Evaluation of a standard model designed to Support Traceability of Product Requirements in a Digital Twin through interoperability testing.

Thomas R. Thurman**[[1]](#footnote-1)**, Asa G. Trainer**[[2]](#footnote-2)**, Rosemary L. Astheimer**[[3]](#footnote-3)**, and Allison Barnard Feeney**[[4]](#footnote-4)**.

ABSTRACT

In the model-based enterprise (MBE) paradigm, enterprises are fueled by the digital thread, an authoritative, integrated information flow that connects all product life cycle phases. This paper presents initial evaluation results in interoperability testing of an enhancement to the ISO 10303-242 standard that includes explicit support for associating a universally unique identifier (UUID) to an individual requirement in support of the digital thread.

Keywords

computer-aided design (CAD), computer-aided inspection (CAI), computer-aided manufacturing (CAM), Digital Thread, engineering design, ISO 10303, ISO 23247, product model data standard, persistent identifier, product and manufacturing information (PMI), Quality Information Framework (QIF), STandard for the Exchange of Product data (STEP), universally unique identifier (UUID).

# Introduction

In the [model-based enterprise](#_bookmark349) ([MBE](#_bookmark349)) paradigm, enterprises are fueled by the digital thread, an authoritative, integrated information flow that connects all product life cycle phases. Standard data representations play a crucial role in facilitating the vision of [MBE](#_bookmark349) due to the role of standards in facilitating interoperability, lowering the cost of solutions, ensuring repeatability of methods, and protecting data assets from obsolescence. Neither widely used product data standards nor commercial engineering software adequately supports universally unique identifiers ([UUIDs](#_bookmark366)) throughout the digital thread, providing a significant roadblock to realizing the model-based enterprise.

The scope of this research is the evaluation of a standard model in interoperability testing. The product data standard evaluated is ISO 10303-242 (AP 242 Managed model-based 3D engineering4).

* This paper discusses evaluation of interoperability testing of UUIDs for a CAD object and aggregated CAD objects that compose a single domain concept in product data standards. UUIDs are included in the STEP product model to address findability issues9,10 in multi-domain and multi-life-cycle engineering and manufacturing contexts.

## Product Data Standards

This section introduces the product data standard considered in this research.

### ISO 10303 (STEP) Application Protocol 242

ISO 10303-242:20224, titled “Managed model-based 3D engineering,” commonly known as AP 242, is a broadly used product information standard that has become increasingly capable of conveying manufacturing requirements to downstream systems12.

The goal of [AP 242](#_bookmark322) is to support a manufacturing enterprise with a range of standardized information models that flow through a long and wide “digital thread” that make manufacturing systems in the enterprise smart13. The information models support classification and identification schemes, data validation, and long-term data retention requirements for industries such as aerospace and defense14. Digital data conformance to [AP 242](#_bookmark322) plays a central role in achieving the goal of successful long-term data retention.

A study16 comparing drawing-based processes with model-based processes concluded that [PMI](#_bookmark358) has the potential to make life cycle processes run faster, with fewer errors, at lower cost. [AP 242](#_bookmark322) offers standards-based models that include the representation of semantically rich PMI that is computer interpretable13. This breakthrough supports manufacturing’s need for model-based [CAM](#_bookmark328) and [coordinate measurement system](#_bookmark333) ([CMS](#_bookmark333)) processes. [AP 242](#_bookmark322) increases the effectiveness of [MBE](#_bookmark349) by enabling a common path for [MBD](#_bookmark347) and model-based manufacturing ([MBM](#_bookmark350)) integration15,17.

The above capabilities make AP 242 the de facto mechanical engineering design and product information data standard. However, the ability to trace data correlated to product characteristics through the life cycle and back to the originating system is lost when downstream applications do not support the as-planned and as-measured capabilities in [AP 242](#_bookmark322). One goal of extending AP 242 with support for UUIDs is to accelerate adoption of support for the as-planned and as-measured capabilities in [AP 242](#_bookmark322).

# Materials and Methods

This section describes the research methodology, data collection through industrial use cases, and analytical methods used to develop and validate the UUID implementation approach.

## Research Methodology

This section discusses the research approach to identify and analyze processes to utilize UUIDs and propose an implementation.

### Establish Reference Model

#### Derived Requirements

This section captures common requirements for identifiers in the context of mechanical design and manufacturing model-based enterprises. Requirements are derived from identified use cases, those found in literature and patents, and specified in standards.

##### Product Identification Requirements

Identifiers for the enterprise (the product design owner), product model class, product part number, and product serial number are typical data elements managed at the manufacturer and customer level.

Consideration of established widely used standards for product marking was outside the scope of this research.

##### Feature Classification and Identification Requirements

A rich body of feature classification is available in [STEP](#_bookmark363) for electrical design-oriented features, mechanical manufacturing process-related features, and pre-defined complex hole features such as counterbored holes.

Example 2

ISO 545922 specifies datum targets and their string value classification. There is a datum in a design with a designator of “C.” There is a datum target feature balloon21 in that design with a designation of “1”. That designation indicates the first datum target in support of datum C. The string “C1” uniquely identifies that target to a human operator. The datum target is associated in the [CAD](#_bookmark326) system to a [UUID](#_bookmark366). Because the [UUID](#_bookmark366) is bound to a [CAD](#_bookmark326) object and not the string “C1”, an extra processing application is required to address the human-computer associativity gap by using the tuple (type name = datum\_target, value = ”C1”) as the name value for the version 5 algorithm.

#### UUID Technical Specifications

This section discusses [UUID](#_bookmark366) encoding requirements and characteristics, types of [UUIDs](#_bookmark366), how they are used with human-readable identifiers, and what requirements they address. It also introduces a requirement that the [UUID](#_bookmark366) application in [STEP](#_bookmark363) applies digital signatures to deter tampering with [UUID](#_bookmark366) instances.

##### ISO/IEC 9834-8 Generation of UUIDs and Their Use in Object Identifiers

ISO/IEC 9834-8:201428, Generation of universally unique identifiers (UUIDs) and their use in object identifiers specifies procedures for [UUID](#_bookmark366) generation and use in the international object identifier tree under the joint UUID arc.

##### IETF RFC 9562 Universally Unique IDentifiers (UUIDs)

RFC 956229clause 6.5 provides an algorithm to create a name-based version 5 [UUID](#_bookmark366). There are two versions of [UUIDs](#_bookmark366) considered: random-number-based and name-based. Both versions meet the functional requirement of uniqueness. ISO/IEC 9834-8:201428 and RFC 956229 are deemed equivalent for this research.

##### UUID Versions and Critical Characteristics

This section briefly describes each UUID version selected for inclusion in the research.

* Version 4 produces a random value each time the algorithm is executed. Version 4 is used in commercially available product data exchange interface implementations.
* Version 5 uses a namespace and name. A compliant implementation produces the same result each time the same name in the same namespace is transformed. The namespace identifier itself is a UUID, and any desired [UUID](#_bookmark366) may be used as a namespace designator. Either [MD5](#_bookmark351) or [SHA-1](#_bookmark362)30 may be used, but backward compatibility is not an issue, although [SHA-1](#_bookmark362) is preferred.

###### Namespace Attribute

The namespace attribute must be a [UUID](#_bookmark366). This document does not recommend including the namespace in the exchange data set; the type classification of the uuid\_attribute identifies whether the uuid string is a version 4 or version 5 [UUID](#_bookmark366). Declaring that the [UUID](#_bookmark366) is version 5 does permit an importing application to determine that the importing organization does not own the imported UUID.

###### Name Attribute

The name attribute required for version 5 can be any unique string determined by the authoring CAD system in the context of the source [CAD](#_bookmark326) model. This document does not support including the name in the exchange data set.

###### Leveraging the Scope of an Object to Improve Efficiency

The scope of an object may be used to improve the efficiency of comparison by reducing the number of queries necessary to detect a change. [STEP](#_bookmark363) uses an object existence dependent model (i.e., one instance is explicitly dependent on another as specified by an attribute or constraints in the ENTITY declaration)33. In such cases, a query can be performed to identify each object that is directly or indirectly dependent on the object selected as the root of the query. [STEP](#_bookmark363) also supports establishing existence dependency using what is known as directed relationship entities. The directed relationship entities relate two instances where the existence of the instance on the related end of the relationship is dependent on the instance on the relating end of the relationship. There are a few cases in [STEP](#_bookmark363) where the relationship is not directed. In those cases, a recommended practice should be created to provide implementation guidance.

#### Proposed Model Framework

This section provides an overview of the proposed information model to address the previously described requirements for [UUIDs](#_bookmark366) in [STEP](#_bookmark363).

The proposed model includes the following high-level benefits:

* The name-based [UUID](#_bookmark366) version meets the functional requirement of the engineering domain requirement for repeatability;
* The random-based [UUID](#_bookmark366) version provides flexibility in implementations that require it;
* Including relationship ENTITY data types specific to [UUID](#_bookmark366) support traceability and associativity;
* Including the proposed models directly in the [STEP](#_bookmark363) information model provides that the [STEP](#_bookmark363) data set is not limited to a specific data representation format;
* The type hierarchy of the proposed model provides for future types to be added in an upwardly compatible manner; and
* Actors that assert ownership of elements in the data set can be distinguished as each actor has a unique namespace.

The remainder of this section describes:

* recommendations proposed in this research for implementing [UUIDs](#_bookmark366) in different contexts;
* recommendations for [UUID](#_bookmark366)encoding structure;
* recommendations for Merkle tree structure; and
* the purpose of key entities and types in the proposed model.

##### Recommendations for Specific Contexts

This subsection provides recommendations for UUID use in the context of [STEP](#_bookmark363) design [APs](#_bookmark316).

###### Recommendations for UUIDs in STEP Design APs

The recommendation for [STEP](#_bookmark363) [APs](#_bookmark316) that include engineering design in their scope (so-called design APs) is to use the name-based [UUID](#_bookmark366) version. [AP 238](#_bookmark320) includes [UUIDs](#_bookmark366) in data set anchor sections conforming to the ISO 10303-21:201634 anchor section approach described in AMS 300-1235. This is identified as version 5 in ISO/IEC 9834-827 and in Request for Comment 956229. Recommended UUID Encoding Structure

The Version 5 [UUID](#_bookmark366) uses two arguments: domain key and name string. A [UUID](#_bookmark366) value is dependent on the combination of the domain key and name string. Neither of those arguments is included in the exchange data set. The originating enterprise is responsible for maintaining a registry of domain keys and name strings. The domain key is typically maintained by the enterprise for its sub-domains36,37.

The name string is proposed to be the internal [CAD](#_bookmark326) [OID](#_bookmark355) or an unambiguous path to the internal [CAD](#_bookmark326) object from the internal root object in the software application model. The [UUID](#_bookmark366) proposed herein is a 36-character string that supports the requirements of version 5 encoding28,2,34. It also supports the version 4 encoding specified in those recommendations and standards.

.

How the Proposed Model Addresses Requirements

###### Model Applied to Shared Design Collaboration Use Case

This section describes how the proposed model addresses the requirements described in the use case. This use case employs [STEP](#_bookmark363) change management entities to support an iteration process with the uuid\_relationship entity specifying the product definition state transition based on changes, additions, or deletions of product data instances with a [UUID](#_bookmark366) attached. An exchange agreement would be needed to identify the class of coordination involved in the collaboration.

In this example, a [UUID](#_bookmark366) is applied to a tolerance where the tolerance class is changed.

Example 7

The initial iteration of a design (Revision ‘-’) of a part has a hole number 33 with a cylindricity\_tolerance instance value of 0.01 mm. In the design pre-processor, the cylindricity\_tolerance instance is given a [UUID](#_bookmark366) of ’6ba7b810-9dad-11d1-80b4-00c000000000’. The following shows a portion of a STEP that captures the UUID and its relationship to the hole and the related cylindricity tolerance.

#1=PRODUCT\_DEFINITION\_FORMATION('-',..,..);

#2=SHAPE\_ASPECT('hole\_33',,);

#10=CYLINDRICITY\_TOLERANCE(#2,..,..,0.01MM);

#11=UUID\_ATTRIBUTE(#10, '6ba7b810-9dad-11d1-80b4-00c000000000');

In Revision A, the hole cylindricity\_tolerance instance value is changed to a position\_tolerance instance with a value of 0.005 mm. In the design preprocessor, the position\_tolerance instance is given a [UUID](#_bookmark366) of ’7ba7b810-9dad-11d1-80b4-00c000000000’. The [CAD](#_bookmark326) source model records the [CAD](#_bookmark326) [OID](#_bookmark355) entry for cylindricity\_tolerance for hole 33 and is marked as ’deleted’ (omitted from the example). The following shows a portion of a STEP file, including changes for Revision A.

#16=PRODUCT\_DEFINITION\_FORMATION('A',..,..);

#22=SHAPE\_ASPECT('hole\_33',,);

#100=POSITION\_TOLERANCE(#22,..,..,0.005MM);

#110=UUID\_ATTRIBUTE(#100, '7ba7b810-9dad-11d1-80b4-00c000000000');

Furthermore, in the same Revision A data set, the uuid\_relationship instance would be populated with:

#1=UUID\_RELATIONSHIP(

'00000000-9dad-11d1-80b4-00c000000000',

'6ba7b810-9dad-11d1-80b4-00c000000000',

'7ba7b810-9dad-11d1-80b4-00c000000000',

'supersedes',$);

The user organization’s quality process establishes traceability requirements for characteristics. User and implementation forums are encouraged to define exchange agreements for appropriate application of the capabilities of the [UUID](#_bookmark366) ENTITY data types.

Example 4

In this case, each [OEM](#_bookmark354) uses the same application and configuration of that application. However, the namespace [UUID](#_bookmark366) reflects the context of the [OEM](#_bookmark354). The critical constraint is that each application instance protects the integrity of the internal [CAD](#_bookmark326) [OID](#_bookmark355)s as the data evolves. Recall that the benefit of the version 5 [UUID](#_bookmark366) is that the retained [OID](#_bookmark355)s generate the same [UUID](#_bookmark366) upon export. New [UUIDs](#_bookmark366) are only created when new [OIDs](#_bookmark355) are created. Recall that the [OID](#_bookmark355) generation process cannot depend on the iteration or revision identifier of the [CAD](#_bookmark326) file being generated. A highly interactive collaboration would bypass the [OEM](#_bookmark354) [PLM](#_bookmark357) system, enabling direct [CAD](#_bookmark326) to [CAD](#_bookmark326) file transmission. In this example, both [OEMs](#_bookmark354) are modifying topology. A hole is added by [OEM](#_bookmark354) Two, but in a different area of the design, so a new uuid\_attribute would be provided for the additional hole.

#### Guidelines and Rules

This section describes recommendations for application behavior when implementing the proposed model.

##### Creating Application

It is recommended that all [CAD](#_bookmark326) applications that claim conformance to [STEP](#_bookmark363) generate UUIDs and that the same UUID be regenerated for unchanged [CAD](#_bookmark326) objects. The internal identifier for the [CAD](#_bookmark326) object is mapped to the [UUID](#_bookmark366). Property changes to the [CAD](#_bookmark326) object other than its internal identifier are ignored when calculating the [UUID](#_bookmark366).

##### Importing Application

It is recommended that all [CAD](#_bookmark326) applications that claim conformance to [STEP](#_bookmark363) shall import [UUID](#_bookmark366)s and maintain their relationship to the product data established in the data set received. A mapping table is recommended to persistently store the external [UUID](#_bookmark366) relation to the internal [CAD](#_bookmark326) [OID](#_bookmark355).

It is recommended that all downstream applications import [UUID](#_bookmark366)s and maintain their relationship to the product data established in the initial data set received. Downstream applications that generate data derived from the imported data must specify the relationship between the exported data and the imported [UUID](#_bookmark366). In a digital twin application, the term downstream implies the receiving system.

##### UUID Data Elements and Rules

The following sections describe the management of UUID data.

###### UUID Namespace Management

Each application that creates [STEP](#_bookmark363) data sets must create a namespace [UUID](#_bookmark366) for its application in the context of the using organization. The details of that transaction are outside the scope of this document. The only requirement proposed for namespace in this document is that the namespace be registered internally in an enterprise. There is no mechanism proposed in this document for exchanging namespaces.

###### UUID Name Management

Each application that creates [STEP](#_bookmark363) product data sets must map internal data structures to a name used to create a [UUID](#_bookmark366). For simplicity of reference, this document refers to a [CAD](#_bookmark326) [OID](#_bookmark355) as the internal representation of the name used by the [UUID](#_bookmark366) version 5 algorithm. Cases exist where [OIDs](#_bookmark355) do not exist; the preprocessor is expected to use the full path from root to object as the internal representation of the name used by the [UUID](#_bookmark366) version 5 algorithm. The only requirement recommended by this document is that the assignment method be repeatable.

Example 5

This example provides a [UUID](#_bookmark366) as input to the version 5 algorithm as a complete illustration. The enterprise provides an enterprise level UUID as a seed ’namespace’ similar to the default ’namespace’29. In this example, the data set filename and [CAD](#_bookmark326) system identifier are concatenated to provide input to the version 5 algorithm to create a [UUID](#_bookmark366). That [UUID](#_bookmark366) is then considered to be the ’namespace’ value for input to the version 5 algorithm that calculates the [UUID](#_bookmark366) for the [CAD](#_bookmark326) object. That ’namespace’ [UUID](#_bookmark366) for the data set filename and [CAD](#_bookmark326) system identifier for the example is e00108f0-b388-5999-bdcc-033c5e0b2203. This example assigns the [CAD](#_bookmark326) object ID value to the ’name’ input of the version 5 algorithm to calculate the [UUID](#_bookmark366) value for the single [CAD](#_bookmark326) object use case.

To generate the [UUID](#_bookmark366), newlines were removed, and each space was converted to a dash. In the data illustrated below, white space was added for clarity.

Internal enterprise data:

Filename: "the CAD FILENAME”

CAD System: "FreeCad version 2.3.4”

Internal CAD model data:

Type: "datum”

CAD object ID: "12345”

value: "A"

Resulting STEP data:

#33=DATUM('','',$,.F.,'A');

#44=V5\_UUID\_ATTRIBUTE(0d8c2a8f-0bcd-59c6-8b39-20aa4e958adf, (#33));

Example 6

This example extends the previous example by including a datum\_target and shape\_aspect\_relationship to form an ordered collection that consists of the datum in the previous example.

Internal CAD model data:

Type: "datum\_target”

CAD object ID: "4567”

value: "A1"

target: "12345"

Resulting STEP data:

#33=DATUM('','',#1,.F.,'A');

#35=DATUM\_TARGET('','',#1,.T.,'A1');

#36=SHAPE\_ASPECT\_RELATIONSHIP('','',#33, 35);

#44=V5\_UUID\_ATTRIBUTE(d6295c16-e110-5a99-994116baabf28480(#35,#36,#33));

###### STEP Object Management

This proposed model allows assigning a [UUID](#_bookmark366) to a single [STEP](#_bookmark363) object or to a collection of [STEP](#_bookmark363) objects. In some cases, a single [CAD](#_bookmark326) object may map to a collection of [STEP](#_bookmark363) objects. Exchange agreements or application protocols will be required to establish object ownership. Examples include:

* data set ownership where the data set can be identified and controlled as a single thing, e.g., a zip file;
* ownership of the product for the data set;
* ownership of the product version for the data set;
* ownership of the product definition for the data set;
* ownership of a feature for the data set;
* ownership of a tolerance for the data set; and
* ownership of a feature and all the properties associated with that feature for the data set.

Each organization should assume that if they don’t own the object, they shall not modify the object. They should instead propose changes to the object.

###### UUID Reuse Prohibited

A [UUID](#_bookmark366) may represent the complete path to the [CAD](#_bookmark326) object from the design root, with a unique key assigned for that path stored in the internal CAD OID. When a CAD object is deleted, that path is deleted in the authoring software, and the internal mapping table entry for the UUID is noted as deleted to prevent reuse. Similar behavior is expected when the UUID represents a specific CAD OID independent of the path.

###### UUID Management in the Context of Data Set Revisions

We propose that a preprocessor that processes a design revision regenerate UUIDs identical to those in the previous design, where the revision does not impact the existence of the CAD object.

Example 7

A [datum\_system](#_bookmark384) indirectly references a situation feature22, which is a plane. A design change causes the plane to change to an axis. The [datum\_system](#_bookmark384) [CAD](#_bookmark326) object is deleted and replaced with a new instance in the authoring system because of the change to an axis. Therefore, the [UUID](#_bookmark366) associated with that instance of [datum\_system](#_bookmark384) would not be reused when the revised design was saved and an updated [STEP](#_bookmark363) data set created. If the provenance traceability for [datum\_system](#_bookmark384) is a critical characteristic in the enterprise, a population of the uuid\_relationship with ’supersedes’ could be provided to indicate the specific [UUID](#_bookmark366) that was the replacement. Alternatively, the design change management functionality in [AP 242](#_bookmark322) could be employed to indicate more complete revision information.

###### Internal Pre-Processor Export Rules

These rules are in addition to those inherited from the [UUID](#_bookmark366) management in the context of data set revisions above. An individual annotation cannot be merged or split during translation to preserve the consistency of the associated [UUID](#_bookmark366). Any individual geometry instance that is tagged with a [UUID](#_bookmark366) must be preserved during translation. The application must export any preserved [UUID](#_bookmark366) in the model. The application must export any created [UUID](#_bookmark366) in the model. The decision of what [CAD](#_bookmark326) objects to assign [UUIDs](#_bookmark366) to is implementation dependent. We recommend that exchange agreements be executed to formalize the selection of [CAD](#_bookmark326) objects.

###### External Pre-Processor Export Rules

These rules are in addition to those inherited from the [UUID](#_bookmark366) management in the context of data set revisions above. The application must export any preserved [UUID](#_bookmark366) in the native [CAD](#_bookmark326) model and any [UUID](#_bookmark366) assignment of [CAD](#_bookmark326) objects specified for the translator configuration. Applications may use a configuration file to specify the types for which [UUIDs](#_bookmark366) shall be created.

###### Post-Processor Import Rules

The application may create a [UUID](#_bookmark366) if it does not exist on import, but it must preserve incoming [UUID](#_bookmark366)s.

###### STEP Data Set State Change by an Application

The application that changes the state of a [STEP](#_bookmark363) data set by saving the data respects the [UUID](#_bookmark366) population from the transmitting organization, decides what new objects need [UUIDs](#_bookmark366), and puts [UUIDs](#_bookmark366) only on those objects when it generates an updated [STEP](#_bookmark363) data set. To identify the specific values that changed, the change management schema in ISO/TS 10303-1824 Application module: Change management39 should be applied to communicate the detailed changes.

###### STEP Data Set State Change Requests

Applications that request changes to a [STEP](#_bookmark363) data set shall support design change management as specified in [AP 242](#_bookmark322). To generate the correct change request records, it may be necessary to preprocess the measurement results from manufacturing and manufacturing process planning.

# Data Collection through Industrial Use Case

The following use case is one of several identified by industrial stakeholders that were used as a source of requirements for [UUIDs](#_bookmark366). An extensive set of use cases is available in published NIST AMS 300-1235.

## Shared Design Collaboration

Two design partners iterate on a shared design. A shared design is interpreted such that at each stage of the workflow, the design data is synchronized so that it is irrelevant which partner generated the data for external use. Several of the [CAD](#_bookmark326) objects are owned by the first design partner, and other [CAD](#_bookmark326) objects are owned by the second design partner. A use case42 discussing CAD/CAE integration provides more detailed information.

### Example 9

[OEM](#_bookmark354) One creates an annotation referencing one cylindrical surface, controlled by a single topological face, that represents a hole in the native model. In post-processing by [OEM](#_bookmark354) Two, that surface is split into two surfaces and two topological faces in the derivative model. For collaboration to be successful, [OEM](#_bookmark354) Two retains knowledge that the two [topological face](#_bookmark374)s in [OEM](#_bookmark354) Two’s model are controlled by the single [topological face](#_bookmark374) assigned a [UUID](#_bookmark366) by [OEM](#_bookmark354) One’s [CAD](#_bookmark326) system, and that annotations apply equally to the combination of the two topological [face](#_bookmark374)s in [OEM](#_bookmark354) Two’s model and to the single face in [OEM](#_bookmark354) One’s model.

### Example 10

An [OEM](#_bookmark354) collaborates with a Supplier on the mechanical design of an equipment container, as illustrated in figure [*2*](#_bookmark32). The product illustrated includes:

* a cover;
* a container; and
* a component chassis.

FIGURE 2

An assembly of components.

A blue and yellow rectangular object

Description automatically generated

The design is unreleased, indicated by a ”0” in the major digit of the revision code. The [OEM](#_bookmark354)and the supplier use a minor digit on the revision identifier to signify an iteration during the design phase.

The Supplier is responsible for designing and delivering the component chassis mounted within the container, as shown. Table *1* illustrates the initial condition of the shared model elements considered.

TABLE 1

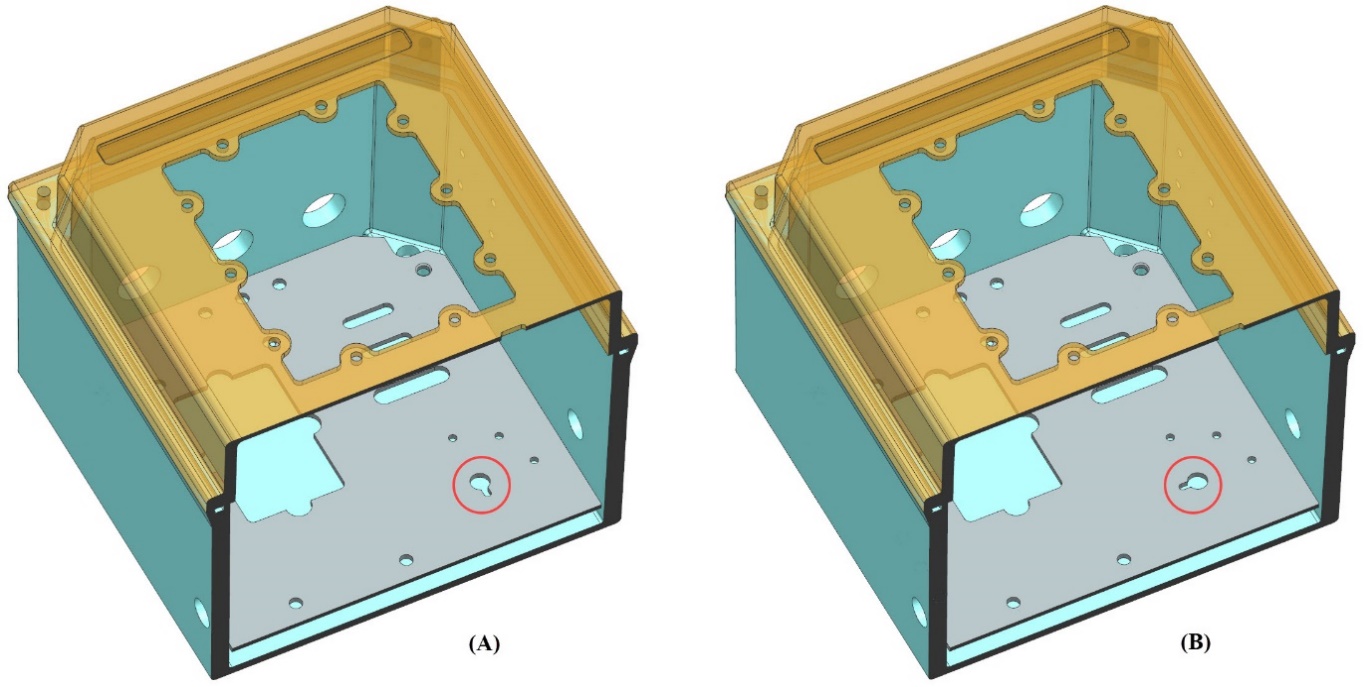
Initial condition of the model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Object Owner | Iteration id | Item | UUID | Data |
| Supplier | 0.0 | Chassis keyway | 1 | Original orientation |
| OEM | 0.0 | Interface hole | 2 | Original interface hole data |

During an iteration of the chassis design, the Supplier analyzes component placements and determines that one component must be reoriented to improve cooling efficiency. To support the component reorientation, the mounting keyway on the chassis is rotated 90°. Figure *3* shows (*A*) the original orientation of the keyway and (*B*) the new position of the keyway. The change requires adjustment to the component’s wiring (not shown) to accommodate the orientation change. The Supplier returns an iteration of the component chassis design and an iteration of the container with an annotated request that the cable interface hole (identified by its [OEM](#_bookmark354)-owned [UUID](#_bookmark366)) in the container wall be moved to the right of the keyway.

FIGURE 3

Keyhole in an assembly. (*A*) close-up of the original keyhole orientation, (*B*) A close-up of the modified keyhole orientation.



The design change report of the keyway change by the Supplier is also identified by its [UUID](#_bookmark366), owned by the Supplier, to make clear to the [OEM](#_bookmark354) what change has been made in the chassis. The details of the change report could include a report identifying the following:

* Supplier-owned (reoriented chassis keyway) data orientation values identified by [UUID](#_bookmark366)(1) supersede Supplier-owned (original chassis keyway) data values identified by [UUID](#_bookmark366)(1), as illustrated in table [*2*](#_bookmark30); and
* [OEM](#_bookmark354)-owned data values identified by [UUID](#_bookmark366)(2) (moved interface hole in the container) supersede [OEM](#_bookmark354)-owned values identified by [UUID](#_bookmark366)(2) (original interface hole), as illustrated in table [*3*](#_bookmark31).

TABLE 2

Supplier-owned data values for the first iteration.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Object Owner | Iteration id | Item | UUID | Data |
| Supplier | 0.1 | Chassis keyway | 1 | Changed orientation |

TABLE 3

Change request report data from the Supplier.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Originator | Object owner | Iteration id | item | UUID | Data value |
| Supplier | OEM | 0.1 | Interface hole | 2 | Technical Details |

Upon receiving the change report, the [OEM](#_bookmark354) will update the chassis design to the latest version supplied and evaluate the request to change the location of the interface hole in the container. Acceptance after evaluation would result in an update to the location of the interface hole in the [OEM](#_bookmark354) design model and transmittal of the updated design to the Supplier. An evaluation resulting in rejection would initiate another round of collaboration.

A cylinder is a combined topological and geometric representation of a hole in a three-dimensional [CAD](#_bookmark326) model that is defined by four attributes: an x,y,z coordinate tuple for location; a line that represents the axis that is the center of symmetry of that cylinder; the radius of that cylinder; and a direction of the line. The internal [CAD](#_bookmark326) [OID](#_bookmark355) is assigned to the resulting topology, and a [UUID](#_bookmark366) is assigned to the [shape\_aspect](#_bookmark418) that represents that hole feature in the [STEP](#_bookmark363) data set. One of four methods is used to modify the hole geometry.

Method 1:One or more of the hole attributes are modified. Because the internal [CAD](#_bookmark326) [OID](#_bookmark355) is maintained, the [UUID](#_bookmark366) remains as was initially defined, relying on the receiving system to audit the design for updates.

Method 2:One or more hole attributes are deleted and recreated while the hole feature definition remains. For example, the point that defines the location of the hole feature is deleted, and a new point is created. Because the internal [CAD](#_bookmark326) feature object remains, and its [OID](#_bookmark355) is maintained, the [UUID](#_bookmark366) does not require an update, relying on the receiving system to audit the design for updates.

Method 3: The hole is deleted, and a new hole feature is defined. Because the initial [CAD](#_bookmark326) feature is removed, a new [CAD](#_bookmark326) [OID](#_bookmark355) will be assigned to the new feature. The [UUID](#_bookmark366) associated to the initial [CAD](#_bookmark326) [OID](#_bookmark355) is flagged to become unavailable for future use, and a new [UUID](#_bookmark366) is associated to the new hole. The uuid\_relationship will support the capability to specify that the new hole feature supersedes the previous hole feature.

Method 4: A new feature is added to the model, resulting in a new [CAD](#_bookmark326) [OID](#_bookmark355). A [UUID](#_bookmark366) is assigned to that new feature. Each new [CAD](#_bookmark326) object would be assigned an [OID](#_bookmark355) and, when exported to [AP 242](#_bookmark322), a [UUID](#_bookmark366).

### Example 11

This example illustrates the application of method one on a simple test model. The model was implemented with guidance from the first release of a recommended practice for [UUID](#_bookmark366)43.

The model is a plate of 50.8 mm x 76.2 mm x 10 mm centered about the coordinate system. There is a 19 mm diameter thru hole located at 6.3 mm, 6.4 mm, 0 mm relative to the coordinate system. An external application or [CAD](#_bookmark326) plug-in identifies the hole feature using its [OID](#_bookmark355) and, following enterprise policy, issues a [UUID](#_bookmark366) for each [CAD](#_bookmark326) object of topology and annotation that defines the hole feature and its related [PMI](#_bookmark358), respectively. An image of the initial state of the [CAD](#_bookmark326) model is shown in figure *4*.

FIGURE 4

CAD model with PMI.

A drawing of a rectangular object with a hole

Description automatically generated

An extract of the topology and annotations related to the hole in version 0.0 of the [AP 242](#_bookmark322)

test file is shown in table [4](#_bookmark213).

TABLE 4

Extract of data from Revision 0.0

|  |  |  |
| --- | --- | --- |
| Element Class | PMI Values | Assigned UUID |
| Advanced face | … | 2dfe9b4e-fd8b-5006-b660- a0da88b5e2ad |
| Cylindrical surface | … | not provided |
| Hole Diameter | ∅ 19 | 814e27a0-acee-5455-a393-2e493d372f4e |
| Diameter tolerance | +/- 0.05 | not provided |
| Hole Vertex | 6.3,6.4,10 | 03338caf-ec0c-518a-a8eb-4b7bb170724a |
| Hole Vertex | 6.3,6.4,0 | 1a70557f-cc2b-57fe-8f80- fbeaaa60edce |
| Feature Control Frame (2) | … | df69d739-ee5e-560e-8670-8822cca41d0b |
| Datum Feature A | A | not provided |
| Datum Feature B | B | not provided |
| Datum Feature C | C | not provided |

The complete data set is available as “NX\_Plate\_w\_Hole\_UUIDS\_Rev\_0.0.stp” at the GitHub opensource repo <https://github.com/usnistgov/UUID>.

The parameters that locate the hole center point are modified in the CAD model during design iteration version 0.1. The thru-hole center parameters are changed to move the hole to -8.3 mm, 8.9 mm, and 10 mm relative to the coordinate system. An external application or [CAD](#_bookmark326) plug-in identifies the hole feature as having the same [OID](#_bookmark355), and therefore, the [UUID](#_bookmark366) for that feature remains unchanged to maintain traceability of the modifications. An extract of the relevant instance data from the [AP 242](#_bookmark322) file of iteration 0.1 is shown in table [*5*](#_bookmark215).

TABLE 5

Extract of data from Revision 0.1 illustrating change in hole location.

|  |  |  |
| --- | --- | --- |
| Element Class | PMI Values | Assigned UUID |
| Hole Vertex | 8.3,8.9,10 | 03338caf-ec0c-518a-a8eb-4b7bb170724a |
| Hole Vertex | 8.3,8.9,0 | 1a70557f-cc2b-57fe-8f80- fbeaaa60edce |

The complete data file is available as “NX\_Plate\_w\_Hole\_UUIDS\_Rev\_0.1.stp” at <https://github.com/usnistgov/UUID>.

In version 0.2, the hole feature is deleted from the CAD model. A new hole feature of the same size, with a new location -6.3 mm, 6.9 mm, 10 mm, is defined. Two scenarios result from differences in how [CAD](#_bookmark326) applications track changes.

All [PMI](#_bookmark358) remain in the model, and those whose references no longer exist become disconnected. The disconnected [PMI](#_bookmark358) are connected to the new hole geometry. Because the [PMI](#_bookmark358) remain, their [UUIDs](#_bookmark366) are maintained.

An extract of the relevant instance data from the [AP 242](#_bookmark322) file of iteration 0.2 is shown in table [*6*](#_bookmark216).

TABLE 6

Extract of data from Revision 0.2, including hole deletion and creation of new hole.

|  |  |  |
| --- | --- | --- |
| Element Class | PMI Values | Assigned UUID |
| Advanced face | … | 3ef5490e-c4c5-5208-8bdc-a75293cd144f |
| Cylindrical surface | … | not provided |
| Hole Diameter | ∅ 19±0.05 | 814e27a0-acee-5455-a393-2e493d372f4e |
| Hole Vertex | 8.3,8.9,10 | ec147a04-aaad-59f0-9417- e9c7cdde9501 |
| Hole Vertex | 8.3,8.9,0 | e3987c2b-bdd9-56ae-8d56-22e2443f972b |
| Feature Control Frame (7) | … | 21dba1d4-f8c9-5c39-9163-510b94836bfb |
| Datum Feature A | A | not provided |
| Datum Feature B | B | not provided |
| Datum Feature C | C | not provided |

The complete data file is available as “NX\_Plate\_w\_Hole\_UUIDS\_Rev\_0.2.stp” at <https://github.com/usnistgov/UUID>.

Version 0.3 adds a chamfer to one edge of the model. The chamfer is a new feature, and the CAD application identifies it with an [OID](#_bookmark355). Following enterprise policy, a new [UUID](#_bookmark366) is assigned to the new feature. Table 7 shows an extract of the relevant instance data from the AP 242 file of iteration 0.3.

TABLE 7

Extract of data from Revision 0.3 illustrating the addition of chamfer face.

|  |  |  |
| --- | --- | --- |
| Element Class | Related PMI | Assigned UUID |
| Advanced Face |  | 3ef5490e-c4c5-5208-8bdc-a75293cd144f |

The complete data file “NX\_Plate\_w\_Hole\_UUIDS\_Rev\_0.3.stp” is available at <https://github.com/usnistgov/UUID>.

# Results

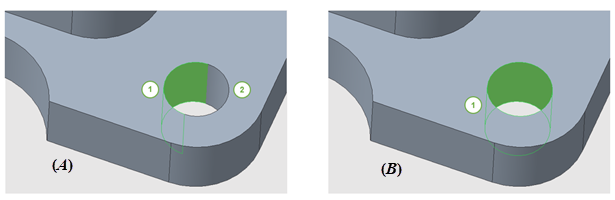
## Results of Recent CAx-IF Test Rounds

Implementation testing by CAx-IF participants has confirmed the utility of UUIDs. Initial testing focused on publishing UUIDs in STEP AP242 files for iterative design use (Round R52J). Subsequent testing (Rounds 53J and R54J) has shown improved export capability, particularly for geometry and topology data, initial examples of publishing UUIDs for PMI data, and early testing of consumption of entities with UUIDs by other systems. Analysis of test results confirms the ability to trace STEP entities from iteration to iteration and from one application to another. The analysis suggests, however, that additional testing focus is required to ensure that identifiers can be mapped appropriately when topology changes occur. Such topological changes may occur between iterations when model design changes or across system boundaries, even with the same iteration, when publishing and consuming applications have different modeling kernels (see the example below). In either case, research has shown that publishing and consuming applications must be careful to account for such changes, and the persistent ID schema has been updated by introducing relationships between entities and their UUIDs to document change, e.g., derive\_from, merge, same\_as, similar\_to, split, and supersedes, allowing traceability to be preserved.

Figure *5* shows an example of the effects of feature representation and UUID assignment. Some systems depict a hole as two half-cylinders and some as a single cylinder. A variation of surface counts arises from differing vendor definitions of topology. Surface count differences can also occur when IDs are assigned to all shape aspects. However, IDs should only be applied where geometry is merged or split.

FIGURE 5

Varying representation of hole topology. (*A*) Hole represented as two half cylinders, (*B*) Hole represented as a cylinder.



Ongoing research and testing will exercise publishing and maintenance of these relationships during transfers to ensure full integration in STEP and reliable data exchange, particularly for roundtrip scenarios.

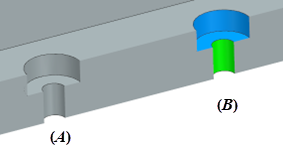
Downstream applications, such as manufacturing and inspection applications, are particularly interested in persistent IDs on PMI data. Another observation from testing is that no vendors, but not all, fully support this capability internally. Further research will seek to understand what options may be used to mitigate this gap.

Another aspect discovered during research and testing is that, in cases of more complex features, such as counterbore holes shown in figure *6*, CAD systems may define the geometry as one feature, and that single complex hole feature may or may not be useful during manufacturing planning (depending on available manufacturing processes). Still, those complex features will generally require decomposition for inspection planning purposes. Determining if each cylinder requires a unique UUID or if the geometry should be combined to have one UUID is under discussion, and systems will likely need to be configurable to allow complex features to be retained or decomposed depending on resource availability and process flexibility.

FIGURE 6

Cross-section of holes with identical geometry. (*A*) Counterbore hole defined as one feature, (*B*) Counterbore hole defined as two cylinders.

A screenshot of a computer

Description automatically generated 

Some vendors still rely on GUIDs, which are not aligned with the developing schema. STEP data processing through specific kernels, like S/B Spatial 3D Interop/ACIS, may also introduce bottlenecks when handling persistent identifiers (PIDs).

UUID propagation rules, including when to retain existing UUIDs versus assigning new ones, remain under review, as specific cases demand careful handling. The decomposition of geometry, especially for inspection applications and QIF compatibility, also requires attention.

For assemblies with external references, it is essential to ensure that the IDs for parts and part versions in the Part 21 file align with those used in the AP242 XML assembly structure.

Additionally, general notes, such as "Unless Otherwise Specified," which are intended for human interpretation, must be applied to features in a machine-readable format to capture all requirements comprehensively.

# Discussion

## Model Implementation Considerations

During the CAX-IF round interoperability testing, the implementors identified items needing an update in the test models and experimental implementations.

The artifacts were updated, and testing was repeated.

One item of note was an addition to the AP 242 data model for providing a single UUID for a set of closely related items. The initial use of this addition is to support data exchange in the shared design collaboration use case where CAD system A provides a periodic representation for a hole, and CAD system B provides an aperiodic representation. It also supports the case where additional topological elements are provided in the hole representation. The capability avoids the complexity of using an array of items in the uuid\_attribute, thus reducing interpretation confusion and maintenance costs and implementation cycle time, by not applying an array (which is ordered) for a use case where the nature of the use case conforms to a set usage.

# Conclusion and Future Research

In this paper, we have described research and recommendations for using UUIDs in product data standards in the design to manufacturing and inspection workflow and examined industrial use cases to discover requirements for using [UUIDs](#_bookmark366) in product data standards.

We discussed [UUID](#_bookmark366) encoding requirements and characteristics, types of [UUIDs](#_bookmark366), how they are used with human-readable identifiers, and what requirements they address

The identification of detailed recommendations for adding [UUIDs](#_bookmark366) to a product model is a novel contribution. However, there is much left to do. In our research, we recommended the behavior of supporting software applications that do not currently exist and will be challenging to implement. In the exchange example “Shared design collaboration,” use case methods 3 and 4 require the ability to maintain traceability that is not available in software. Exporting and importing multiple model types (such as native CAD, neutral formats, and others) from diverse sources and their variations in geometry definition is not trivial and requires significant effort to track.

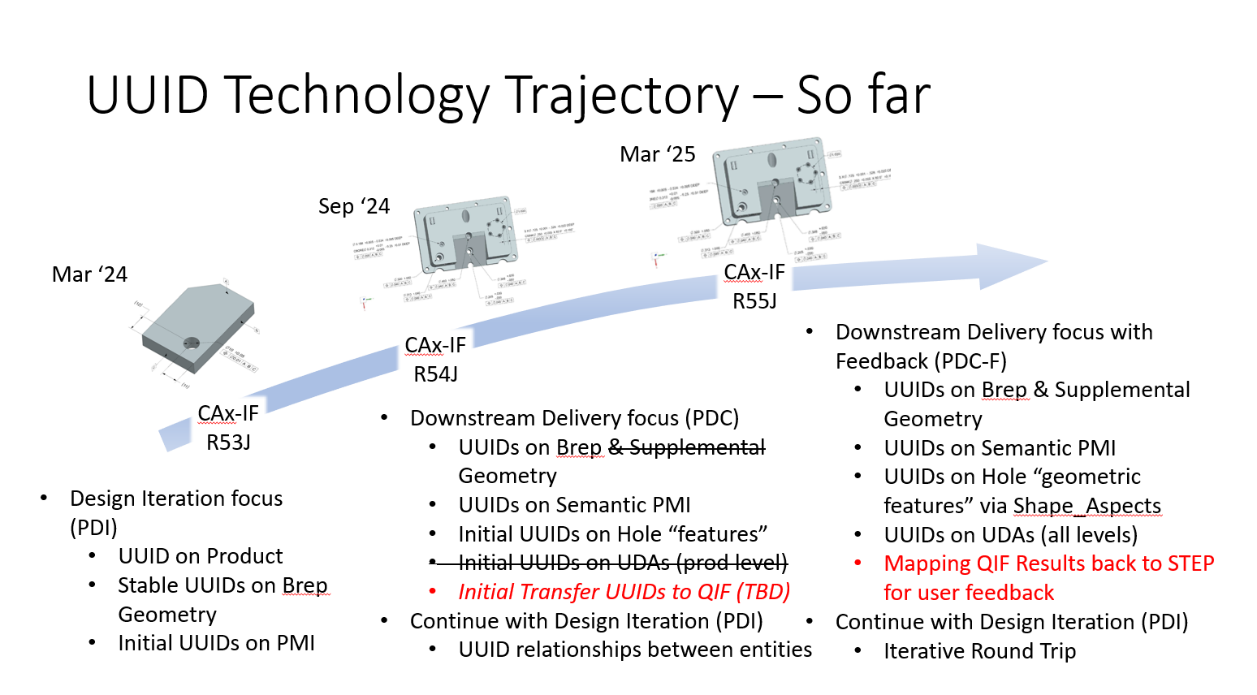
When sending the data to a [CMS](#_bookmark333), each annotation and its associated geometry in the model must be matched to one of the hundreds or thousands of configured machines, [PMI](#_bookmark358) requirements applied, and a collision-free measurement solution generated. The software systems utilizing this information must then manage gigabytes (possibly terabytes) of critical as-measured data, converted to actionable information, from multiple measurement sources in real-time, all while maintaining rock-solid persistence and interoperability. The amount of detailed data collected by an enterprise or supply chain will vary as the product design and the supply chain communication network matures.

In the near term, validation of the new [AP 242](#_bookmark322) [UUID](#_bookmark366) constructs will continue in [CAx-IF](#_bookmark330)46 interoperability test rounds, as shown in figure *7*. The earlier [CAx-IF](#_bookmark330) test rounds focused on design iteration. In round R54J, the [CAx-IF](#_bookmark330) experimented with multi-domain interoperability by manually testing the delivery of [UUIDs](#_bookmark366) to metrology applications through [QIF](#_bookmark359). In the coming round R55J, the STEP to QIF multi-domain testing will mature, and, depending on its success, manual testing of feedback from metrology applications back to design will be attempted, and round-trip design iteration will also be explored.

Additionally, approaches for enhancing the identification and communication of object state changes using Merkle trees will be investigated to move further along the path to concept-based change management.

FIGURE 7

CAx-IF UUID test round objectives.



# References

1. Industrial automation systems and integration – Product data presentation and exchange – Part 1: Overview and fundamental principles, ISO 10303-1(2024) (Geneva, Switzerland: International Organization for Standardization, approved January 2024). <https://www.iso.org/standard/83105.html>
2. Industrial automation systems and integration – Product data representation and exchange – Part 209: Application protocol: Multidisciplinary analysis and design, ISO 10303-209(2014) (Geneva, Switzerland: International Organization for Standardization, approved December 2014). <https://www.iso.org/standard/59780.html>
3. Industrial automation systems and integration – Product data representation and exchange – Part 238: Application protocol: Managed model-based integrated manufacturing, ISO 10303-238(2022) (Geneva, Switzerland: International Organization for Standardization, approved September 2022). <https://www.iso.org/standard/84898.html>
4. Industrial automation systems and integration – Product data representation and exchange – Part 242: Application protocol: Managed model-based 3D engineering, ISO 10303-242(2022) (Geneva, Switzerland: International Organization for Standardization, approved December 2022). <https://www.iso.org/standard/84667.html>
5. *MTConnect Standard Version 2.2.0,* ANSI MTC1.4-2018(2023) (MTConnect Institute: approved August 8, 2023). [https://do](https://docs.mtconnect.org/MBSD_MTConnect_Part_1_2-2-0.pdf) [cs.mtconnect.org/MBSD\_MTConnect\_Part\_1\_2-2-0.pdf](https://docs.mtconnect.org/MBSD_MTConnect_Part_1_2-2-0.pdf)
6. *Quality Information Framework Standard (QIF) – An integrated model for manufacturing quality information,* ANSI (2020) (Rochester, NY: Dimensional Metrology Standards Consortium, approved August 2020). <https://qifstandards.org/qif-download>
7. *Quality Information Framework Standard (QIF) – An integrated model for manufacturing quality information,* ISO 23952 (2020) (Geneva, Switzerland: International Organization for Standardization, approved July 2020). [https://www.iso.org/standard/7](https://www.iso.org/standard/77461.html) [7461.html](https://www.iso.org/standard/77461.html)
8. *Business Process Model and Notation v2.0.2*, (2023) (Milford, MA: Object Management Group, approved December 2013). <https://www.omg.org/spec/BPMN/2.0.2/PDF>
9. N Juty, SM Wimalaratne, S Soiland-Reyes, J Kunze, C. Goble, T Clark, “Unique, persistent, resolvable: Identifiers as the foundation of FAIR,” *Data Intelligence* 2, no. 1-2 (January 01, 2020):30-39. <https://doi.org/10.1162/dint_a_00025>
10. T Hedberg Jr, ME Sharp, TM Maw, MM Helu, MM Rahman, S Jadhav, JJ Whicker, A Barnard Feeney, “Defining requirements for integrating information between design, manufacturing, and inspection,” *International Journal of Production Research* 60, no. 11 (May 11, 2021): 3339-3359, <https://doi.org/10.1080/00207543.2021.1920057>
11. SH Suh, JH Cho, HD Hong “On the architecture of intelligent STEP-compliant CNC,” *International Journal of Computer Integrated Manufacturing* 15, no. 2 (November 08, 2010):168–177, <https://doi.org/10.1080/09511920110056541>
12. A Trainer, *“Validation for downstream computer aided manufacturing and coordinate metrology processes,”* NISTGCR 15-1009, National Institute of Standards and Technology, (Gaithersburg, MD, 2016). <https://doi.org//10.6028/NIST.GCR.16-003>
13. A Barnard Feeney, SP Frechette, V Srinivasan, “A portrait of an ISO STEP tolerancing standard as an enabler of smart manufacturing system,” *Journal of Computing and* *Information Science in Engineering* 15, no. 2 (June 2015):021001, <https://doi.org/10.1115/1.4029050>
14. WD Williams, “A business case for long-term archiving & retrieval of the model-based enterprise’s data,” (Sandia National Lab, Albuquerque, NM, February 2012). [https:](https://www.osti.gov/servlets/purl/1106409)[//www.osti.gov/servlets/purl/1106409](https://www.osti.gov/servlets/purl/1106409)
15. A Trainer, T Hedberg, A Barnard Feeney, K Fischer, P Rosche, “Gaps analysis of integrating product design, manufacturing, and quality data in the supply chain using model-based definition,” *ASME 2016 11th International Manufacturing Science and Engineering Conference – Volume 2: Materials; Biomanufacturing; Properties, Applications and Systems; Sustainable Manufacturing - MSEC2016-8792, V002T05A003*. (September 2016). [https://doi.org/10.1115/](https://doi.org/10.1115/MSEC2016-8792) [MSEC2016-8792](https://doi.org/10.1115/MSEC2016-8792)
16. T Hedberg Jr, J Lubell, L Fischer, L Maggiano, A Barnard Feeney, “Testing the Digital Thread in Support of Model-Based Manufacturing and Inspection,” *Journal of Computing and Information Science in Engineering* 16, no. 2 (June, 2016):021001, <https://doi.org/10.1115/1.4032697>
17. K Fischer, P Rosche, A Trainer, A Barnard Feeney, T Hedberg Jr, “Investigating the impact of standards-based interoperability for design to manufacturing and quality in the supply chain,” NISTGCR 15-1009 (2015) (Gaithersburg MD). <https://doi.org/10.6028/NIST.GCR.15-1009>
18. *Industrial systems, installations and equipment, and industrial products – Identification of terminals within a system,* IEC 61666:2010 (2010) (International Electrotechnical Commission, approved August 19, 2010) <https://webstore.iec.ch/en/publication/5705>
19. Industrial automation systems and integration – Product data representation and exchange – Part 210: Application protocol for electronic interconnect, assembly, and packaging design, ISO 10303-210:2021(2021) (Geneva, Switzerland: International Organization for Standardization, approved March 2021). <https://www.iso.org/standard/72406.html>
20. *Composite Part Drawings,* ASME Y14.37(2019) (The American Society of Mechanical Engineers, 2019). <https://www.asme.org/codes-standards/find-codes-standards/y14-37-composite-part-drawings/2019/pdf>
21. *Technical product documentation – Digital product definition data practices*, ISO 16792 (2021) (Geneva, Switzerland: International Organization for Standardization, April 2021). https://www.iso.org/standard/73871.html
22. *Geometrical product specifications (GPS) – Geometrical tolerancing – Datums and datum systems*, ISO 5459(2011) (Geneva, Switzerland: International Organization for Standardization, August 2011). <https://www.iso.org/standard/40358.html>
23. DW Gillman, F Farance, “Metadata and data harmonization” (2009) https:[//nces.ed.gov/FCSM/pdf/2009FCSM\_Gillman\_X-A.pdf](https://nces.ed.gov/FCSM/pdf/2009FCSM_Gillman_X-A.pdf)
24. *Measurement Data Reporting*, ASME Y14.45(2021) (The American Society of Mechanical Engineers, 2021). <https://www.asme.org/codes-standards/find-codes-standards/y14-45-measurement-data-reporting/2021/pdf>
25. *First Article Inspection Requirement Rev C*, SAE AS9102C (2023) (SAE International, approved June 28, 2023). <https://store.accuristech.com/standards/sae-as9102c?product_id=2568218>
26. Industrial systems, installations and equipment and industrial products – Structuring principles and reference designations: Part 1: Basic rules, ISO 81346-1(2022) (Geneva, Switzerland: International Organization for Standardization, approved October March 2022). <https://www.iso.org/standard/82229.html>
27. EF Codd, “A relational model of data for large shared data banks,” *Communications of the ACM* 13(6):377–387, (1970).
28. Information technology – Procedures for the operation of object identifier registration authorities: Part 8: Generation of universally unique identifiers and their use in object identifiers, ISO/IEC 9834-8:2014(2014) (Geneva, Switzerland: International Organization for Standardization and International Electrotechnical Commission, approved August 2014). <https://www.iso.org/standard/62795.html>
29. K Davis, B Peabody, P Leach, “Request for Comment 9562: Universally Unique IDentifiers (UUIDs)” (Internet Engineering Task Force (IETF), 2024).
30. Q Dang, *Secure Hash Standard*, NIST FIPS 180-4 (Gaithersburg, MD: U. S Department of Commerce, National Institute of Standards and Technology, 2015). <https://doi.org/10.6028/NIST.FIPS.180-4>
31. T Hedberg Jr, S Krima, JA Camelio, “Method for enabling a root of trust in support of product data certification and traceability,” *Journal of Computing and Information* *Science in Engineering* 19, no. 4 (December 2019):041003, <https://doi.org/10.1115/1.4042839>
32. Information technology – Procedures for the operation of object identifier registration authorities: Part 1: General procedures and top arcs of the international object identifier tree, ISO/IEC 9834-1:2012(2012) (Geneva, Switzerland: International Organization for Standardization and International Electrotechnical Commission, approved May 2012). <https://www.iso.org/standard/58055.html>
33. WF Danner, DT Sanford, Y Yang “*STEP (STandard for the Exchange of Product Model Data) Resource Integration: Semantic & Syntactic Rules*,” (Gaithersburg, MD: U.S. Department of Commerce, National Institute of Standards and Technology, 2016). <https://doi.org/10.6028/NIST.IR.4528>
34. Industrial automation systems and integration – Product data representation and exchange – Part 21: Implementation methods: Clear text encoding of the exchange structure, ISO 10303-21(2016) (Geneva, Switzerland: International Organization for Standardization, approved March 2016). <https://www.iso.org/standard/63141.html>
35. TR Thurman, AG Trainer, A Barnard Feeney, M Hardwick, M Hedlind, R Astheimer, *Research Results and Recommendations for Universally Unique Identifiers in Product Data Standards, NIST AMS 300-12* (Gaithersburg, MD: U.S. Department of Commerce, National Institute of Standards and Technology, 2024). <https://doi.org/10.6028/NIST.AMS.300-12>
36. A Trainer, G Krishnan, Y Varvak, S Berkeley. Method and System for Management of Heterogeneous Assemblies. U. S. Patent 6,473,673, filed May 24, 1999, and issued October 29, 2002.
37. A Trainer, G Krishnan, Y Varvak, S Berkeley. Methods and Systems for Managing Synchronization of a Plurality of Information Items of a Computer-Aided Design Data Model. U. S. Patent 8,818,769. Filed October 28, 2010, and issued May 3, 2012.
38. ISO TC 184/SC 4 N532:1997 Guidelines for application interpreted model development.
39. Industrial automation systems and integration – Product data representation and exchange – Part 1824: Application module: Change management, ISO 10303-1824(2019) (Geneva, Switzerland: International Organization for Standardization, approved November 2010). <https://www.iso.org/standard/78671.html>
40. T, Boone, “Discussion of ‘Technical data package’ at fall workshop summary” (Air Force Research Lab, 2022-5873, 2022).
41. J De Nijs, “Enabling Advanced Analytics Use Cases that Require Manufacturing Data”, (Copyright Lockheed Martin 2024, PIRA CET2024020117, 2022).
42. R Kirkwood, J Sherwood, “Sustained CAD/CAE integration: integrating with successive versions of STEP or IGES files,” *Engineering with Computers* 34 (March 29, 2017):1–13, <https://doi.org/10.1007/s00366-017-0516-z>
43. A Trainer, T Thurman T, R Lipman, P Rosche, J Boy, *“CAx-IF Recommended Practices for Persistent IDs for Design Iteration and Downstream Exchange, Release 1.0”,* (CAx Interoperability Forum, 2024). [https://www.mbx-if.org/home/wp-content/u ploads/2024/05/rec\_pracs\_PID\_v1.pdf](https://www.mbx-if.org/home/wp-content/u%20ploads/2024/05/rec_pracs_PID_v1.pdf)
44. J Michaloski, First Article Inspection Requirement Report Generation from QIF Using C++, CodeSynthesis, and Mozilla Xerces (2016)
45. Thurman T, Trainer A, Barnard Feeney A, Astheimer R (2016) First Article Inspection Requirement Report Generation from QIF Using C++, CodeSynthesis, and Mozilla Xerces. Available at <https://github.com/usnistgov/QIF>
46. MBx Interoperability Forum, “CAx Interoperability Forum.” 2024, <https://www.mbx-if.org/home/cax/>

1. TRThurman Consulting, Marion, Iowa, 52302, USA; ORCID 0000-0002-1550-9913 [↑](#footnote-ref-1)
2. Trainer Engineering Associates, Dighton, MA, 02715, USA; ORCID 0009-0009-7060-7493 [↑](#footnote-ref-2)
3. Smart Connected Systems Division, National Institute of Standards and Technology, Gaithersburg, MD, 20899, USA; ORCID iD 0009-0001-0534-5054 [↑](#footnote-ref-3)
4. Smart Connected Systems Division, National Institute of Standards and Technology, Gaithersburg, MD, 20899, USA; ORCID iD 0000-0002-0866-9572 [↑](#footnote-ref-4)