DMSC have come to an agreement that, to avoid confusion and to foster harmonization, both standards (STEP in the final AP242 Edition 4 schema and QIF beginning in QIF 4.0) and this recommended practice will refer to such persistent IDs as Universally Unique Identifiers (UUIDs).

**The following are out of scope for this edition of the document:**

- The use of the STEP Part 21 Edition 3 Anchor Section for assigning UUIDs after the original publishing of the STEP document [Future scope][2].
- Feedback paths back to design [Future scope see 5.3].
- Detailed implementation recommendations for QIF pre and post processors[3].
- This document focuses on cross domain exchange between STEP and QIF only. Evaluating QIF files for quality compliance to QIF 3.0 is not considered in the context of this Recommended Practice. Readers are referred to Section 5.4 of the QIF 3.0 standard (Appendix A-3) for QIF quality compliance purposes.
- Mapping manufacturing features from STEP to QIF.
- Mapping process plans from STEP to QIF.
- Support for exchange between STEP AP242 Edition 4 and QIF version 4.0 is out of scope for this document version. The document will be revised to support QIF 4.0 once that version is released.

# 3 Document Identification

Details of the generation and assignment of UUIDs in STEP are presented in the CAx-IF Recommended Practices for Persistent IDs for Design Iteration and Downstream Exchange[4]. Readers are encouraged to refer to that Recommended Practices document for further information.

Downstream applications consuming STEP data in the context of this document require the application of persistent identifiers in the STEP data per the above Recommended Practices document. Identification of conforming STEP files shall be determined by referring to FILE_DESCRIPTION entity in the STEP file as defined in Document Identification section (Section 3) of that Recommended Practice.

# 4 Definitions

Entity instance:

STEP P21 file: a lexical record in the data section

QIF: an XML element or attribute

[2] The Use of Part 21 Edition 3 Anchor Section approach to UUID assignment precludes any modification of an existing STEP file for this edition of this document.

[3] This document does not provide a mapping from STEP attributes to QIF attributes. The QIF snippets provided in this document are those in scope for the interoperability studies supported by this document. Items shown in bold in the QIF snippets provided are generated by the QIF application to establish cross-references to the reference STEP product model.

[4] Cax-IF Recommended Practices for Persistent IDs for Design Iteration and Downstream Exchange, Release 1.70, October 29, 2025 at the time of publication of this Recommended Practices document.

# 5 Persistent IDs

A mechanism for generating uniformly consistent, cross-application entity IDs is required for the traceability processes to work. These IDs need to be unique to prevent clashes between entity identifiers. In the issue summary for Jira Issue #TCSC10303-14[5] (See Appendix C) written by Tom Thurman (see also BRUTUS #23), the scope of such uniqueness was suggested as only being required within the context of a specific product.[6] This might be considered sufficient if only the first use case – design iteration – was required. In the context of the second use case, however, particularly the metrology use case, persistent (permanent), universally unique identifiers are necessary and need to be applied to product (as well as product effectivity, i.e. serialized product artifacts, if they exist), to individual semantic PMI entities (characteristics), to individual geometric features, and to model attributes in the product.

## *5.1 Formulation of Identifiers (UUIDs)*

Fortunately, persistent, universally unique identifiers have been in use in the information technology domain for a long time. Such a 'universally unique identifier' (UUID) is a 128-bit number used to identify information in computer systems, e.g. operating systems, databases, and communications processes. Note that the term 'globally unique identifier' (GUID) is also still found in the literature. UUIDs have been standardized by the Open Software Foundation (OSF) and are documented as part of ISO/IEC 11578:1996 "Information technology – Open Systems Interconnection – Remote Procedure Call (RPC)" and more recently in ITU-T Rec. X.667 | ISO/IEC 9834-8:2005. Most computing platforms provide convenient support for generating them, and for parsing their textual representation. More detail about UUIDs can be found on Wikipedia. Within the engineering domain, such GUIDs or UUIDs are already in place and being used in the Industry Foundation Classes (IFC) format of the Building Information Model (BIM) and in the Quality Information Framework (QIF) standard for the Metrology domain.

In the above standard there are 5 possible versions of UUIDs:

- Version 1 UUIDs are generated from a time and a node id (usually the MAC address)
- Version 2 UUIDs are generated from an identifier (usually a group or user id), time, and a node id
- Versions 3 and 5 produce deterministic UUIDs generated by hashing a namespace identifier and a name that must be unique in the scope of that namespace
- Version 4 UUIDs are generated using a random or pseudo-random number.

The QIF standard does not require or express a preference for any particular UUID version, if it is one of the standard five. As of the writing of this document, several QIF enabled applications are generating PIDs using UUID Version 4 i.e. via random numbers.

Based on research, there is some desirability to be able to consistently reproduce a UUID from some namespace and name data. Versions 3 and 5 allow this but the algorithm of Version 3 has been deprecated in favor of Version 5. Though there has been some discussion of registering a namespace specifically for these engineering information exchange use cases, and the namespace hierarchy might include the name and version of the generating preprocessor, no definitive plans have been put in place. The name data for the seed to generate this UUID version might be a fully qualified path from the product identifier to the entity in question. See the example below.

[5] Note that the Jira repository is a closed database of STEP-related issues under the control of ISO and access is limited to ISO Technical Committee members.

[6] The major focus of that effort was on the persistence of manifold solid brep models for a product across design iterations where many iterations were of data other than the solid model.

A version 5 UUID could be constructed, for example, from:

- A pre-defined namespace: uuid.NAMESPACE_DNS
  - For example: UUID('6ba7b810-9dad-11d1-80b4-00c04fd430c8')
- Model Identification String (SHA-1 hash of string) composed of

  Filename: “nist_ftc_06_asme1_nx900_rd.prt”

  Type: “PMI Feature Control Frame”

  Persistent ID: “ID 879819”

The function **uuid.uuid5**[7] with arguments **(uuid.NAMESPACE_DNS, "nist_ftc_06_asme1_nx900_rd.prt PMI Feature Control Frame - ID 879819")**

renders **UUID('491b0d21-fc8e-50d0-874f-6b7f5a95c47a')**.

It is important to note that, if all hash string elements are the same, a UUID generator using this or other similar UUID generation function will generate the same UUID again.

Selecting the Model Identification String (SHA-1 Hash String, stronger hash methods e.g. SHA-2 or SHA-3 or others can be employed in future implementations) is the responsibility of the owner (creator) application. The creator application shall concatenate string elements including the following:

- a unique part name or part number,
- a part instance serial number (if it exists),
- an entity type or full path from the product to the individual entity, and
- a persistent entity ID.

In the remainder of this recommended practice document, we will refer to the term UUID when describing these unique identifiers, irrespective of the method of generation (see footnote 7 below), and unless noted, we will be referring to type 5 (hash-based namespace/namestring) UUIDs.

## *5.2 IDs in STEP*

There are two valid approaches to storing UUIDs in STEP. The first is via an internal identifier within the context of the Part 21 data section. This method is preferred by data modelers and makes sense for the original publishers of STEP from the source CAx system and will be described in Section 4.2.1 below. This first method will be the focus for the current edition of the document.

The second is the storing of such identifiers not in the data section but rather within the Part 21 Edition 3 Anchor Section. It is expected that, in a future version of this document, this second method may be advanced as the appropriate method to add identifiers to a STEP file after it has been published, either to allow addition of identifiers to legacy STEP data or for those applications that intend to append new content to an existing STEP file where those applications are not the original creator of the file.

---

[7] Most operating systems, scripting languages, programming languages, and database applications support this particular UUID generation method. In the example shown, Python’s uuid module generates the type 5 uuid via the function call uuid.uuid5(). Other examples for this purpose include the java.util.UUID class methods, the Linux system util-linux package function uuidgen, and the Bash script’s uuidgen command. For a detailed survey of UUIDs refer to „Beyond Randomness: A Detailed Study on UUID Standards, Data Integrity, and Identifier Design Across Storage Systems“, A. R. Sinha, IJIRCT ISSN 2454-5988, 2023

## *5.3 Workflows between STEP and Downstream Systems*

Workflows between STEP and downstream systems and formats may be either forwards, i.e. downstream, or backwards, i.e. upstream, providing feedback paths (see Figure 1 below). The various feedback paths shown are facilitated by PID exchange across domain boundaries. The feedback paths are considered necessary and desirable but are out of scope for the current recommended practices document which focuses on feedforward workflows. Feedback workflow paths, using digital model-based feedback processes will be considered for future development[8].

The current version of this Recommended Practice document focuses on the metrology workflow and, specifically, the flow of product information, including PIDs, from STEP AP242 Edition 4 files to QIF Plan files and, from there to QIF Results files. Future versions of this Recommended Practice will expand to include the manufacturing workflow. The harvesting of near-real-time machining process data and metrology data via MTConnect process streams (shown in Figure 1) shows promise to inform and optimize process planning. This Recommended Practice will, however, limit focus to direct exchange of metrology execution data to metrology analysis (shown in Figure 2). Future versions of this Recommended Practice will expand to include real-time process streams via MTConnect.

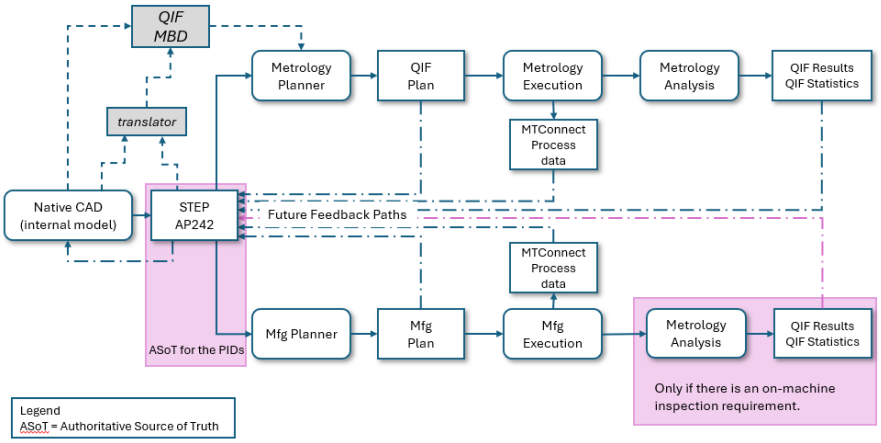


*Figure 1 - Design to Downstream Workflows, extended*

[8] A marked up drawing or model has been annotated with handwritten or digital marks, lines, shapes, or notes to communicate changes, feedback, questions, or instructions. This process, also known as redlining, is used in manufacturing to facilitate communication and quality control by providing efficient ways to suggest alterations to a design before it is finalized or built.

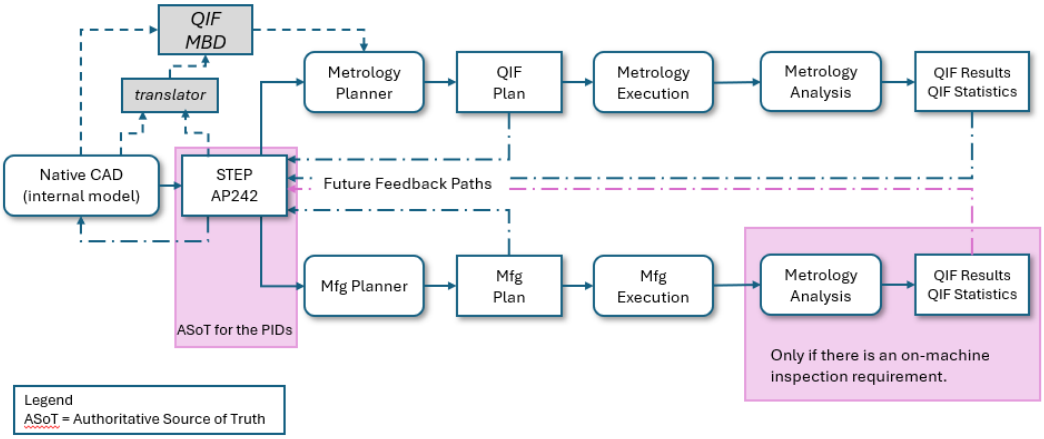


*Figure 2 - Design to Downstream Workflows, near-term*

The design user will publish (export) STEP AP242 Edition 4 data with UUIDs assigned to specific STEP entity instances as defined in Footnote 4 (previous page). Testing to date has restricted UUID assignments to a subset of the complete allowed list of entities. The UUIDs from this reduced subset of STEP entities shall be mapped to QIF elements as shown in Table 1 on the following page by the post-processor of the consuming application. Because providing the explicit reference path in STEP to a STEP data object is out of scope for this document, only the source and target data objects are provided.

| ***STEP entity*** | ***QIF element*** | ***Notes*** |
|---|---|---|
| ***PRODUCT***<br>*(via **PRODUCT_DEFINITION_SHAPE**)*<br>*PRODUCT.id* | *<Product><PartSet><Part><UUID>*[9]<br><br>*<Product><PartSet><Part><ModelNumber>* | *See Section 4.4.1 below* |
| ***PRODUCT_VERSION***<br>*(via **PRODUCT_DEFINITION_FORMATION***<br>*or*<br>***PRODUCT_DEFINITION_FORMATION_WITH_SPECIFIED_SOURCE**)* | *<Product><PartSet><Part><Version>* | |
| ***MANIFOLD_SOLID_BREP*** | *<Body> @form=SOLID* | |
| ***ADV_BREP_SHAPE_REP*** | *[Ignore]* | |
| *Topology to features, for feature recognition* | | |
| ***OPEN_SHELL or CLOSED_SHELL*** | *[ignore]* | |
| ***ADVANCED_FACE*** | *<SurfaceFeatureNominal> of appropriate sub-type* | *as appropriate for the type of ADVANCED_FACE imported; See Section 5.4.3 below* |
| ***EDGE_CURVE*** | *<LineFeatureNominal>, <CircularArcFeature-Nominal>, etc..* | *as appropriate for the type of EDGE_CURVE translated; See Section 5.4.3below* |

[9] Sequence of XML elements denotes a path through a QIF XML structure.

| | | |
|---|---|---|
| | *subtype* | |
| *PMI (model based, semantic) -* | | |
| *DATUM* | *Query of the Datum Feature instance to identify the Datum instance* | |
| ***DATUM_FEATURE**, not DATUM* | *<FeatureNominal> and <OtherNonShapeFeatureNominal> that are referenced by a <DatumDefinition>* | *See Section 5.4.3.1* |
| ***DIMENSIONAL_SIZE*** | *<DiameterCharacteristicDefinition> if dimensional_size is for a hole. Otherwise use an appropriate subtype of characteristic definition* | *See Section 5.4.3.2* |
| ***DIMENSIONAL_LOCATION*** | *<DistanceBetweenCharacteristicDefinition>* | |
| ***GEOMETRIC_TOLERANCE*** *(SubTypes –> e.g. POSITION_TOLERANCE)* | *<CharacteristicDefinition> of appropriate subtype* | *as appropriate for type of PMI entity translated; See Section 5.4.4* |
| ***TOLERANCE_ZONE** and Tol_Zone_Def* | *<CharacteristicNominal> of appropriate subtype, associated to the GeometricTolerance Characteristic Definition* | |
| ***DATUM_TARGET*** | ***<DatumTargetNominal>** with appropriate subtype* | |
| ***Derived DATUMs (per PMI Rec Prac)*** | *<DatumDefinition> This is a compound object in QIF so not detailed herein.* | |
| ***SURFACE TEXTURE PROPERTY and Surface Texture Parameter*** | *<Characteristic> subtype* | |
| ***Supplemental Geometry*** | *<FeatureNominal> subtypes (e.g.,* | |

| | | |
|---|---|---|
| ***(associated with PMI elements or as required for downstream purposes, construction geometry out of scope)*** | *PlaneFeatureNominal, CircleFeatureNominal)* | |
| ***User Defined Attributes***<br>***Descriptive (String), Value, and Measure***<br>***assigned to PROPERTY_DEFINITION & associated REP_ITEMs*** | *<AttributeStr>*<br>*<AttributeD1> or <AttributeI1>*<br>*<AttributeD1> or <AttributeI1> + <UnitType>* | *as appropriate for the type of Attribute entity imported* |

*Table 1 - Mapping of STEP entities to QIF elements*
*(for PID Referencing Only)*

## *5.4 QIF Structures for Persistent Identifiers*

Persistent Identifiers (UUIDs by that name or by the name QPIdType) in QIF are discussed in Section 5.13 of the QIF 3.0 Standard[10] and may be applied at several levels including Document, Part, Assembly, Datum, Feature, and Characteristic.

To support cross-domain exchange where STEP is the source for design data, QIF allows the ability to include references to that external data, including the capture of Persistent Identifiers (in the form of UUIDs) from that external data. For the purposes of this initial draft Recommended Practices document, these references to external data with UUIDs shall be included at the Part, Datum, Feature, and Characteristic level and are described below.

Note that, when there are differences between the QIF structure examples shown here and the QIF structures contained in the example data of Appendix B, the structures shown in the example data shall take precedence.

### 5.4.1 Part

An example of the use of a Part element with DefinitionExternal based on a DigitalModel includes the QIF Entity ID and the following elements -

- QIF Element EntityId shall be populated with the STEP UUID for the entity Product_Definition_Shape.[11]
- QIF Element Part Name shall be populated from the STEP entity Product.Name and
- QIF Element File Name shall be populated from the source STEP file name from which part data is retrieved.

This QIF 3.0 Part structure is shown below.[12]

```
<Product>
  <PartSet n="1">
    <Part id="114">
      <DefinitionExternal n="1" id="113">
        <DigitalModel id="9999">
          <Name>V5_Plate_w_Hole</Name>
          <File>
            <Name>pdi_0-e5-242.stp</Name>
            <Format>
              <OtherDigitalModelFormat>STEP_AP242e4</OtherDigitalModelFormat>
            </Format>
```

[10] QIF 3.0 – 2018 is published by DMSC and is available at no cost at https://qifstandards.org/download/; QIF 3.0 was designated an ANSI standard in 2020 and published by ISO as ISO23952:2020 in the same year. A new QIF version 4.0 document is in DRAFT. Please contact Dr. Tom Kramer, Project Manager for the QIF 4.0 Working Group at DMSC, for details on the QIF 4.0.

[11] In this document, there is exactly one product and one product_definition_shape in the STEP data file. In future editions of this document, the example will be extended to accommodate multiple products.

```
</File>
<Entities n="10">
 <Entity id="9998">
  <EntityId>427affa6-ae35-5e83-8002-18fa37cecad1</EntityId>
  <Description>STEP PRODUCT_DEFINITION_SHAPE</Description>
 </Entity>
 <Entity id="9997">
  <EntityId>edc2d377-65ac-56f0-b654-5932834935f8</EntityId>
  <Description>STEP DATUM_FEATURE [SAR] DATUM A</Description>
 </Entity>
 <Entity id="9996">
  <EntityId>c61ef055-f208-55c2-befc-8a97f1a87289</EntityId>
  <Description>STEP DATUM_FEATURE [SAR] DATUM B</Description>
 </Entity>
 <Entity id="9995">
  <EntityId>37feecb0-d21e-5087-908b-d3585a2e2eb6</EntityId>
  <Description>STEP DATUM_FEATURE [SAR] DATUM C</Description>
 </Entity>
 <Entity id="9994">
  <EntityId>abeedfe4-90ba-5cfc-b176-3dea41f6fb09</EntityId>
  <Description>STEP ADVANCED_FACE -- PLANE (Geometry)</Description>
 </Entity>
 <Entity id="9993">
  <EntityId>5a6db061-8e96-5521-8404-577d99fb78f8</EntityId>
  <Description>STEP ADVANCED_FACE -- PLANE (Geometry)</Description>
 </Entity>
 <Entity id="9992">
  <EntityId>1940cba9-e52f-5ee2-9b32-d683f2da6101</EntityId>
  <Description>STEP ADVANCED_FACE -- PLANE (Geometry)</Description>
 </Entity>
 <Entity id="9991">
  <EntityId>477ccd2d-7740-5cc5-92f2-e1e2deac6b12</EntityId>
  <Description>STEP DIMENSIONAL_SIZE (Diameter)</Description>
 </Entity>
 <Entity id="9990">
  <EntityId>aa86a375-f69f-5fb5-8a8f-3e540dff4618</EntityId>
  <Description>STEP CYLINDRICAL_SURFACE (Geometry)</Description>
 </Entity>
```

```
        <Entity id="9989">
          <EntityId>a8f9c629-f614-5645-84b7-e81560cedd06</EntityId>
          <Description>STEP GEOMETRIC_TOLERANCE (Position)</Description>
        </Entity>
      </Entities>
    </DigitalModel>
  </DefinitionExternal>
 </Part>
</PartSet>
</Product>
```

### 5.4.2 DATUM / DatumDefinition

The QIF 3.0 DatumDefinition structure shall indirectly reference the STEP DATUM entities[13] and their related geometric entities using two references – one to a FeatureNominal and the FeatureNominal's EntityExternalId and a second one to an OtherNonShapeFeatureNominal and the OtherNonShapeFeatureNominal's EntityExternalId. The two EntityExternalIds point to the DatumDefinition's geometric reference and Datum_Feature reference, respectively, in the external STEP UUID entity list of the QIF Product structure (Section 5.4.1). In the example below, the FeatureNominalIds point to PlaneFeatureNominals and Datum_Features for the three Datums - A, B, and C.

```
<DatumDefinition id="23">
  <DatumLabel>A</DatumLabel>
  <FeatureNominalIds n="2">
    <Id>5</Id>
    <Id>93</Id>
  </FeatureNominalIds>
</DatumDefinition>
<DatumDefinition id="24">
  <DatumLabel>C</DatumLabel>
  <FeatureNominalIds n="2">
    <Id>6</Id>
    <Id>94</Id>
  </FeatureNominalIds>
</DatumDefinition>
<DatumDefinition id="25">
  <DatumLabel>B</DatumLabel>
  <FeatureNominalIds n="2">
```

[13] For details on relationships between STEP DATUM_FEATURE and DATUM, reference the CAx-IF PMI Recommended Practices document or STEP AP242 standard.

```
    <Id>7</Id>
    <Id>95</Id>
   </FeatureNominalIds>
  </DatumDefinition>
```

### 5.4.3 Features

#### 5.4.3.1 Feature - DATUM_FEATURE (Example - Plane)

For a STEP DATUM_FEATURE, QIF shall identify a FeatureNominal of a type corresponding to the STEP DATUM_FEATURE which points to the external UUID entity ID in the QIF Product structure (Section 5.4.1).

Examples of these identifications for three PlaneFeatureNominals are shown below.[14] This PlaneFeatureNominal shall be referenced by a DatumDefinition to complete the QIF dataset for a STEP DATUM_FEATURE.

```
  <PlaneFeatureNominal id="5">
   <FeatureDefinitionId>1</FeatureDefinitionId>
   <EntityExternalIds n="1">
    <Id>9994</Id>
   </EntityExternalIds>
   <Location>-40 -1.4210854715202e-14 10</Location>
   <Normal>0 0 1</Normal>
  </PlaneFeatureNominal>
  <PlaneFeatureNominal id="6">
   <FeatureDefinitionId>2</FeatureDefinitionId>
   <EntityExternalIds n="1">
    <Id>9993</Id>
   </EntityExternalIds>
   <Location>-0 2.77555756156289e-17 0</Location>
   <Normal>0 -1 0</Normal>
  </PlaneFeatureNominal>
  <PlaneFeatureNominal id="7">
   <FeatureDefinitionId>3</FeatureDefinitionId>
   <EntityExternalIds n="1">
    <Id>9992</Id>
   </EntityExternalIds>
```

[14] The planning application shall provide specialized types of FeatureNominal using qualified algorithms but testing of those algorithm implementations is not within the scope of this document.

```
    <Location>0 60 0</Location>
    <Normal>1 0 0</Normal>
  </PlaneFeatureNominal>
```

### 5.4.3.2 Feature – OtherNonShapeFeatureNominal

The OtherNonShapeFeatureNominal shall be used to establish a reference between a DatumDefinition (Section 5.4.2) and the ExternalEntityId that points to the STEP Datum_Feature (Section 5.4.1). Note that an OtherNonShapeFeatureDefinition is also included in the FeatureDefinitions structure as a placeholder to complete the OtherNonShapeFeatureNominal.

```
<OtherNonShapeFeatureDefinition id="92">
   <Description>placeholder</Description>
</OtherNonShapeFeatureDefinition>

<OtherNonShapeFeatureNominal id=93>
    <FeatureDefinitionId>92</FeatureDefinitionId>
    <EntityExternalIds n=”1”>
        <Id>9997</Id>
    </EntityExternalIds>
</OtherNonShapeFeatureNominal>

<OtherNonShapeFeatureNominal id=94>
    <FeatureDefinitionId>92</FeatureDefinitionId>
    <EntityExternalIds n=”1”>
        <Id>9996</Id>
    </EntityExternalIds>
</OtherNonShapeFeatureNominal>

<OtherNonShapeFeatureNominal id=95>
    <FeatureDefinitionId>92</FeatureDefinitionId>
    <EntityExternalIds n=”1”>
        <Id>9995</Id>
    </EntityExternalIds>
</OtherNonShapeFeatureNominal>
```

#### 5.4.3.3 Feature - Cylinder

An example of identification for the Cylinder FeatureNominal is shown below.[15] An EntityExternalId element is added to point to the external UUID entity ID in the QIF Product structure (Section 5.4.1).

```
<CylinderFeatureNominal id="11">
  <FeatureDefinitionId>4</FeatureDefinitionId>
    <EntityExternalIds n=”1”>
        <Id>9990</Id>
    </EntityExternalIds>
    <Axis>
      <AxisPoint>-30 45 0</AxisPoint>
      <Direction>0 0 1</Direction>
    </Axis>
</CylinderFeatureNominal>
```

### 5.4.4 Characteristics

For QIF Characteristics, the STEP identification data shall be captured in the applicable CharacteristicNominal element. An EntityExternalId element is added to point to the external UUID entity ID in the QIF Product structure (Section 5.4.1).

Examples of these identifications for a DiameterCharacteristicNominal and a PositionCharacteristicNominal are shown below.

```
<DiameterCharacteristicNominal id="19">
  <CharacteristicDefinitionId>17</CharacteristicDefinitionId>
  <EntityExternalIds n=”1”>
     <Id>9991</Id>
  </EntityExternalIds>
  <TargetValue>10</TargetValue>
</DiameterCharacteristicNominal>
<PositionCharacteristicNominal id="20">
            <CharacteristicDefinitionId>18</CharacteristicDefinitionId>
     <EntityExternalIds n=”1”>
     <Id>9989</Id>
  </EntityExternalIds>
```

[15] The planning application shall provide specialized types of FeatureNominal using qualified algorithms but testing of those algorithm implementations is not within the scope of this document.

```
    </PositionCharacteristicNominal>
  </CharacteristicNominals>
```

### 5.4.5 Passing STEP UUIDs downstream within QIF

STEP UUIDs that are assigned to QIF CharacteristicNominals shall propagate (i.e., be 'copied') downstream into QIF Results data to continue the traceability thread from "as-designed" characteristics (PMI) to "as-measured" characteristics. The Product (<Part>) structure of Section 5.4.1 may also propagate downstream into QIF Results data to continue the traceability thread[16]. An EntityExternalId element is added to point to the external UUID entity ID in the QIF Product structure (Section 5.4.1).

The following example shows how CharacteristicNominals and their corresponding EntityExternalIds are represented in the context of the QIF Results data -

```
  <CharacteristicNominals n="2">
    <DiameterCharacteristicNominal id="4">
      <CharacteristicDefinitionId>3</CharacteristicDefinitionId>
      <EntityExternalIds n="1">
        <Id>9991</Id>
      </EntityExternalIds>
      <TargetValue>10</TargetValue>
    </DiameterCharacteristicNominal>
    <PositionCharacteristicNominal id="8">
      <CharacteristicDefinitionId>7</CharacteristicDefinitionId>
      <EntityExternalIds n="1">
        <Id>9989</Id>
      </EntityExternalIds>
    </PositionCharacteristicNominal>
  </CharacteristicNominals>
```

***The following paragraph is out of scope for current vendor testing; some experimentation will be done for feasibility purposes –***

Doing so will allow the ability to pass back feedback to upstream users, but the process is realized by combining as-measured results with the original STEP source files (to create new STEP files) using the UUIDs to synchronize "as-measured" characteristic results to "as-designed" PMI. The upstream user will receive a clearly delineated source of truth for the combined data.

[16] Please refer to Section 2 Scope and the example data of Appendix B - TEST8 and its link in B-1.

## 5.5 Demonstration of Traceability between STEP and QIF via UUIDs

A test of traceability between STEP and QIF Plans and QIF Results files was performed using an AI Large Language Model (AI LLM). This AI LLM model was populated with relationship rules formulated for cross-domain traceability analyses. Both forward and backward trace analyses were performed on the example model data (see Figure 3 below) and that analysis verifies that the supplemental structures shown in this Recommended Practice document confirm cross-domain traceability – including traceability between Characteristics in QIF results, Characteristics, Features, and DatumDefinitions in QIF Plans, and equivalent STEP entities (refer to Appendix B).

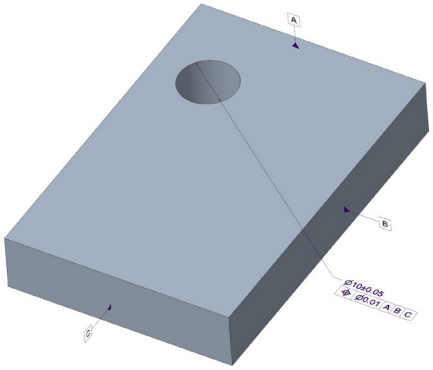


*Figure 3 - Example Test Model*

## 5.6 Preprocessor and Postprocessor Recommendations for Downstream Uses

**Preprocessor Recommendations:**

All STEP preprocessors shall generate UUIDs for each instance of each entity data type that they wish to permanently identify and shall ensure that all UUIDs assigned to entities from the CAD model are as described in Footnote 4 (page 8).

**Postprocessor Recommendations:**

QIF postprocessors will begin by using QIF 3.0 syntax and structures unless otherwise noted. Support for QIF 4.0 structures will be added to this document and be tested once QIF 4.0 has been released.

All QIF postprocessors, upon first import of a STEP model, shall retain incoming UUIDs for all instances of identified STEP entities and map UUIDs assigned to the equivalent QIF elements as described in this document. This document does not provide queries for detailed traversal of STEP data model to the final target ENTITY instance and attribute. For example, STEP UUIDs are provided for the ADVANCED_FACE or EDGE_CURVE and the QIF postprocessors shall follow the topology links from ADVANCED_FACE or EDGE_CURVE to underlying[17] geometry elements needed for FeatureNominals in QIF.

All postprocessors, upon the first import of the STEP model, shall retain in the mapping the product UUID and STEP file name information for the model as described above.

Subsequent to postprocessing, the imported model data may be used for further metrology planning purposes including the addition of new metrology specific entities (geometry,

[17] Specifically, the attribute: `face_geometry inherited from face_surface.`

features, datums, PMI, UDAs, process information, etc) ***as long as none of the imported entities that had UUIDs assigned are modified in any way.***

The postprocessing system shall, if new content is added, create and assign UUIDs to new entities as needed per QIF schema rules to further facilitate exchange using the method described above for preprocessors.

# Appendix A - Availability of Implementation Schemas

## *A.1 STEP AP242*

The AIM / MIM Longform EXPRESS schemas supporting the concepts described in this document can be downloaded from the MBx-IF homepage under **MBx > Resources > EXPRESS Schemas** or using the direct link:

https://www.mbx-if.org/home/mbx/resources/express-schemas/

The page lists the schemas for:

1. AP214 Edition 3 (2010)
2. AP203 Edition 2 (2011)
3. AP242 Edition 1 (2014)
4. AP242 Edition 2 (2019)
5. AP242 Edition 3 (2022)
6. AP242 Edition 4 (2025)

**Note:**

- The approaches described in this document require at least AP242 Ed.4.

## *A.2 Quality Information Framework (QIF) 3.0*

All three sets of normative documents that make up the QIF 3.0 standard (text, XML Schema files, and XSLT constraint checking files) are available for download free of charge at:

1. https://qifstandards.org/download

An on-line browser for the entire QIF 3.0 XML Schema model is at:

2. https://qualityinformationframework.github.io/qif3-browser/qif3.html

QIF 3.0 XML Schema files that describe exactly the same model but have corrections to the key and keyref constraints are available at the following site along with the originals: here

# Appendix B – Example Data and Traceability Analyses

Example data for the initial feasibility testing of cross-domain exchange between STEP AP242 Edition 4 and QIF 3.0 using two approaches –

- TEST7 – duplicated STEP mapping table and duplicated FeatureNominals and CharacteristicNominals in Results and
- TEST8 – abbreviated Results that eliminate duplication and use external QIF IDs [xId]

1. **PID_QIF_STEP_TEST7.txt**
    a. Summary description of data package for TEST7 test.
2. pdi_0-e5-242.stp
    a. Source STEP file from CAx-IF R55J, published from Elysium (CATIA V5 source), with v5 UUIDs.
3. pdi_0-e5-242_STEP_UUIDs_TEST7.qif
    a. QIF Plan file, published from a Metrology Planning system after ingesting published STEP file;
    b. includes STEP v5 UUIDs defined as ExternalIDs in a <Product><Part><DefinitionExternal> table and
    c. attached to <DatumDefinition>s, <FeatureNominal>s, and CharacteristicNominal>s as appropriate,
4. pdi_0-e5-242_Results_TEST7tk.qif
    a. QIF Results file, published from a Metrology Execution/Analysis system;
    b. includes STEP v5 UUIDs defined as ExternalIDs in a <Product><Part><DefinitionExternal> table and
    c. attached to <DiameterCharacteristicNominal> and <PositionCharacteristicNominal > for <DiameterCharacteristicDefinition> and <PositionCharacteristicDefinition>, respectively.

Example traceability results for the above example data includes the following –

5. trace_fwd_TEST7_v7a_2025-12-16.html
    - AI LLM analysis of the above example data for forward traceability
    - Outcome – Successful trace from STEP source data through QIF Plan to QIF Results for Product, Datum, Geometry (Features), and PMI (Characteristics) was achieved without errors.
6. trace_back_TEST7_2025-12-16.html
    - AI LLM analysis of the above example data for backward traceability
    - Outcome – Successful trace from QIF Results through QIF Plan to STEP source data for Characteristics (STEP PMI), Features (Geometry), Datum, and Product was achieved without errors. Failed Measurement in QIF Results were also successfully traced back to STEP PMI.

7. **PID_QIF_STEP_TEST8.txt**
    a. Summary description of data package for TEST8 test.
8. pdi_0-e5-242.stp
    a. Source STEP file from CAx-IF R55J, published from Elysium (CATIA V5 source), with v5 UUIDs.
9. pdi_0-e5-242_STEP_UUIDs_TEST8.qif (the same data content as in TEST7)
    a. QIF Plan file, published from a Metrology Planning system after ingesting published STEP file.

   b. includes STEP v5 UUIDs defined as ExternalIDs in a <Product><Part><DefinitionExternal> table and
   c. attached to <DatumDefinition>s, <FeatureNominal>s, and CharacteristicNominal>s as appropriate.

10. pdi_0-e5-242_Results_TEST8tk.qif
    a. abbreviated QIF Results file, published from a Metrology Execution/Analysis system.
    b. removes STEP v5 UUIDs defined as ExternalIDs in a <Product><Part><DefinitionExternal> mapping table from TEST7 and
    c. instead uses only xIDs on <CharacteristicItems> attached to <CharacteristicMeasurement>s for each of three <MeasurementResultsSet>s.

Example traceability results for the above example data includes the following –

11. back_trace_TEST7_v8_20251216.html
    - AI LLM analysis of the above example data for back traceability
    - Outcome – Successful trace from abbreviated QIF Results through QIF Plan to STEP source data for Characteristics (STEP PMI), Features (Geometry), Datum, and Product was achieved without errors. Failed Measurement in QIF Results were also successfully traced back to STEP PMI.

The above data and analyses can be retrieved from the links below -

### *B.1 Example Data and Traceability Analyses (TEST7, items #1 through #6 above)*

Link to TEST7 example data zip file in PDES Teams PID Project directory

Link to TEST7 example data zip file in MBx IF/CAx-IF/Draft Recommended Practices/Rec Prac for Cross Domain Exchange directory

### *B.2 Example Data and Traceability Analyses (TEST8, items #7 through #11 above)*

Link to TEST8 example data zip file in PDES Teams PID Project directory

Link to TEST8 example data zip file in MBx IF/CAx-IF/Draft Recommended Practices/Rec Prac for Cross Domain Exchange directory

### *B.3 STEP Source file augmented with mapped QIF Results data as Feedback*

This section is for future functionality.

# Appendix C – JIRA USER STORY (Jira Issue #TCSC10303-14)

Note that the Jira repository is a closed database of STEP-related issues under the control of ISO and access is limited to ISO Technical Committee members.

Content of user story repeated here:

As a user, I want to identify product data by an identifier that comply with RFC 4122 and ISO/IEC 9834-8:2014 requirements.

As a design engineer, I want to share design CAD data with another user without re-creating the entire CAD model tree.

---The need is to re-identify the CAD elements when data is received. The receiving system will accurately determine data that has not been modified.

---

---CAD maintained design data is assumed to change only identifiers for data that has been modified.

Note: 2023-09-15: The above assumption is invalid. CAD internal objects persist until they are deleted; the consequence is that the value of the version 5 UUID for a CAD object persists until the CAD internal object is deleted.

```
Example: A design system chooses to use the CAD internal object id to populate the
'name' variable for the generation of a UUID version 5 (where the 'namespace' vari-
able is fixed).
A CAD model has a hole in it.
The hole is represented as an internal CAD object.
On the first save of the model the hole is assigned a UUID of 'xx1234yy'.
On the second save of the model the hole has been moved.
The hole UUID value for the second save of the model remains 'xx1234yy'.
```

---

As an implementor, I want the classification of version 4 or version 5 to be explicitly declared in the data set.

As a user of ISO 29002, I wish to share hints on identifiers with others who understand the hint descriptions that I use.

As a user of a digital twin environment, I need my software applications to support identifiers.

As a user of a digital twin environment, I need to follow relationships between identifiers and exchange and share the provenance of product data in the digital twin environment based on a list of those relationships.

As a user of measurement equipment, I need to identify specific geometrically located position on the surface of a product where an anomalies is found.

As a user, I want to identify individual product data elements, random collections of data, or data where the identifiers are related in a tree structure.

As a user, I want to exchange or share a hash function that can be used to as a seed for a version 5 identifier where the hash is over a specified subset of a product data set.

As a partner, supplier, or manufacturing engineer, I want my applications software to be able to automatically consume the identifiers.

As a user, I recognize that the namespace and name values used as input to the version 5 algorithm are not exchanged and therefore must be maintained by the user and implementor organizations as appropriate.

As a user I want to optimize the cost of comparing two data sets.

---