Quality Information Framework (QIF) – An Integrated Model for Manufacturing Quality Information

Part 2: Quality Information Framework (QIF) Library – Information Model and XML Schema Files Version 2.0

QIF Version 2.0

ANSI/QIF Part 2–2014

Dimensional Metrology Standards Consortium, Inc. (DMSC)

Prepared and Published by:
Dimensional Metrology Standards Consortium, Inc. (DMSC)
1350 Alsbury Blvd., #514
Burleson, Texas 76028
Tel: (817) 461-1092
email: support@qifstandards.org

www.QIFStandards.org

Contents

Foreword v

Introduction vii

1 Scope 1

2 Conformance 2

3 Normative References 3

4 Terms and Definitions 4

5 Symbols and Abbreviated Terms 4

6 QIF Library XML schema files 5

6.1 Auxiliary.xsd 5

6.2 Characteristics.xsd 6

6.2.1 Characteristics *element* 6

6.2.2 Characteristic definitions, nominals, items, and actuals 6

6.2.3 DefaultCharacteristicDefinitions 9

6.2.4 DefaultToleranceDefinitions 9

6.2.5 CharacteristicGroups 9

6.2.6 Constraint checking for characteristics 10

6.2.7 ToleranceZones 10

6.2.8 Substitution groups for characteristics 10

6.3 Expressions.xsd 11

6.4 Features.xsd 11

6.4.1 Features *element* 11

6.4.2 Feature types 12

6.4.3 Constraint checking for features 14

6.4.4 Feature construction methods 14

6.4.5 Substitution groups for features 15

6.5 GenericExpressions.xsd 15

6.6 Geometry.xsd 15

6.7 IntermediatesPMI.xsd 17

6.8 Primitives.xsd 18
6.9 PrimitivesPD.xsd 19
6.10 PrimitivesPMI.xsd 19
6.11 Statistics.xsd 20
6.11.1 Characteristic Statistics Evaluation Types 20
6.11.2 Criterion types 21
6.11.3 Other items 22
6.12 Topology.xsd 22
6.13 Traceability.xsd 23
6.13.1 *InspectionTraceabilityType* 23
6.13.2 *PreInspectionTraceabilityType* 24
6.13.3 *ProductTraceabilityType* 25
6.13.4 *ActualProductTraceabilityType* 25
6.13.5 *ManufacturingProcessTraceabilityType* 26
6.14 Units.xsd 27
6.14.1 FileUnits 27
6.14.2 Conversions 29
6.14.3 FileUnitsExample 29
6.14.4 Instance File Example Using Units 30
6.15 Visualization.xsd 31
7 Data dictionary for QIF Library Data Model 33
Annex A - Location of QIF 2.0 XML Library schema files 34
Annex B - Graphical conventions used in the data dictionary 35
Bibliography 38

Figures

Figure 1 – Characteristics element 6
Figure 2 – Characteristic types 9
Figure 3 – Tolerance zone types 10
Figure 4 – Features element 12
Figure 5 – Comparison of feature definitions in QIF and DMIS 13
Figure 6 – Hierarchy of type definitions for feature data object types (example) 14
Figure 7 – Individual geometry types 16
Figure 8 – Geometry set types 16
Figure 9 – Substitute feature algorithms 17
Figure 10 – Datum targets 17
Figure 11 – Alignment operations 18
Figure 12 – Datums and datum reference frames 18
Figure 13 – Types with enumeration or user definition 19
Figure 14 – CharacteristicStatsEval types (continued on next page) 20
Figure 15 – Criterion types 22
Figure 16 – Individual topology types 22
Figure 17 – Topology set types 23
Figure 18 – Elements of InspectionTraceabilityType 24
Figure 19 – Elements of *PreInspectionTraceabilityType* 25
Figure 20 – Elements of *ProductTraceabilityType* 25
Figure 21 – Elements of *ActualProductTraceabilityType* 26
Figure 22 – Elements of *ManufacturingProcessTraceabilityType* 26
Figure 23 – FileUnits element 27
Figure 24 – Derivation hierarchy of values with units 28
Figure 25 – Conversion of units 29
Figure 26 – FileUnits snippet 30
Figure 27 – Instance file snippets using units 31
Figure 28 – VisualizationSet 32

Foreword

The Dimensional Metrology Standards Consortium (DMSC, Inc.) is an American National Standards Institute (ANSI) accredited standards developing organization, as well as an A-Liaison to the International Organization for Standardization (ISO). The mission of the DMSC is to identify urgently needed standards in the field of dimensional metrology, and to promote, foster, and encourage the development and interoperability of these standards, along with related and supporting standards that will benefit the industry as a whole. More information about the DMSC can be found at www.dmisstandard.org.

The Quality Information Framework (QIF) was developed by domain experts from the manufacturing quality community representing a wide variety of industries and quality measurement needs. Contributors to this document of version 2.0 of the QIF standard include:

- Capvidia
- Honeywell Federal Manufacturing and Technology
- Lockheed Martin Missiles and Fire Control
- Metrosage
- Mitutoyo America Corp.
- National Institute of Standards and Technology
- Origin International Inc.

More information about DMSC's QIF effort can be found at www.qifstandards.org.

The bulk of the work on this document was performed by the Complete and Accurate Model Based Design (CAMBD) Working Group and the Quality Information Framework (QIF) Working Group, revised as needed and given final approval for ANSI balloting by the DMSC's Quality Measurement Standards (QMS) Committee.

The QIF standard, version 2.0, consists of the following parts under the general title Quality Information Framework (QIF) — An Integrated Model for Manufacturing Quality Information:

Part 1: Overview and Fundamental Principles Version 2.0

Part 2: QIF Library Information Model and XML Schema Files Version 2.0

Part 3: QIF Model Based Definition (MBD) Information Model and XML Schema File Version 2.0

Part 4: QIF Plans Information Model and XML Schema File Version 2.0

Part 5: QIF Resources Information Model and XML Schema File Version 2.0

Part 6: QIF Rules Information Model and XML Schema File Version 2.0

Part 7: QIF Results Information Model and XML Schema File Version 2.0

Part 8: QIF Statistics Information Model and XML Schema File Version 2.0

Parts 1 and 2 describe the overview and central concepts of the QIF. Parts 3 through 8 describe information models for the six *application areas* of QIF, namely MBD, Plans, Resources, Rules, Results, and Statistics.

The inaugural QIF standard, version 1.0, was published in 2013. This document is a component of the second release of the QIF suite of standards, denoted version 2.0. The QIF version 2.0 documents cancel and replace all documents of version 1.0. QIF version 2.0 is solely a product of the DMSC and its committees and working groups.

Each major release of the QIF standard is composed of several *Parts* documents. Individual *Parts* are designated with the version number of the major QIF revision, even when the document is new. QIF version 2.0 includes revisions of the version 1.0 documents Part 1 (Overview), Part 2 (QIF Library), Part 3 (Plans) and Part 4 (Results), and 4 new Parts that did not exist in version 1.0. Also, QIF Part 3 – 2013 was re-designated as Part 4 and its name was changed from QMPlans to QIF Plans. QIF Part 4 – 2013 was re-designated as Part 7 and its name was changed from QMResults to QIF Results.

This version of Part 2, designated QIF Part 2 – 2014, is a revision of the original standard QIF Part 2 – 2013. Its scope was expanded to cover the new XML schema files. Its content was revised to reflect changes made to the QIF information model. In QIF version 1.0 the QIF Library contained eight schema files. With the addition of model based design, statistics, and rules in version 2.0, the number of schema files in the library increased to fifteen, some names were changed, and some type definitions were transferred from one file to another.

HTML-based data model viewers

The DMSC will make available an html-file based data dictionary for the entire QIF information model as an aid to understanding QIF. This data dictionary is non-normative material, but describes the normative content of the QIF data model. The html files facilitate viewing the complete data model, including all six application areas and Library content, using pictures and text. A user has the ability, through an internet browser, to follow navigation links forward and backward through the data model description using mouse clicks.

Introduction

QIF version 2.0 approaches system-wide interoperability by partitioning the data model between a library of fifteen common, reusable components, and six data models for unique application areas, all connected by the QIFDocument model. The reusable library components are referenced throughout the comprehensive quality data model, thereby ensuring interoperability between any data producers and consumers that implement the QIF formats in their software.

This Part of QIF deals with the QIF Library.

The information covered by files in the QIF Library encompasses:

- boundary representation models of the sort found at the core of commercial computer-aided design (CAD) systems
    - geometry
    - topology
    - mesh representations
    - visualization
- product manufacturing information (PMI)
    - geometric dimensioning and tolerancing information as expressed in ASME Y14.5M-1994
    - Most of the material in ASME Y14.5-2009
    - information expressed by the DMIS 5.2 standard
    - first article inspection report information as defined in the AS9102a standard.
- rules to use in generating quality plans and programs
    - basic generic expressions
    - expressions specific to QIF rules
- quality statistics

The QIF Library, and QIF in general cover both pre-measurement and post-measurement PMI information.

Quality Information Framework (QIF) — An Integrated Model for Manufacturing Quality Information

Part 2: Quality Information Framework (QIF) Library — Information Model and XML Schema Files

1 Scope

This standard defines a data dictionary, centered on dimensional metrology, but not limited to it. The data dictionary is a human-readable data model of manufacturing quality data. The data model consists of definitions of data types (*simpleType* or *complexType* in XSDL), *element*s, and the logical relationships between them. This standard uses XML schema definition language (XSDL) to define the structure of the data model, and the rules for writing and reading XML instance files that are suitable for computer-to-computer conveyance of manufacturing data. Semantic information is defined by the in-line documentation, which will allow solution providers to automate metrology interoperability.

This Part of the standard defines the QIF Library, which is a collection of partial models, each of which contains related information items. Each partial model is contained in a separate XML schema file. These partial models have been designed to be used as components of complete QIF application area models. The files and the area covered by each are:

- Auxiliary.xsd – types for auxiliary geometry
- Characteristics.xsd – characteristic types
- Expressions.xsd – expression types specific to QIF, used in the QIF rules application
- Features.xsd – feature types
- GenericExpressions.xsd – domain-independent arithmetic and Boolean expression types
- Geometry.xsd – geometric types (half of the core of CAD)
- IntermediatesPMI.xsd – miscellaneous PMI types with ids
- Primitives.xsd – basic types used throughout QIF
- PrimitivesPD.xsd – basic numerical types used in product description
- PrimitivesPMI.xsd – basic PMI types used in QIF
- Statistics.xsd – types that support QIF statistics
- Topology.xsd – topology types (the other half of the core of CAD)
- Traceability.xsd – quality measurement traceability information
- Units.xsd – types for units of measurement and values with units
- Visualization.xsd – types for graphical display of products, including PMI

The XML schema files are normative material of the QIF standard, but their content is not replicated in this document due to its large size. DMSC will publish all QIF XML schema files in a compressed folder file named “QIF_2.0_XMLSchemaFiles.zip” which is available at www.qifstandards.org.

2 Conformance

Software programs that implement this specification to write QIF XML instance files must:

- follow the rules of XML when writing QIFDocument instance files
- generate instance files that validate against the QIFDocument schema
- employ semantics of the information written that complies with the referenced standards and with the QIF data dictionary in this specification.

Software programs that implement this specification to read QIF instance files must:

- be able to read any valid QIF XML instance file and extract all numerical and semantic data correctly.

3 Normative References

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ANSI/DMIS 105.2, Part 1-2009, *Dimensional Measuring Interface Standard, DMIS 5.2 Standard, Part 1*. Also available as ISO 22093:2011 *Industrial automation systems and integration -- Physical device control -- Dimensional Measuring Interface Standard (DMIS)*

ASME B1.7 - 2006, *Screw Threads: Nomenclature, Definitions, and Letter Symbols*

ASME Y14.36 - 1996, *Surface Texture Symbols*

ASME Y14.6 - 2001, *Screw Thread Representation*

ASME Y14.5M-1994 (reaffirmed 2004), *Dimensioning and Tolerancing - Engineering Drawing and Related Documentation Practices*

ASME Y14.5-2009, *Dimensioning and Tolerancing - Engineering Drawing and Related Documentation Practices*

ASME Y14.41 (2003), *Digital Product Definition Data Practices*

Extensible Markup Language (XML) 1.0 (Fifth Edition), W3C Recommendation 26 November 2008

ISO/IEC 9834-8:2008. *Information technology -- Open Systems Interconnection -- Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components*

ISO/IEC 11578:1996: "Information technology - Open System Interconnection - Remote Procedure Call (RPC)"

ISO/IEC Guide 99:2007 (E/F) – *International vocabulary of metrology – Basic and general concepts and associated terms* (VIM)

XML Schema Part 1: Structures Second Edition, W3C Recommendation 28 October 2004

XML Schema Part 2: Datatypes Second Edition, W3C Recommendation 28 October 2004

4 Terms and Definitions

Terms and definitions used in this document are given in Part 1 of this standard.

5 Symbols and Abbreviated Terms

ANSI	American National Standards Institute
ASCII	American Standard Code for Information Interchange
ASME	American Society of Mechanical Engineers
CAD	Computer-Aided Design
CAIPP	Computer-Aided Inspection Process Planning
CAM	Computer-Aided Machining or Computer-Aided Manufacturing
CMM	Coordinate Measuring Machine
COTS	Commercial Off-The-Shelf
DME	Dimensional Measuring Equipment
DMIS	Dimensional Measuring Interface Standard
DMSC	Dimensional Metrology Standards Consortium
DRF	Datum Reference Frame
ERP	Enterprise Resource Planning
GD&T	Geometric Dimensioning and Tolerancing
GPS	Geometrical Product Specifications
GUID	Globally Unique Identifier
ISO	International Organization for Standardization
MES	Manufacturing Execution Systems
MRI	Measurement Resources Information
MRP	Materials Resource Planning
MSA	Measurement Systems Analysis
PDPMI	Product Definition with Product Manufacturing Information

PMI	Product Manufacturing Information
QIF	Quality Information Framework
QMS	Quality Measurement Standards (a DMSC committee)
QPId	QIF Persistent Identifier
R&R	Repeatability and Reproducibility
SI	The International Systems of Units
SPC	Statistical Process Control
SQC	Statistical Quality Control
STEP	Standard for the Exchange of Product model data (ISO 10303)
UUID	Universally Unique Identifier
XML	eXtensible Markup Language
XSDL	XML Schema Definition Language

6 QIF Library XML schema files

There are fifteen XML schema files in the QIF Library. This section presents major details about each of the schema files. The sizes of these files vary widely, from less than 300 lines for Auxiliary.xsd to well over 10,000 lines for Features.xsd and Characteristics.xsd. The sizes of the subsections of this section vary similarly.

The descriptions of the contents of the schema files in this section are high-level to give the reader a general understanding of each library file. For full details of every type, the data dictionary files (preferably the HTML version, which provides two-way navigation) should be used. Annex A describes the graphical conventions used in the QIF data dictionaries. Part 1, Sections 4.3 and 4.4 describe terms defined in the XML schema definition language standard.

6.1 Auxiliary.xsd

The Auxiliary.xsd schema file contains types for:

- auxiliary geometry (points, lines and planes)
- CAD coordinate systems

Auxiliary geometry is geometry items that a user might want to see that are not in the boundary representation of an object, such as the axis of a cylinder. This includes, for example, ***PointAuxiliaryType***, ***LineAuxiliaryType***, and ***PlaneReferenceType***.

CAD coordinate systems are coordinate systems defined in CAD and included in QIF to support CAD to CAD and CAD to DME transfer.

6.2 Characteristics.xsd

6.2.1 Characteristics *element*

Characteristic information in the QIF data model is modeled in the Characteristics.xsd schema file. The top level type defined in the file is the ***CharacteristicAspectsListsType***. This is used in the QIFDocument schema as the type of the **Characteristics** *element*, as shown in Figure 1.

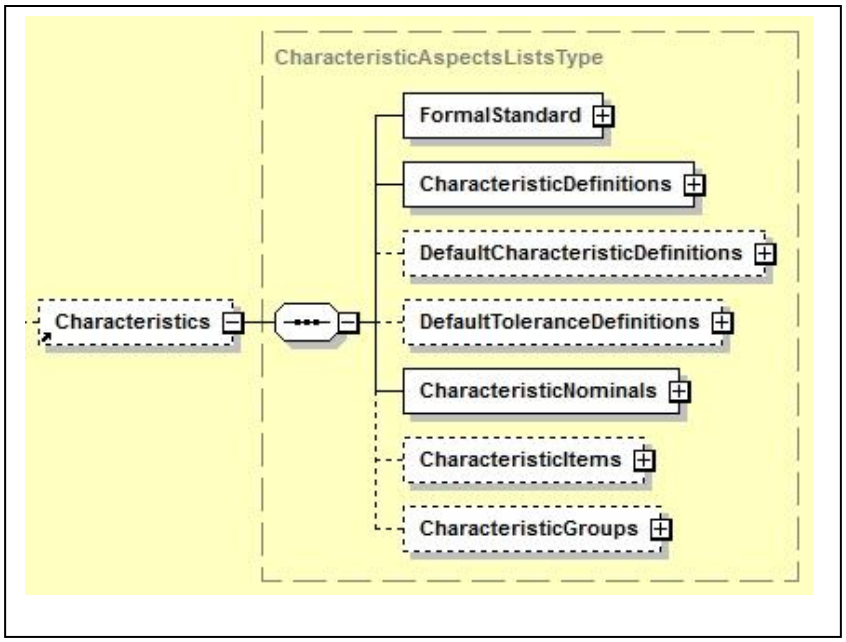

Figure 1 – Characteristics element

6.2.2 Characteristic definitions, nominals, items, and actuals

Characteristics have four aspects: *definition*, *nominal*, *actual*, and *item*. How the aspects relate to one another is described in Part 1, Section 6.7.3. Lists of three of the four aspects are in the **Characteristics** *element*. The missing aspect, *actual*, is in measurement results. The **Characteristics** *element* also contains default characteristics, default tolerances, and characteristic groups.

The XML Schema language definitions for characteristics form a type hierarchy by using XSDL *extension*s to derive more specialized types from more general types. There are four hierarchies, one for each aspect. Each of the four is headed by ***CharacteristicBaseType***. The prototype followed by each of the four hierarchies is shown in Figure 2. The word ***Aspect*** is shown in the names in the figure where one of the four aspect names (***Item***, ***Definition***, ***Nominal***, or ***Actual***) would appear. In the figure, inheritance is indicated by indenting an additional level. For example, the inheritance path to ***CylindricityCharacteristicAspectType*** is:

CharacteristicBaseType
CharacteristicAspectBaseType
GeometricCharacteristicAspectBaseType
FormCharacteristicAspectBaseType
CylindricityCharacteristicAspectType

The ***CharacteristicBaseType*** and other types whose name includes ***Base*** are abstract types and cannot be instantiated. The hierarchy has 46 leaf types that are not abstract and can be instantiated. Most of the leaf types exist because they are included in ASME Y14.5. Several of the types exist only so that users can create user-defined characteristics for which there is a defined unit. These are the ones containing ***Area***, ***Force***, ***Mass***, ***Pressure***, ***Speed***, ***Temperature***, and ***Time***. None of the predefined characteristics uses those units. User-defined characteristics not using any of those units may be defined with either ***UserDefinedAttributeCharacteristicAspectType*** or ***UserDefinedUnitCharacteristicAspectType***.

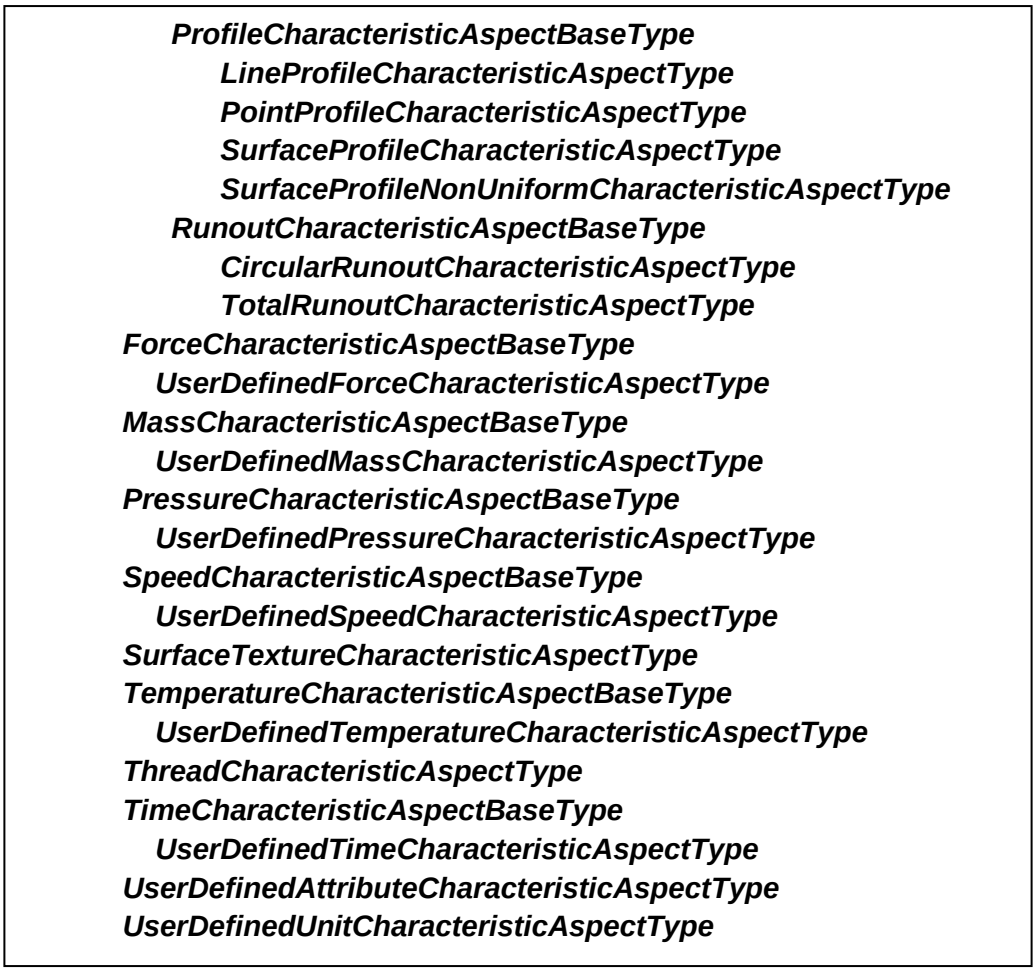

Figure 2 – Characteristic types

6.2.3 DefaultCharacteristicDefinitions

The **DefaultCharacteristicDefinitions** *element* shown in Figure 1 is list of default or “unless otherwise specified” characteristic definitions.

6.2.4 DefaultToleranceDefinitions

The **DefaultToleranceDefinitions** *element* shown in Figure 1 is a list of tolerance definitions with ids. The list may be a mix of **LinearTolerance** and **AngularTolerance** elements. A tolerance in the list can be referenced by id from a characteristic definition that uses a tolerance of the correct type. *Key*/*keyref* checks are included for this.

6.2.5 CharacteristicGroups

The **CharacteristicGroups** *element* shown in Figure 1 is a list of characteristic groups. Each of these is of ***CharacteristicGroupType*** or one of its derived types, ***CharacteristicManufacturingProcessGroupType*** and

CharacteristicSimultaneityGroupType. A characteristic group may be transformed as a whole and have a status as a whole.

6.2.6 Constraint checking for characteristics

Between QIFDocument.xsd and Characteristics.xsd there are 138 (46 x 3) *keyref*s for checking references between the aspects of characteristics. The three sorts of references between characteristics that these check are:

- actual to item
- item to nominal
- nominal to definition

For example, if the **NominalId** *element* of an instance of ***SurfaceProfileCharacteristicItemType*** is populated, its value must be the **id** of a ***SurfaceProfileCharacteristicNominalType***.

The actual-to-item checks are in QIFDocument.xsd. The other two types of checks are in Characteristics.xsd.

6.2.7 ToleranceZones

Characteristics.xsd defines types that describe the shapes of tolerance zones used in the characteristics as shown in Figure 3.

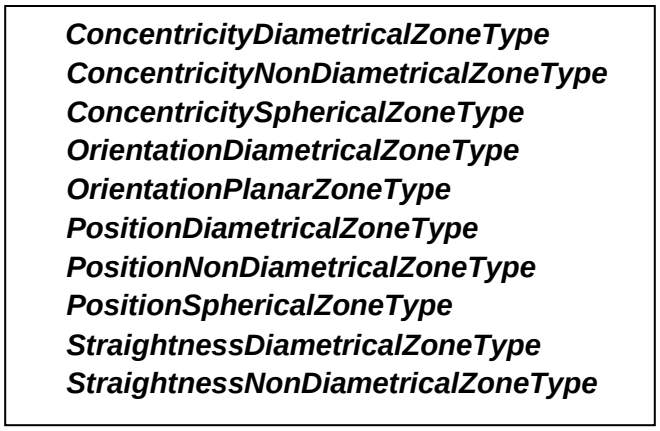

Figure 3 – Tolerance zone types

6.2.8 Substitution groups for characteristics

Characteristics.xsd defines five *substitutionGroup*s. One of these is for **CharacteristicGroup**, and has only two substitutable *element*s. Each of the other four *substitutionGroup*s is for a characteristic aspect and has a head whose type is the base type for the aspect plus 46 members (one for each instantiable characteristic type).

6.3 Expressions.xsd

The Expressions.xsd schema file defines expressions for supporting QIFRules.xsd. They are intended for use when a feature (possibly associated with a characteristic) is being examined for one of three purposes: (1) picking the number of target hit points on the feature for probing, (2) picking the strategy (pattern) for hit points, or (3) picking a substitute feature algorithm for fitting the feature to hit points. The (possibly automated) system that is making the decisions is working in an environment in which levels of sampling rigor exist and a specific level of sampling rigor has been set. The system is expected to have access to information about the feature and the characteristic (if there is one). Sampling rigor is discussed further in Part 6.

The following expression types derived from Boolean expression are defined.

CharacteristicIsType
FeatureIsDatumType
FeatureIsInternalType
SamplingRigorIsType
ShapeClassIsType

The following expression types derived from arithmetic expression are defined.

ArithmeticFeatureParameterType
ArithmeticCharacteristicParameterType
FeatureLengthType
FeatureAreaType

6.4 Features.xsd

6.4.1 Features *element*

Measurement features are generated by analyzing the association between (1) geometric dimensions and tolerances (GD&T) and (2) product design geometric elements as specified in design models or drawings. In the QIF realm and widely accepted in the metrology and manufacturing quality fields, features are defined as the underlying targets to which GD&T is applied. These features are often defined as points, curves, planes, spheres, etc. that can actually be seen on the physical part to be measured. The characteristics defined in Characteristics.xsd apply primarily to features; for example a diameter characteristic might apply to a circle or cylinder, and a perpendicularity characteristic might apply to two planes.

The Features.xsd schema file defines types that describe dimensional metrology shape features. The top level type defined in the file is the ***FeatureAspectsListsType***. This is used in the QIFDocument schema as the type of the **Features** *element*, as shown in Figure 4.

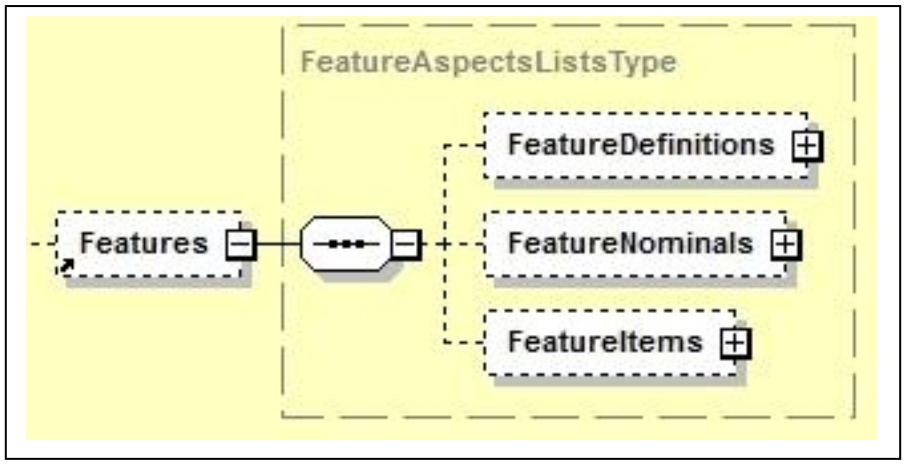

Figure 4 – Features element

6.4.2 Feature types

In all, the file defines 29 instantiable feature types plus base feature types. Features have four aspects: *definition*, *nominal*, *actual*, and *item*. Each of the 29 feature types has a type for each of the four aspects. Lists of three of the aspects are included in the ***FeatureAspectsListsType***. A list of the fourth aspect, *actual*, is included in measurement results. Feature descriptions in natural language are given in the in-line documentation of the feature types. The file also:

- defines methods of constructing features
- contains the *key*/*keyref* pairs that constrain references among the four aspects of features
- defines the *subsitutionGroup*s for feature aspects.

All of the feature types in DMIS 5.2 have an equivalent in QIF. QIF has four others that do not have an equivalent in the DMIS 5.2 standard. A comparison of feature definitions in QIF and DMIS is shown in Figure 5. QIF features are described using Cartesian coordinates. Notes may be attached to any of the feature aspects.

QIF Feature	Equivalent DMIS Feature
Arc	ARC (format 1)
Circle	CIRCLE
Composite (*base type*)	
Compound	COMPOUND
Pattern	PATERN
ProfileGroup	*no equivalent*
RunoutGroup	*no equivalent*
Cone	CONE
ConicalSegment	CONRADSEGMNT
Cuboid	RCTNGL
Cylinder	CYLNDR
CylindricalSegment	CYLRADSEGMNT
EdgePoint	EDGEPT
Ellipse	ELLIPS
ElongatedCylinder	ELONGCYL
ExtrudedCrossSection	*no equivalent*
Generic	GEOM, OBJECT
Line	LINE
OppositeLines	CPARLN
OppositePlanes	PARPLN, SYMPLN
Plane	PLANE
Point	POINT
PointDefinedCurve	GCURVE
PointDefinedSurface	GSURF
Sphere	SPHERE
SphericalSegment	SPHRADSEGMNT
SurfaceOfRevolution	REVSURF
Threaded	*no equivalent*
ToroidalSegment	TORRADSEGMNT
Torus	TORUS

Figure 5 – Comparison of feature definitions in QIF and DMIS

The hierarchy of feature type derivations is illustrated in Figure 6, using the example of the circle feature family. The derivations are implemented using XSDL *extension*s. As shown in Figure 6, the most basic feature type is ***FeatureBaseType***. From that, a base type is derived for each of the four aspects (definition, nominal, actual, and item), and from each of those, 29 specific types are derived, only one of which (circle) is shown in the figure.

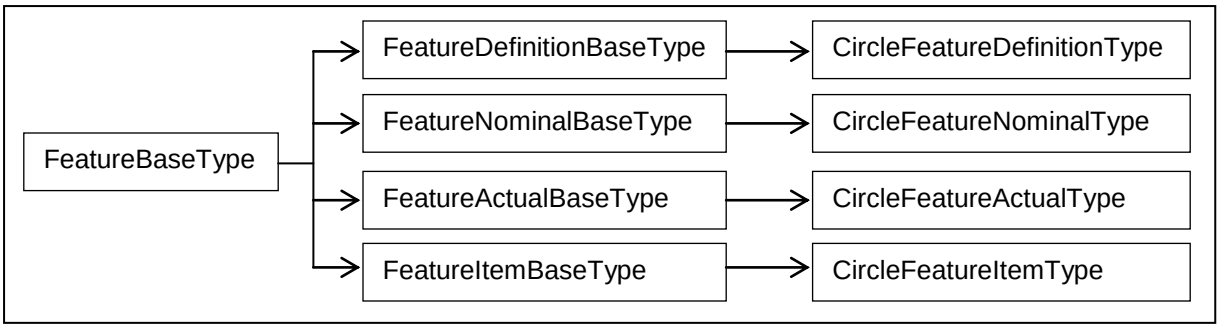

Figure 6 – Hierarchy of type definitions for feature data object types (example)

6.4.3 Constraint checking for features

Between QIFDocument.xsd and Characteristics.xsd there are 87 (29 x 3) *keyref*s for checking references between the aspects of features. The three sorts of references between features that these check are:

- actual to item
- item to nominal
- nominal to definition

For example, if the **FeatureNominalId** *element* of an instance of ***CircleFeatureActualType*** is populated, its value must be the **id** of a ***CircleFeatureNominalType***.

The actual to item checks are in QIFDocument.xsd. The other two types of checks are in Features.xsd.

6.4.4 Feature construction methods

As described in section 6.19.4 of Part 1 of this standard, the Features.xsd schema file provides construction methods for features. These are defined for all types of features except the generic feature and the composite features. There are several stereotypical construction methods that may be applied to the 24 feature types that may be constructed:

- copy – all
- transform – all
- cast – all except PointDefinedCurve and PointDefinedSurface
- best fit – all except Point and EdgePoint
- recompensated – all except Point and EdgePoint
- from scan – Arc, Circle, Cone, Cylinder, EdgePoint, Ellipse, Line, OppositeLines, OppositePlanes, PointDefinedCurve, Point, Sphere, Torus
- extract – Arc, Line, Plane, PointDefinedCurve, PointDefinedSurface
- intersection – Circle, Ellipse, Line, Point
- projection – Arc, Circle, Ellipse, Line, OppositeLines, Point
- tangent through – Circle, Line, Plane

In addition to these generally applicable construction methods, several other construction methods are applicable to one or a few feature types.

- Circle – FromCone, Tangent
- Line – Midline, Parallel, Perpendicular

- Plane – Midplane, Offset, Parallel, Perpendicular
- Point – FromCone, CenterOfGravity, Pierce, Midpoint, MovePoint, MovePointVector, MovePointAxis, Extreme
- Thread – FromCylinder

6.4.5 Substitution groups for features

Finally, the Features.xsd schema file defines four *substitutionGroup*s (one for each feature aspect), each of which has a head that is the base type for the aspect, plus 29 members (one for each instantiable feature type).

6.5 GenericExpressions.xsd

The GenericExpressions.xsd schema file defines Boolean expression types and simple arithmetic expression types. Arithmetic expression types defined in the file that represent arithmetic operators are: ***NegateType***, ***PlusType***, ***MinusType***, ***TimesType***, and ***DividedByType***. ***ArithmeticConstantType*** is also an arithmetic expression type. The common comparisons between arithmetic expressions that return a Boolean result are defined: ***ArithmeticEqualType***, ***GreaterThanType***, ***GreaterOrEqualType***, ***LessThanType***, ***LessOrEqualType***. The Boolean expression types ***BooleanEqualType***, ***ConstantIsType***, ***NotType***, ***AndType***, and ***OrType*** are defined. These expressions are defined for use in QIFExpressions.xsd and QIFRules.xsd. They are currently not used elsewhere in the QIF model.

All expressions other than ***ArithmeticConstantType***, ***ConstantIsType***, ***NegateType***, and ***NotType*** are binary.

All expressions are modeled the same way as everything else is modeled, not in condensed symbolic notation. This makes instances of the expressions verbose but very easy to process. No separate expression parser is needed.

6.6 Geometry.xsd

The Geometry.xsd schema file defines the geometry types shown in Figure **7** and the geometry set types shown in Figure 8. Subhierarchies of the derivation hierarchy shown in Figure **7** are also shown in Part 3. The role of the geometry types in product definition is covered thoroughly in Part 3 and is not described here.

The Geometry.xsd file also defines a mathematical core type for each of the instantiable (leaf) types in Figure 7 and two enumerations.

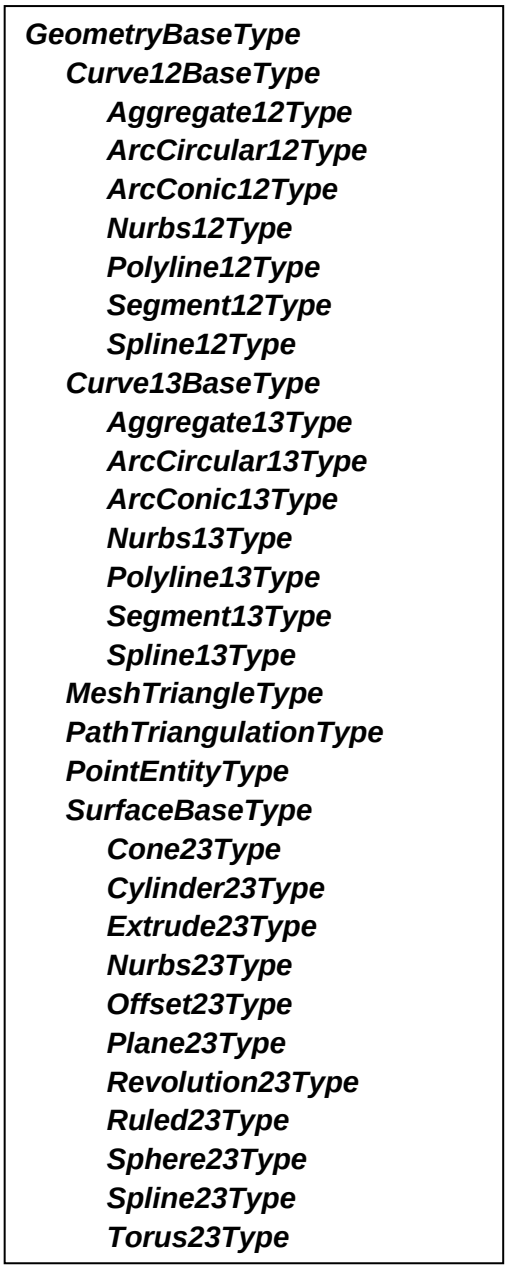

Figure 7 **– Individual geometry types**

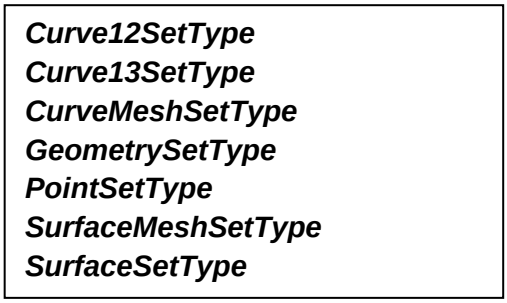

Figure 8 – Geometry set types

The ***GeometrySetType*** in Figure 8 consists of elements of the other six types.

6.7 IntermediatesPMI.xsd

The IntermediatesPMI.xsd schema file defines 136 miscellaneous types used in other schema files. Many of these types either define ids or use ids. Twenty-seven of them are enumerations.

The hierarchy of substitute feature algorithm types shown in Figure 9 is included.

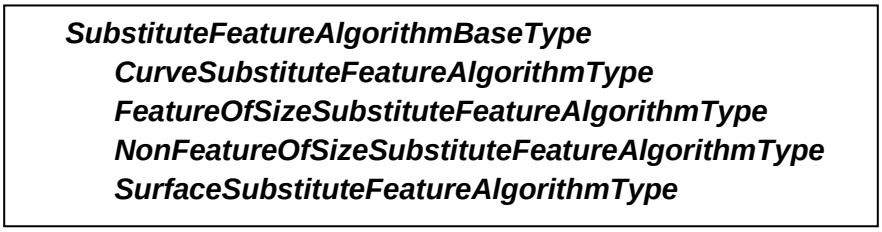

Figure 9 – Substitute feature algorithms

The hierarchy of datum target types shown in Figure 10 is included along with a *substitutionGroup* for them.

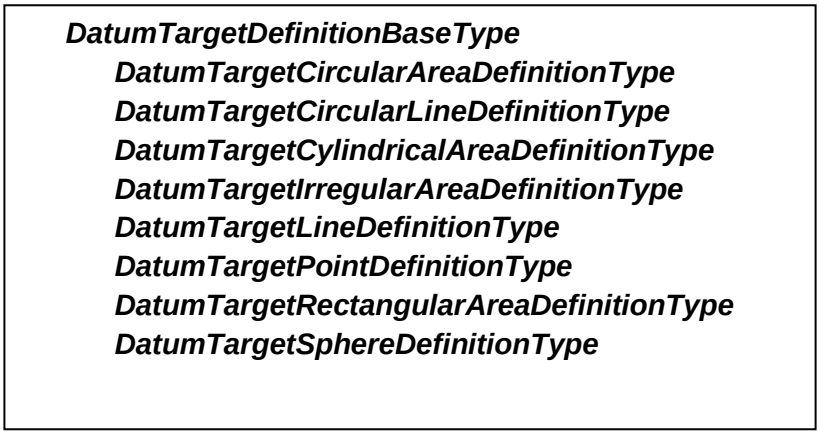

Figure 10 – Datum targets

The hierarchy of alignment operation types shown in Figure 11 is included along with a *substitutionGroup* for them.

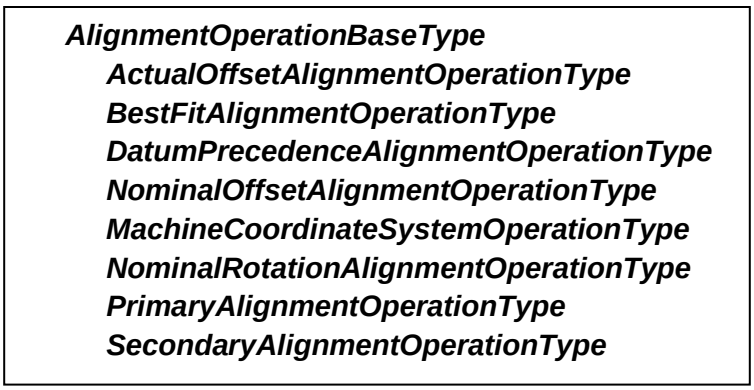

Figure 11 – Alignment operations

The types dealing with datums and datum reference frames shown in Figure 12 are defined.

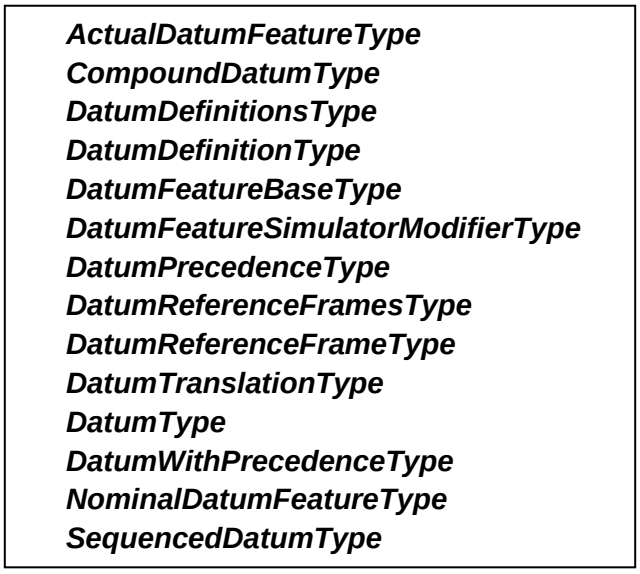

Figure 12 – Datums and datum reference frames

Other types are defined dealing, among other things, with the following: events, notes, threads, coordinate systems, transforms, and points that are to be measured or were measured.

6.8 Primitives.xsd

The Primitives.xsd schema file defines 70 miscellaneous types used in other schema files. Types in Primitives.xsd might be used in CAD as well as in quality measurement. This includes, for example: ***PointType***, ***UnitVectorType***, and ***TransformMatrixType***. Types in this file do not have an **id** *attribute*.

Fourteen of the types have an *attribute* named N and represent collections such as arrays and lists. In these types, N is the number of items.

The definitions of the AttributeBaseType and seven derived types are included along with the definitions of seven global elements (one of each type) in a substitution group. These are made available to users for including data in an instance file that is not modeled elsewhere in QIF. The seven elements are:

- AttributeBool – an xs:boolean
- AttributeStr – an xs:string
- AttributeQPId – a QPId for identifying the containing element
- AttributeI1 – a single xs:integer
- Attribute I2 – two xs:integers
- AttributeD3 – three xs:doubles
- AttributeUser – binary data or any user-defined XML data.

An Attributes element including any number of these elements may be inserted in an instance file at many places. All the Attributes elements are optional.

6.9 PrimitivesPD.xsd

The PrimitivesPD.xsd schema file defines 2 miscellaneous *simpleTypes* and 7 *complexTypes* that are used in CAD product definition but not in PMI.

6.10 PrimitivesPMI.xsd

The PrimitivesPMI.xsd schema file defines miscellaneous types that do not have ids and are not used in geometry or topology but are used elsewhere. This includes about a dozen enumerations and two dozen *complexType*s. Four of the *complexType*s are scale types. Another four are used for datum targets. Nine *complexType*s offer a choice between an enumeration value and a user-defined value. These are shown in Figure 13.

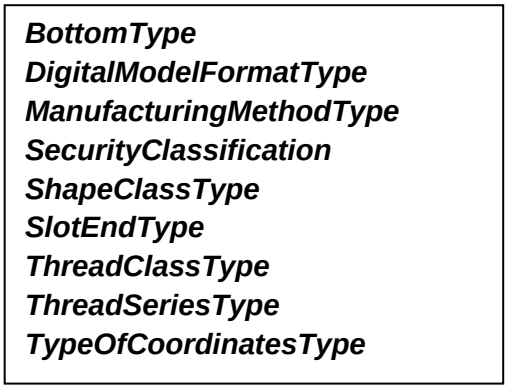

Figure 13 – Types with enumeration or user definition

6.11 Statistics.xsd

The Statistics.xsd schema file supports the QIFStatistics application. It defines type hierarchies and miscellaneous other types used in quality statistics. This includes 134 *complexType*s and 15 *simpleType*s.

6.11.1 Characteristic Statistics Evaluation Types

The file defines a hierarchy of characteristic statistics evaluation types as shown in Figure 14. This is very similar to the characteristics hierarchy but omits some of the intermediate types of that hierarchy. Statistics may be reported for each one of these whose name does not include ***Base***, thus offering various aggregated statistics in addition to statistics for all of the leaf node types.

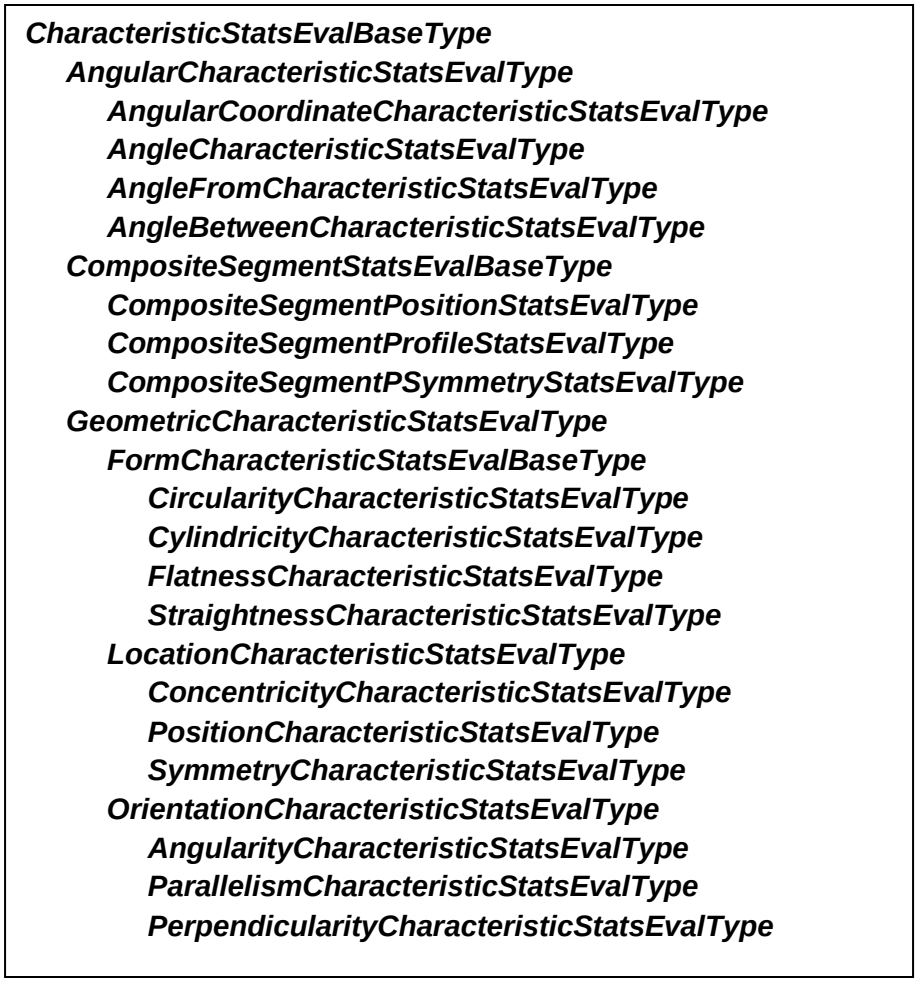

Figure 14 – CharacteristicStatsEval types (continued on next page)

Figure 14 (continued from previous page)

6.11.2 Criterion types

The file defines three Criterion types, one of which has a derived type for each of ten unit types, as shown in Figure 15. A Criterion type defines a numerical limit outside of which an issue (e.g., a process control issue) will exist.

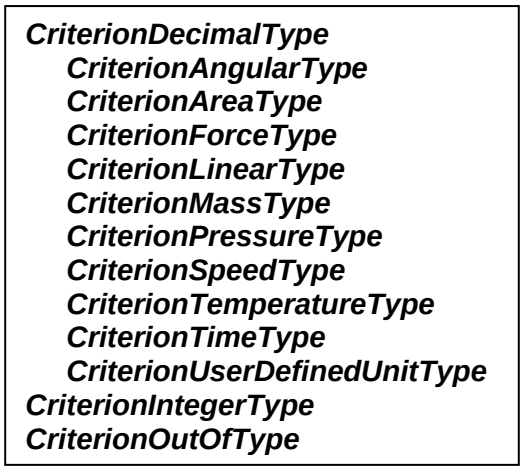

Figure 15 – Criterion types

6.11.3 Other items

The Statistics.xsd schema file defines a dozen enumeration types for statistics. For three of these, a list type is defined.

6.12 Topology.xsd

The topology.xsd schema file defines the individual topology types shown in Figure 16 and the topology set types shown in Figure 17 plus supporting enumeration types. The role of the topology types in product definition is covered thoroughly in Part 3 and is not described here.

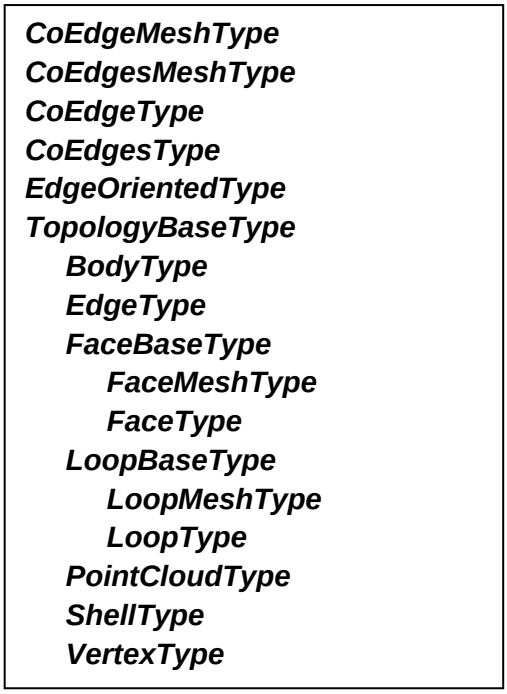

Figure 16 – Individual topology types

Figure 17 – Topology set types

The ***TopologySetType*** in Figure 17 consists of elements of the other seven types.

6.13 Traceability.xsd

The purpose of traceability information is to make it possible to associate QIF data files with the product inspected and the resources used to design it, manufacture it, program its inspection, and inspect it. The Traceability.xsd schema file defines types that describe the circumstances of a planned or completed quality measurement.

6.13.1 *InspectionTraceabilityType*

The ***InspectionTraceabilityType*** contains information applicable to an entire set of inspection results. An instance of the ***InspectionTraceabilityType*** is used (optionally) as the value of the **InspectionTraceability** *element* in both the ***MeasurementsResultsType*** (multiple results sets) and the ***MeasurementResultsType*** (single results set) of the QIFResults schema. It is also used as the value of the **InspectionTraceability** *element* in the ***StatisticalStudyResultsBaseType*** (and, hence, all derived types) in the QIFStatistics schema. All 23 *element*s of ***InspectionTraceabilityType*** are optional. The *element*s are shown in Figure 18.

Figure 18 – Elements of InspectionTraceabilityType

6.13.2 *PreInspectionTraceabilityType*

The ***PreInspectionTraceabilityType*** contains information similar to that of the ***InspectionTraceabilityType***, except that is it tailored for use when inspection has not yet occurred. ***PreInspectionTraceabilityType*** is the type of the **PreInspectionTraceability** *element* of the ***QIFDocumentType*** in QIFDocument.xsd and the ***StatisticalStudyPlanBaseType*** (and, hence, all its derived types) in QIFStatistics.xsd. Most *element*s of ***PreInspectionTraceabilityType*** are optional. The **AsmPathIds** indicate what is to be inspected. All elements are optional, except for the **FormalStandard**, which is required. The *element*s are shown in Figure 19.

Figure 19 – Elements of *PreInspectionTraceabilityType*

6.13.3 *ProductTraceabilityType*

The ***ProductTraceabilityType*** defines traceability information for a component (i.e., part or assembly). It is used (optionally) as the value of the **Traceability** *element* of the ***ComponentType*** in Product.xsd. The *element*s of ***ProductTraceabilityType*** are all optional. They are shown in Figure 20.

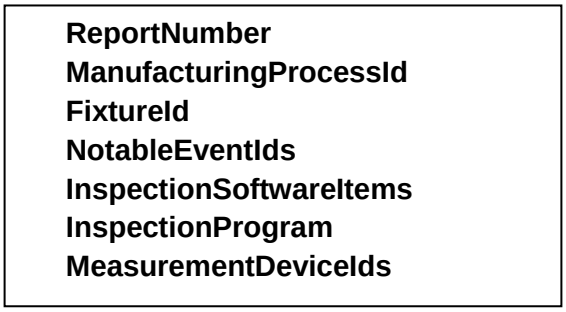

Figure 20 – Elements of *ProductTraceabilityType*

6.13.4 *ActualProductTraceabilityType*

The ***ActualProductTraceabilityType*** defines traceability information for an actual component. It is used (optionally) as the **Traceability** *element* of the ***ActualComponentType*** in QIFResults.xsd. Its 15 *element*s are all optional. They are shown in Figure 21.

Figure 21 – Elements of *ActualProductTraceabilityType*

6.13.5 *ManufacturingProcessTraceabilityType*

The ***ManufacturingProcessTraceabilityType*** defines traceability information for a manufacturing process. It is used in the **ManufacturingProcessTraceabilities** *element* of the ***QIFDocumentType.*** Once a **ManufacturingProcessTraceability** is listed there, it may be referenced using the **ManufacturingProcessId** *element* found at several places in the QIF model. The *element*s of ***ManufacturingProcessTraceabilityType*** are all optional. They are shown in Figure 22. There is also a required **id** *attribute* which may be used for referencing.

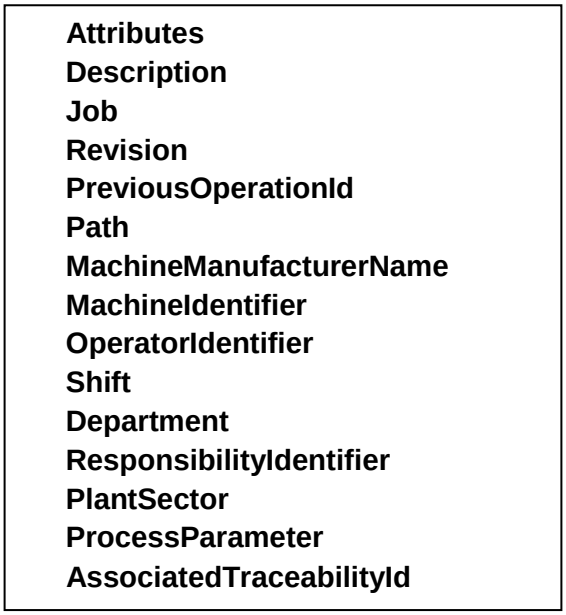

Figure 22 – Elements of *ManufacturingProcessTraceabilityType*

Significant supporting types defined in Traceability.xsd include ***EnvironmentType*****,** ***InspectionProgramType*****,** and ***PartialInspectionType*****.**

6.14 Units.xsd

See also section 6.15, QIF Handling of Units, of Part 1 of the QIF standard regarding units.

6.14.1 FileUnits

The top level type defined in the file is the ***FileUnitsType***. This is used in the QIFDocument.xsd schema as the type of the **FileUnits** *element*, as shown in Figure 23. The **FileUnits** *element* gives explicit primary units for the file and optionally gives other units and user defined units that may be used in the file. The **FileUnits** *element* is optional, but it is strongly recommended that it be used. In the absence of a **FileUnits** *element*, the default units are SI units.

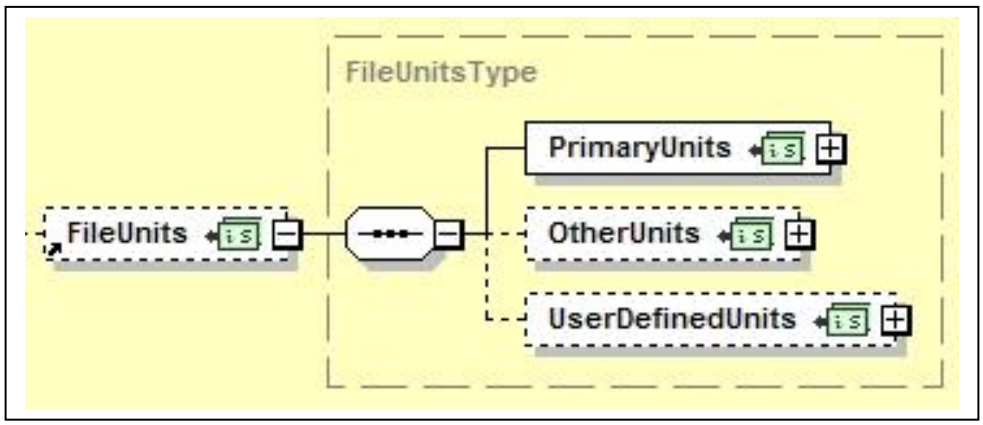

Figure 23 – FileUnits element

The Units.xsd schema file defines units for values of angle, area, force, length, mass, pressure, speed, temperature, and time. The length unit is called *linear* unit to avoid confusion with the length characteristic. The file also defines the following corresponding value types that have units. The default unit is shown following each value type.

AngleValueType radian
AreaValueType square meter
ForceValueType newton
LinearValueType meter
MassValueType kilogram
PressureValueType pascal
SpeedValueType meter per second
TemperatureValueType kelvin
TimeValueType second

A user defined characteristic type is defined in Characteristics.xsd for each of the defined unit types, and the **TargetValue** in each of those is the corresponding value type.

The Units.xsd file uses optional *attribute*s extensively in order that numbers appearing in instance files do not require a lot of accompanying markup.

The derivation hierarchy of values with units is shown in Figure 24. The items shown after the type names are the *attribute*s that are added in the type. *Attribute*s that may be used with a value with any type of unit in an instance file include the name of the unit, the number of decimal places in the value, and the number of significant figures in the value. If a value represents an actual value, it may also have *attribute*s giving combined uncertainty and mean error. All of these *attribute*s are optional.

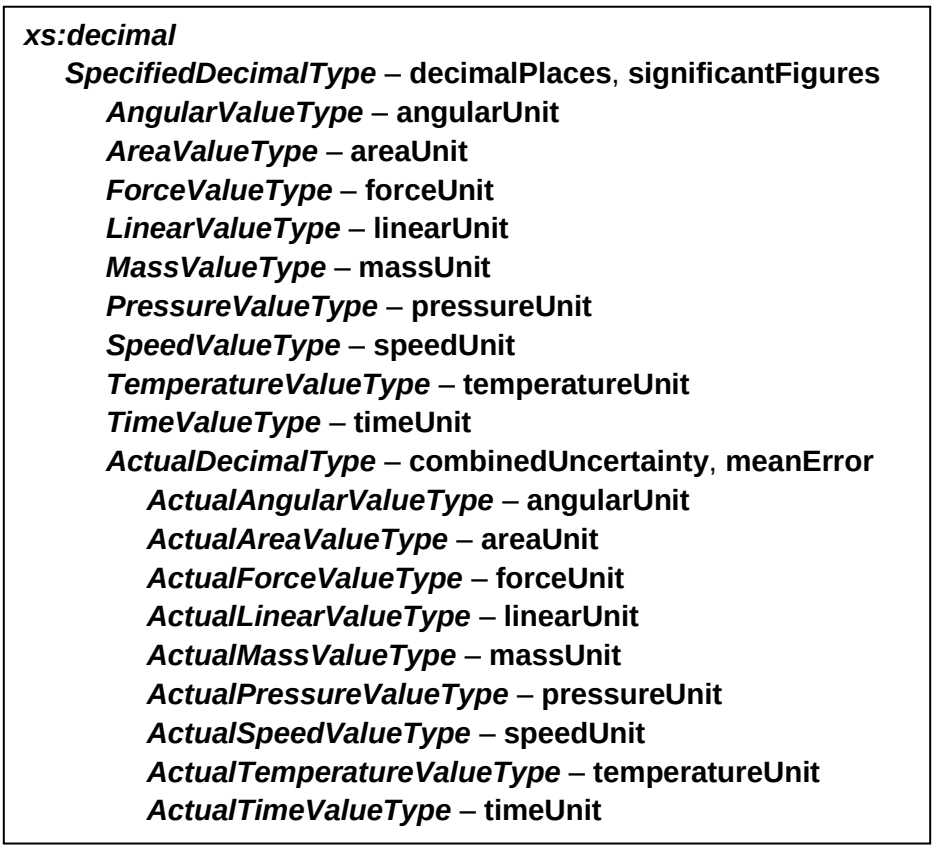

Figure 24 – Derivation hierarchy of values with units

The Units.xsd schema file also includes a ***UserDefinedUnitType*** and a ***UserDefinedUnitValueType*** that is a number with a required *attribute* that is the name of a ***UserDefinedUnitType***. The number may also have the optional *attribute*s mentioned above. The ***UserDefinedUnitValueType*** may be used in an instance file to define a user defined unit characteristic.

6.14.2 Conversions

The ***UnitConversionType*** has *element*s **Factor** and **Offset**. These are the parameters for a conversion from non-SI units to SI units. To convert a non-SI unit value X to an SI unit value S, use the equation: S = ((X plus **Offset**) times **Factor**). Figure 25 shows values of **Factor** and **Offset** for common units. More precise values of some **Factor**s may be needed in some applications.

```
Angle conversion to radians
  degree            Factor=0.017453293  Offset=0
Area conversion to square meters
  square inch       Factor=0.00064516   Offset=0
  square foot       Factor=0.09290304   Offset=0
  square millimeter Factor=0.000001     Offset=0
Force conversion to newtons
  kilogram          Factor=9.80665      Offset=0
  ounce             Factor=0.2780139    Offset=0
  pound             Factor=4.448222     Offset=0
Length conversion to meters
  foot              Factor=0.3048       Offset=0
  inch              Factor=0.0254       Offset=0
  millimeter        Factor=0.001        Offset=0
Mass conversion to kilograms
  gram              Factor=0.001        Offset=0
  ounce             Factor=0.02834952   Offset=0
  pound             Factor=0.4535924    Offset=0
Pressure conversion to pascals
  kilopascal        Factor=1000.0       Offset=0
  psi               Factor=6894.757     Offset=0
Speed conversion to meters per second
  feetPerSecond     Factor=0.3048       Offset=0
  inchesPerSecond   Factor=0.0254       Offset=0
  mmPerSecond       Factor=0.001        Offset=0
Temperature Conversion to kelvin
  Fahrenheit        Factor=0.555555556  Offset=459.67
  Celsius           Factor=1.0          Offset=273.15
  Rankine           Factor=0.555555556  Offset=0
Time conversion to seconds
  hour              Factor=3600.0       Offset=0
  minute            Factor=  60.0       Offset=0
```

Figure 25 – Conversion of units

6.14.3 FileUnitsExample

Figure 26 shows a snippet of an instance file containing a **FileUnits** element. The **SIUnitName** and the **UnitConversion** *element*s are optional. In the snippet, the **AngleUnit** uses them but the **LinearUnit** does not. The snippet shows one entry in the **OtherUnits** and one entry in the **UserDefinedUnits**.


```
<FileUnits>
  <PrimaryUnits>
    <AngularUnit>
      <SIUnitName>radian</SIUnitName>
      <UnitName>degree</UnitName>
      <UnitConversion>
        <Factor>0.017453293</Factor>
      </UnitConversion>
    </AngularUnit>
    <LinearUnit>
      <UnitName>inch</UnitName>
    </LinearUnit>
    <TemperatureUnit>
      <SIUnitName>kelvin</SIUnitName>
      <UnitName>Fahrenheit</UnitName>
      <UnitConversion>
        <Factor>0.555555556</Factor>
        <Offset>459.67</Offset>
      </UnitConversion>
    </TemperatureUnit>
  </PrimaryUnits>
  <OtherUnits>
    <LinearUnit>
      <UnitName>mm</UnitName>
    </LinearUnit>
  </OtherUnits>
  <UserDefinedUnits>
    <UserDefinedUnit>
      <WhatIsMeasured>electrical resistance</WhatIsMeasured>
      <UnitName>ohm</UnitName>
    </UserDefinedUnit>
  </UserDefinedUnits>
</FileUnits>
```

Figure 26 – FileUnits snippet

6.14.4 Instance File Example Using Units

Three snippets from an instance file are shown in Figure 27. Assuming that the FileUnits shown in Figure 26 is in the instance file, the snippets are all valid.

The first cylinder has a diameter of 19 mm; the linearUnit mm is given explicitly. That is valid because mm is given as one of the OtherUnits. The decimalPlaces value of 1 is valid even though no decimal places appear in “19”.

The second cylinder has a diameter of 2.53 inches. The diameter has no explicit linearUnit, but the linearUnit for the diameter is implicitly inch because inch was given as the LinearUnit in the PrimaryUnits of the FileUnits.

The nominal angularity characteristic has an angle of 10.00145 degrees, although only the first four figures are significant. The angle has no explicit angularUnit, but the angularUnit for the

Angle is implicitly degree because degree was given as the AngularUnit in the PrimaryUnits of the FileUnits.

```
<CylinderFeatureDefinition id="62">
  <Diameter linearUnit="mm" decimalPlaces="1">19</Diameter>
</CylinderFeatureDefinition>
  ...
<CircleFeatureDefinition id="72">
  <Diameter>2.53</Diameter>
</CircleFeatureDefinition>
...
<AngularityCharacteristicNominal id="45">
  <CharacteristicDefinitionId>42</CharacteristicDefinitionId>
  <Angle significantFigures=”4”>10.00145</Angle>
</AngularityCharacteristicNominal>
```

Figure 27 – Instance file snippets using units

6.15 Visualization.xsd

The Visualization.xsd schema file defines types needed for visualizing products, including PMI items.

Most of the information required for PMI visualization is in the PMI item being displayed. The way it should be visualized is prescribed by ASME Y14.5 and associated standards (e.g. ASME Y14.41). However the placement of the PMI item on a 2D drawing or in a 3D model based design is controlled by the user, as are the lines connecting the visualized PMI with the visualized feature. Additional text may also accompany a PMI item and may use various fonts. These additional items are what the Visualization.xsd schema file defines. Any QIF application providing PMI visualization should be capable of displaying PMI from this visualization information.

The top level type for displaying PMI items is the ***VisualizationSetType***, which is the type of the **VisualizationSet** *element* of the ***ProductType*** as shown in Figure 28. The ***VisualizationSetType*** has (1) a **FontSet** *element* that is a list of **Font** *element*s and (2) a **PMIDisplaySet** *element* that is a list of **PMIDisplay** *element*s.

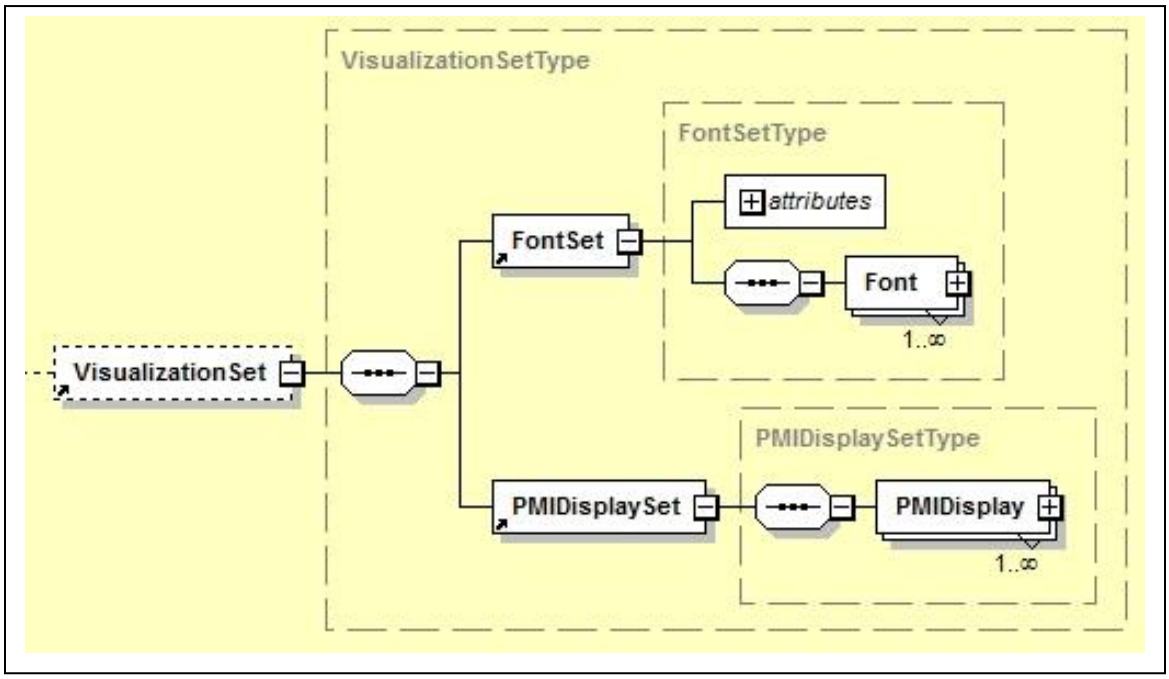

Figure 28 – VisualizationSet

The **Font** *element*s are of ***FontType***. A ***FontType*** is described by **Name** and **Size** *element*s. It also has *attribute*s indicating whether the font is bold and/or italic and/or underlined. Another *attribute* gives an index number for the font.

The **PMIDisplay** *element*s are of ***PMIDisplayType***. The *element*s of the ***PMIDisplayType*** include a required **Reference** *element* that is the id of a PMI item to be displayed and the following optional *element*s:

- **Attributes** (user data)
- **Color**
- **Plane**
- **Texts**
- **Leader** (or one of 5 substitutes that are specialized types of leader)
- **WitnessLines**
- **Frames**
- **Balloon**

For visualization of products there are two instantiable types of view, ***CameraType***, and ***SavedViewType***. Both of these include (1) the usual parameters for a computer graphics camera (2) parameters for a view, including color, transparency, etc. and (3) a set of user definable attributes that have data types but no built-in meaning. The ***SavedViewType*** has, in addition, five sets of ids of QIF items, including, for example, the ids of characteristics that should be shown and the ids of characteristics that should not be shown.

7 Data dictionary for QIF Library Data Model

The QIF library consists of fifteen XML schema files that describe the common elements of the QIF information model. The files are normative material, and all QIF XML schema files are bundled together in a compressed folder file named “QIF_2.0_XMLSchemaFiles.zip”, which is available at www.qifstandards.org.

Data dictionaries describe the Library information model in human-readable format, and are also normative material. Due to their large size, the fifteen library data dictionaries are contained in separate files:

- “QIF_2.0_DataDictionary_Auxiliary.pdf”
- “QIF_2.0_DataDictionary_Characteristics.pdf”
- “QIF_2.0_DataDictionary_Expressions.pdf”
- “QIF_2.0_DataDictionary_Features.pdf”
- “QIF_2.0_DataDictionary_GenericExpressions.pdf”
- “QIF_2.0_DataDictionary_Geometry.pdf”
- “QIF_2.0_DataDictionary_IntermediatesPMI.pdf”
- “QIF_2.0_DataDictionary_Primitives.pdf”
- “QIF_2.0_DataDictionary_PrimitivesPD.pdf”
- “QIF_2.0_DataDictionary_PrimitivesPMI.pdf”
- “QIF_2.0_DataDictionary_Statistics.pdf”
- “QIF_2.0_DataDictionary_Topology.pdf”
- “QIF_2.0_DataDictionary_Traceability.pdf”
- “QIF_2.0_DataDictionary_Units.pdf”
- “QIF_2.0_DataDictionary_Visualization.pdf”.

The data dictionary files are normative material, and are all bundled together in a compressed folder file named “QIF_2.0_LibraryDataDictionaries.zip”, which is available at www.qifstandards.org.

Annex A - Location of QIF 2.0 XML Library schema files

The QIF information model is expressed in XML schema definition language. All QIF XML schema files are normative and are bundled into a single compressed folder file called “QIF_2.0_XMLSchemaFiles.zip”, which is available for download at www.qifstandards.org.

Annex B - Graphical conventions used in the data dictionary

This section describes the graphical conventions used in the QIF data dictionaries. The data dictionaries describe the structure of the information models and the manufacturing quality semantics of the data types.

The rules of encoding QIF instance files are primarily defined in the XML schema files, but the data dictionaries express many of the same requirements via the pictures and table entries.

Data type definitions are indicated by a box with beveled corners on the left side, as in Figure B.1.

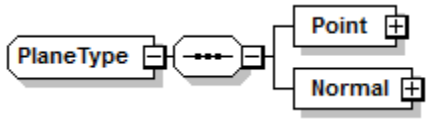

Figure B.1 – Notation for a type definition, *PlaneType*.

Rectangular boxes indicate data *element*s. A solid rectangle indicates a required *element*, whereas a dotted rectangle indicates an optional *element*. If an object is not designated optional, then it is required by default. Small boxes on the right hand end of *element* boxes, containing either "-" or "+" are used to indicate that one of the following conditions exist:

- A (“+”) indicates that the additional structures or *element*s below this node have been hidden in this diagram.
- A (“-“) indicates that additional structures or *element*s below this node exist and are visible on the diagram.

The absence of any box at the right hand end of an *element* box indicates that the type of the *element* is a primitive type without any substructure, e.g., *xs:decimal*. In this case, there will also be three bars in the upper left corner of the *element* box (examples appear in Figure B.3). The beveled box with 3 dots on a line represents the XML *sequence* operator. It indicates that the object to the left is composed of all of the *element*s to the right, in top to bottom order.

Type definitions can be reused to generate data *element*s, as shown by a yellow box in dotted lines, with the name of the type definition at the top. Figure B.2 shows that **ZonePlane** is an *element* of type ***PlaneType***.

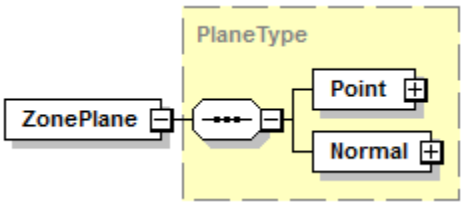

Figure B.2 – Reuse of the type definition *PlaneType* to generate element ZonePlane.

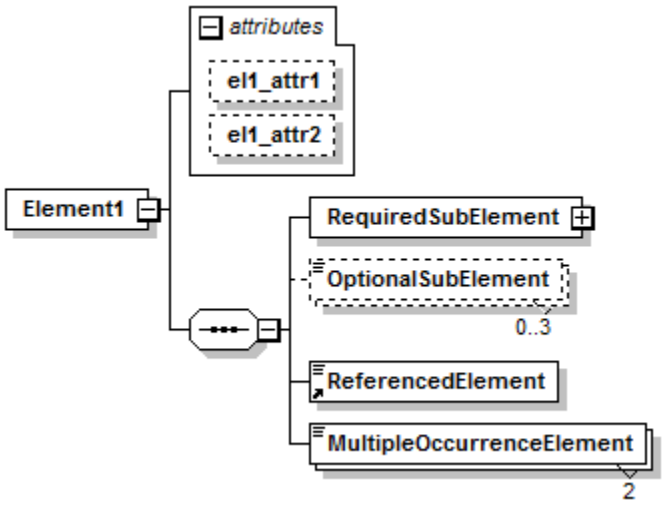

Figure B.3 – Notation for *elements*, sub-*elements*, and *attributes*.

Figure B.3 contains examples of numerous information modeling notations. *Element* definitions in XML schema files can be reused by "reference", indicated by an arrow in the lower left corner of the **ReferencedElement** box. *Element*s may appear in an XML instance document more than once. Figure B.3 shows the **OptionalSubElement** notated with two numerals separated by an ellipsis, e.g., "0..3", that indicates the number of occurrences as an inclusive range. The **OptionalSubElement** may occur zero, 1, 2, or 3 times as sub-elements of **Element1**. Where there is a single cardinality numeral, the *element* must occur exactly that number of times in the instance file. For example, the *element* **MultipleOccurrenceElement** must occur exactly two times as sub-*elements* of **Element1**. Information items can be instantiated in XML as *elements* or *attributes*. An *element*'s *attributes* are shown in the data dictionaries as solid-lined boxes that are explicitly labeled *attributes*, as shown at the top of the diagram.

Figure B.4 shows an example *element* definition where exactly one of the three sub-*element* choices must be given. The beveled box with three square dots and a "switch" line () indicate the XML *choice* structure. When **Element2** is instantiated in an XML instance file, it must have exactly one sub-*element* chosen among the three sub-*elements* shown.

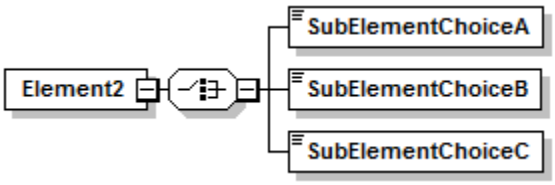

Figure B.4 – The *choice of* notation.

The data dictionaries are grouped by XML schema file. It is characteristic of QIF definitions to use types declared in other XML schema files. The sharing of definitions specified in other files is indicated by the XML schema file directive *include*.

Bibliography

[1] SAE AS9102a (2004-01), *Aerospace First Article Inspection Requirement*

[2] Walmsley, Priscilla., 2002. *Definitive XML Schema*. Prentice Hall, Upper Saddle River, NJ, USA

[3] ASME B89.7.2 – 1999, *Dimensional Measurement Planning*

