

IPC-2581B-WAM1

2016 - December

Generic Requirements for Printed Board Assembly Products Manufacturing Description Data and Transfer Methodology

Supersedes IPC-2581B

September 2013

An international standard developed by IPC

Association Connecting Electronics Industries



The Principles of Standardization

In May 1995 the IPC's Technical Activities Executive Committee (TAEC) adopted Principles of Standardization as a guiding principle of IPC's standardization efforts.

Standards Should:

- Show relationship to Design for Manufacturability (DFM) and Design for the Environment (DFE)
- Minimize time to market
- Contain simple (simplified) language
- Just include spec information
- Focus on end product performance
- Include a feedback system on use and problems for future improvement

Standards Should Not:

- Inhibit innovation
- Increase time-to-market
- Keep people out
- Increase cycle time
- Tell you how to make something
- Contain anything that cannot be defended with data

Notice

IPC Standards and Publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Existence of such Standards and Publications shall not in any respect preclude any member or nonmember of IPC from manufacturing or selling products not conforming to such Standards and Publication, nor shall the existence of such Standards and Publications preclude their voluntary use by those other than IPC members, whether the standard is to be used either domestically or internationally.

Recommended Standards and Publications are adopted by IPC without regard to whether their adoption may involve patents on articles, materials, or processes. By such action, IPC does not assume any liability to any patent owner, nor do they assume any obligation whatever to parties adopting the Recommended Standard or Publication. Users are also wholly responsible for protecting themselves against all claims of liabilities for patent infringement.

IPC Position Statement on Specification Revision Change

It is the position of IPC's Technical Activities Executive Committee that the use and implementation of IPC publications is voluntary and is part of a relationship entered into by customer and supplier. When an IPC publication is updated and a new revision is published, it is the opinion of the TAEC that the use of the new revision as part of an existing relationship is not automatic unless required by the contract. The TAEC recommends the use of the latest revision. Adopted October 6, 1998

Why is there a charge for this document?

Your purchase of this document contributes to the ongoing development of new and updated industry standards and publications. Standards allow manufacturers, customers, and suppliers to understand one another better. Standards allow manufacturers greater efficiencies when they can set up their processes to meet industry standards, allowing them to offer their customers lower costs.

IPC spends hundreds of thousands of dollars annually to support IPC's volunteers in the standards and publications development process. There are many rounds of drafts sent out for review and the committees spend hundreds of hours in review and development. IPC's staff attends and participates in committee activities, typesets and circulates document drafts, and follows all necessary procedures to qualify for ANSI approval.

IPC's membership dues have been kept low to allow as many companies as possible to participate. Therefore, the standards and publications revenue is necessary to complement dues revenue. The price schedule offers a 50% discount to IPC members. If your company buys IPC standards and publications, why not take advantage of this and the many other benefits of IPC membership as well? For more information on membership in IPC, please visit www.ipc.org or call 847/597-2872.

Thank you for your continued support.



IPC-2581B-WAM1

Generic Requirements for Printed Board Assembly Products Manufacturing Description Data and Transfer Methodology

Developed by Product Data Description (Laminar View) Subcommittee
(2-16) of the Electronic Product Data Description Committee (2-10) of
IPC

Supersedes:

IPC-2581B - September 2013
IPC-2581A - May 2012
IPC-2581 with Amendment 1 -
May 2007
IPC-2581 - March 2004

Contact:

IPC
3000 Lakeside Drive, Suite 105N
Bannockburn, Illinois
60015-1249
Tel 847 615.7100
Fax 847 615.7105

This Page Intentionally Left Blank

Acknowledgment

Any document involving a complex technology draws material from a vast number of sources across many continents. While the principal members of the Product Data Description (Laminar View) Subcommittee (2-16) of the Electronic Product Data Description Committee (2-10) are shown below, it is not possible to include all of those who assisted in the evolution of this standard. To each of them, the members of the IPC extend their gratitude.

Electronic Product Data Description Committee	Product Data Description (Laminar View) Subcommittee	Technical Liaison of the IPC Board of Directors
Chair Gary M. Ferrari FTG Circuits	Chair Karen McConnell Northrop Grumman Corporation	Bob Neves Microtek (Changzhou) Laboratories

Product Data Description (Laminar View) Subcommittee

Edward Acheson, Cadence Design Systems Inc.	Innovators, Ltd.	Hemant Shah, Cadence Design Systems Inc.
Craig Armenti, Zuken USA, Inc.	Ronny Kovartovsky, Frontline PCB Solutions	Chris Shaw, Fujitsu Network Communications
Kjell Asp, Ericsson AB	Kevin Kusiak, Lockheed Martin Space Systems Company	Rainer Taube, Taube Electronic GmbH
Jimmy Baccam, Lockheed Martin Missiles & Fire Control	Jeff Lewis, Holaday Circuits Inc.	Karel Tavernier, Ucamco
Robert Bagsby, Rockwell Collins	Humair Mandavia, Zuken USA, Inc.	Denise Turley, Fujitsu Network Communications
Tomas Bergsten, Ericsson AB	Jason Marsh, Nextflex	John Vaughn, Fujitsu Network Communications
Gary Carter	Karen McConnell, Northrop Grumman Corporation	Wenju Wang, Cadence Design Systems Inc.
Max Clark, Mentor Graphics Corporation	Kristopher Moyer, Aerojet Rocketdyne	Steve Watt, Zuken USA, Inc.
Joseph Clark, DownStream Technologies, LLC	William Newhard, DownStream Technologies, LLC	Jamie Wise, WISE Software Solutions Inc.
Amy Clements, Zuken USA, Inc.	Sheldon Parnes, Ansys	
Larry Frost, Sanmina Corporation	James Pierce, Axiom Electronics, LLC	
Terry Hoffman, Cisco Systems Inc.		
Satoshi Kojima, NEC Solution		

A special note of thanks goes to the following individuals for their dedication to bringing this project to fruition. We would like to highlight those individuals who made major contributions to the development of this standard.

Edward Acheson, Cadence Design Systems Inc.	Ronny Kovartovsky, Frontline PCB Solutions	Chris Shaw, Fujitsu Network Communications
Kent Balias, TTM Technologies	Humair Mandavia, Zuken USA, Inc.	John Vaughn, Fujitsu Network Communications
Tomas Bergsten, Ericsson AB	William Newhard, DownStream Technologies, LLC	Steve Watt, Zuken USA, Inc.
Gary Carter	James Pierce, Axiom Electronics, LLC	Jamie Wise, WISE Software Solutions Inc.
Joseph Clark, DownStream Technologies, LLC	Hemant Shah, Cadence Design Systems Inc.	
Terry Hoffman, Cisco Systems Inc.		

This Page Intentionally Left Blank

TABLE OF CONTENTS

1	SCOPE	1
1.1	Focus and intent	1
1.2	Notation	1
2	APPLICABLE DOCUMENTS	2
2.1	Documentation conventions	2
3	REQUIREMENTS	5
3.1	Rules concerning the use of XML and XML Schema	7
3.1.1	File readability and uniformity	7
3.1.2	File markers	7
3.1.3	File extension	7
3.1.4	File remarks	7
3.1.5	Character set definition	7
3.1.6	Child Element Order	7
3.1.7	XML Validation and Schema File Location	8
3.2	Naming attributes within a IPC-2581 file	8
3.2.1	Data organization and identification rules	8
3.2.2	The Use of XML elements and types	9
3.2.3	Attribute base types (governing templates)	9
3.2.4	Coordinate system and transformation rules	11
3.3	Transformation characteristics (Xform)	12
3.3.1	The x and y Offset attributes	13
3.3.2	The rotation attribute	13
3.3.3	The mirror attribute	14
3.3.4	The scale attribute	14
3.3.5	The x and y Location attributes	14
3.4	Base Elements	15
3.4.1	Line	15
3.4.2	Arc	16
3.4.3	PolyBegin	17
3.4.4	Polygon	17
3.4.5	Polyline	18
3.4.6	Cutout	19
3.4.7	Outline	20
3.4.8	PinRef	21
3.4.9	Location	21
3.5	Substitution Groups	22
3.5.1	ColorGroup	24
3.5.2	Fiducial	26
3.5.3	FirmwareGroup	26
3.5.4	FontDef	28
3.5.5	LineDescGroup	32
3.5.6	FillDescGroup	34

3.5.7	PolyStep	36
3.5.8	Simple	38
3.5.9	StandardPrimitive	39
3.5.10	StandardShape	64
3.5.11	UserPrimitive	64
3.5.12	UserShape	68
3.5.13	Feature	68
4	CONTENT	69
4.1	FunctionMode	70
4.1.1	Schema Sections	71
4.1.2	Modes	75
4.1.3	FunctionMode Element.....	75
4.2	StepRef	77
4.3	LayerRef	77
4.4	BomRef	78
4.5	AvlRef	79
4.6	DictionaryStandard	80
4.7	DictionaryUser	81
4.8	DictionaryFont	82
4.9	DictionaryLineDesc	83
4.10	DictionaryFillDesc	85
4.11	DictionaryColor	86
4.12	DictionaryFirmware	87
5	LOGISTIC HEADER	88
5.1	LogisticHeader	88
5.2	Role	89
5.3	Enterprise	90
5.4	Person	92
6	HISTORY RECORD	93
6.1	HistoryRecord	93
6.2	FileRevision	95
6.3	SoftwarePackage	95
6.4	ChangeRec	97
7	BOM (Material List).....	98
7.1	BOM Header	100
7.2	BomItem	100
7.2.1	BomDes.....	102
7.2.2	Characteristics	110
8	ELECTRONIC COMPUTER AIDED DESIGN (ECAD)	114
8.1	CadHeader.....	114
8.1.1	Spec	115
8.1.2	Property.....	135
8.1.3	ChangeRec	136
8.2	CadData.....	137

8.2.1	Layer	139
8.2.2	Stackup	142
8.2.3	Step	149
8.2.4	DfxMeasurementList	196
9	APPROVED VENDOR LIST (AVL)	199
9.1	AvlHeader	200
9.2	AvlItem	201
9.2.1	AvlVmpn	202
10	GLOSSARY	204
10.1	Process Flow Descriptions	204
10.2	Terms and Definitions	205
10.3	Enumerated strings of 2581	206
11	REFERENCE INFORMATION	206
11.1	IPC	206
11.2	American National Standards Institute	207
11.3	Department of Defense	207
11.4	Electronic Industries Association	207
11.5	International Organization for Standards (ISO)	207
Appendix A	IPC-7351 Naming Convention for Land Patterns	208
Appendix B	Panel Instance File	212
Appendix C	Potential Reference Designator Assignment for Non Electrical Items	215
Appendix D	Summary of Changes from Revision B	216

This Page Intentionally Left Blank

Generic Requirements for Printed Board Assembly Products Manufacturing Description Data and Transfer Methodology

1 SCOPE

This standard specifies the XML schema that represents the intelligent data file format used to describe printed board and printed board assembly products with details sufficient for tooling, manufacturing, assembly, and inspection requirements. This format may be used for transmitting information between a printed board designer and a manufacturing or assembly facility. The data is most useful when the manufacturing cycle includes computer-aided processes and numerical control machines.

The data can be defined in either English or International System of Units (SI) units. The format is a convergence of the IPC-2511 "GenCAM" and the Valor Computerized Systems "ODB-X" format structure.

1.1 Focus and intent

The generic format requirements are provided in a series of standards focused on printed board manufacturing, assembly, and inspection testing. This standard series consists of a generic standard (IPC-2581) that contains all the general requirements. There are seven sectional standards that are focused on the XML details necessary to accumulate information in the single file, that address the needs of the manufacturing disciplines producing a particular product.

The sectional standards (IPC-2582 through 2588) paraphrase the important requirements and provide suggested usage and examples for the topic covered by the sectional standard.

1.2 Notation

Although the data would be contained in a single file, the file can have different purposes as described in Section 4. The XML schema used for this standard follows the notations set forth by the W3C and is as follows:

- element – Element appears exactly one time
- element? – Element may appear 0 or 1 times
- element* – Element may appear 0 or more times
- element+ – Element may appear 1 or more times

Any IPC-258X file is composed of a high level element (IPC-2581) that contains up to six sub-elements:

- Content – information about the contents of the 258X file
- LogisticHeader – information pertaining to the order and supply data
- HistoryRec – change information of the file
- Bom – Bill of Materials (Material List) information
- Ecad – Computer Aided Design (engineering) information
- Avl – Approved Vendors List information

2 APPLICABLE DOCUMENTS

The following documents contain requirements which, when referenced, constitutes provisions of IPC-2581. At the time of publication, the editions indicated were valid. All documents are subject to revision and parties entering into agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below.

The revision of the document in effect at the time of solicitation **shall** take precedence.

IPC-T-50 Terms and Definitions for Interconnecting and Packaging Electronic Circuits

IPC-2501 Definition for Web-Based Exchange of XML Data

IPC-2524 PWB Fabrication Data Quality Rating System

IPC-2511 Generic Requirements for Implementation of Product Manufacturing Description Data and Transfer XML Schema Methodology

IPC-2571 Generic Requirements for Electronics Manufacturing Supply Chain Communication - Product Data eXchange (PDX)

IPC-2576 Sectional Requirements for Electronics Manufacturing Supply Chain Communication of As-Built Product Data - Product Data eXchange

IPC-2578 Sectional Requirements for Supply Chain Communication of Bill of Material and Product Design Configuration Data - Product Data eXchange

IPC-7351 Generic Requirements for Surface Mount Design and Land Patterns

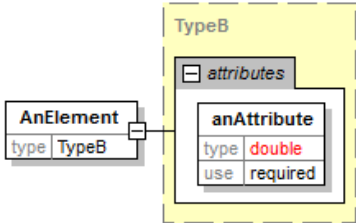
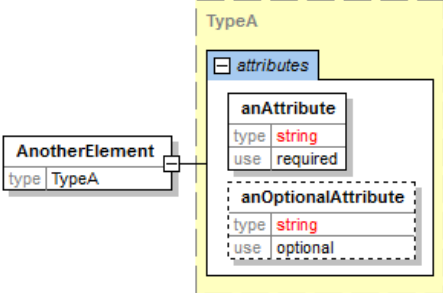
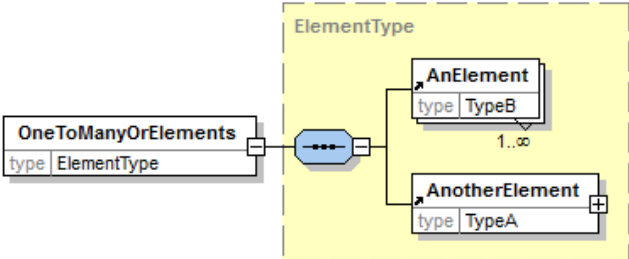
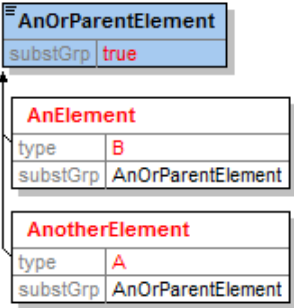
2.1 Documentation conventions

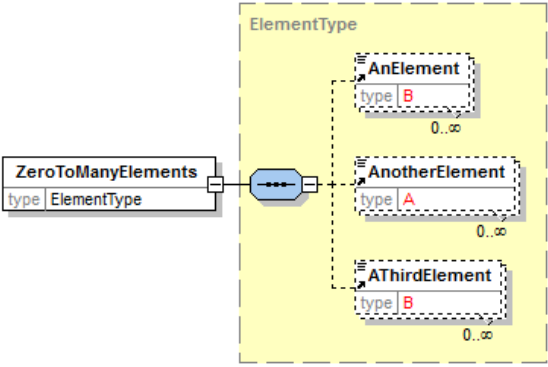
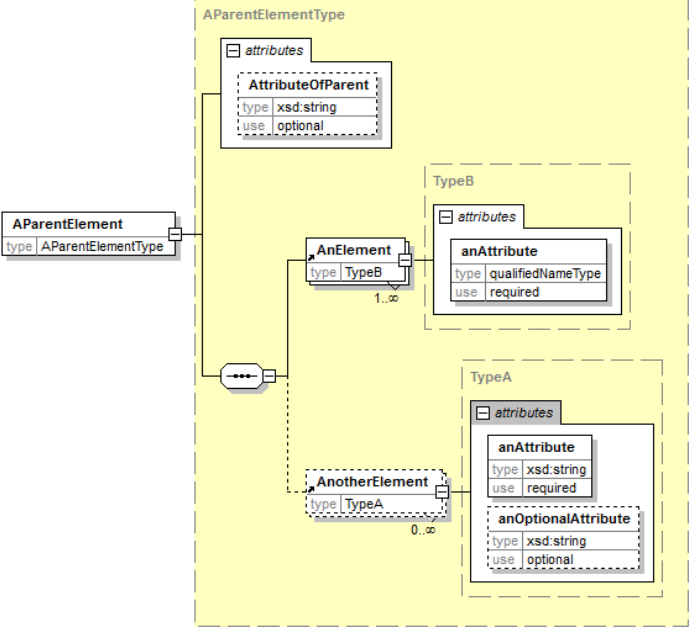
The XML file format standard and the XML Schema definition language standard, as defined the by World Wide Web Consortium (W3C), have been adopted by IPC for use in the IPC-2500 series of standards.

In addition to the text based schema notation, this document provides graphical representation of the structure of the file format. The XML diagrams are designed to effectively illustrate the structure and cardinality of elements and attributes that make up any IPC-258X file. The notation in the graphics does not provide a complete visualization of the schema definition for the file format, but it does provide a good top down overview. Should there be any conflict between the graphical notation and the schema notation, the authoritative definition is the schema notation.

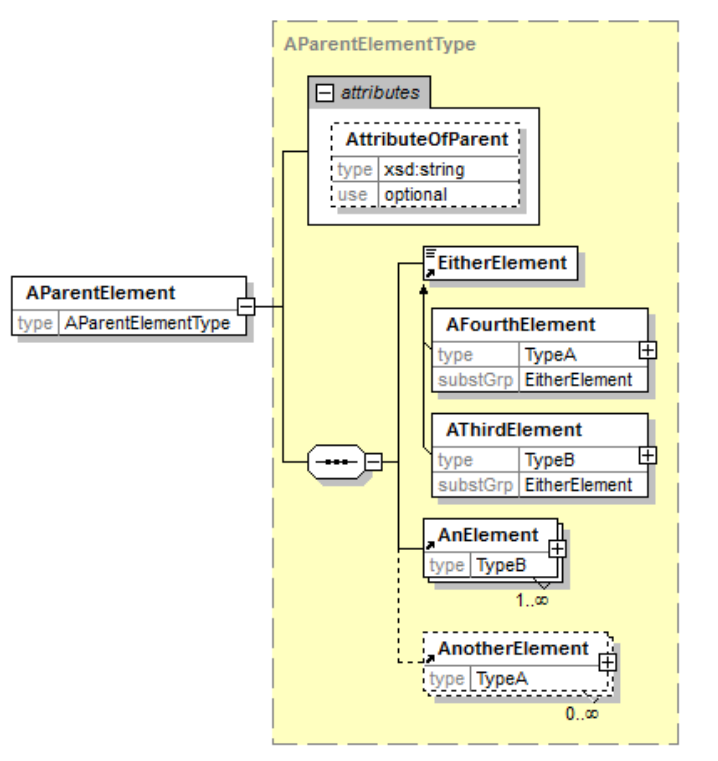
Table 1 provides an overview of the graphical notation used in the document.

Table 1 Graphical Notation Overview

This diagram depicts an element named AnElement that is of type TypeB. There is one attribute, named anAttribute, which is of type double. The attribute is required.	 The diagram shows a box for 'AnElement' with 'type TypeB'. A line connects it to a dashed box for 'TypeB'. Inside 'TypeB' is a box for 'attributes' containing 'anAttribute' with 'type double' and 'use required'.
Example: <AnElement anAttribute="14.44e-3"/> Note that all attribute values must be enclosed in quotes, regardless of type.	
This diagram depicts an element named AnotherElement that is of type TypeA with two attributes. The attribute anAttribute is required. The second attribute, anOptionalAttribute, is optional. Both attributes are of type string.	 The diagram shows a box for 'AnotherElement' with 'type TypeA'. A line connects it to a dashed box for 'TypeA'. Inside 'TypeA' is a box for 'attributes' containing 'anAttribute' (type string, use required) and 'anOptionalAttribute' (type string, use optional).
Examples: <AnotherElement anAttribute="red" anOptionalAttribute="a string" /> <AnotherElement anAttribute="blue" />	
The ElementType OneToManyOrElements is the parent of an unordered list of one or more instances of the elements AnElement and AnotherElement. The type TypeA occurs only once while TypeB can occur many times.	 The diagram shows a box for 'OneToManyOrElements' with 'type ElementType'. A line connects it to a dashed box for 'ElementType'. Inside 'ElementType' is a list containing 'AnElement' (type TypeB, 1..∞) and 'AnotherElement' (type TypeA, 1).
The arrow indicates a substitution of the element named AnOrParentElement. The substitution declares that one and only one occurrence is allowed. The AnOrParentElement can have one of AnElement or AnotherElement as a child element.	 The diagram shows a box for 'AnOrParentElement' with 'substGrp true'. Below it are two boxes: 'AnElement' (type B, substGrp AnOrParentElement) and 'AnotherElement' (type A, substGrp AnOrParentElement). An arrow points from the parent box to the child boxes.

The optional indication for the three element children shows that none may be present or many versions of all three are possible.	 The diagram shows a box labeled 'ZeroToManyElements' with 'type ElementType'. It is connected to a dashed box labeled 'ElementType'. Inside 'ElementType' are three elements: 'AnElement' (type B, 0..∞), 'AnotherElement' (type A, 0..∞), and 'AThirdElement' (type B, 0..∞). Each element has a dashed border and a small 'is' icon in the top left corner.
This diagram depicts an element, AParentElement, of type AParentElementType. This element has one attribute, attributeOfParent, which is optional. The lines indicate that occurrences of AnElement and AnotherElement appear in the order with AnElement indicated as required while AnotherElement is optional. Both elements have a required attribute; however, if AnotherElement is instantiated, anOptionalAttribute is also possible.	 The diagram shows a box labeled 'AParentElement' with 'type AParentElementType'. It is connected to a dashed box labeled 'AParentElementType'. Inside 'AParentElementType' are: an 'attributes' section with 'AttributeOfParent' (type xsd:string, use optional); an 'AnElement' (type TypeB, 1..∞) connected to a dashed box 'TypeB' containing 'anAttribute' (type qualifiedNameType, use required); and an 'AnotherElement' (type TypeA, 0..∞) connected to a dashed box 'TypeA' containing 'anAttribute' (type xsd:string, use required) and 'anOptionalAttribute' (type xsd:string, use optional). All elements have dashed borders and small 'is' icons.

This diagram depicts a type, AParentElementType, that contains a sequence starting with one of AThirdElement or AFourth element followed by 1-n AnElement and an optional final AnotherElement. The AParentElement also has an optional attribute.



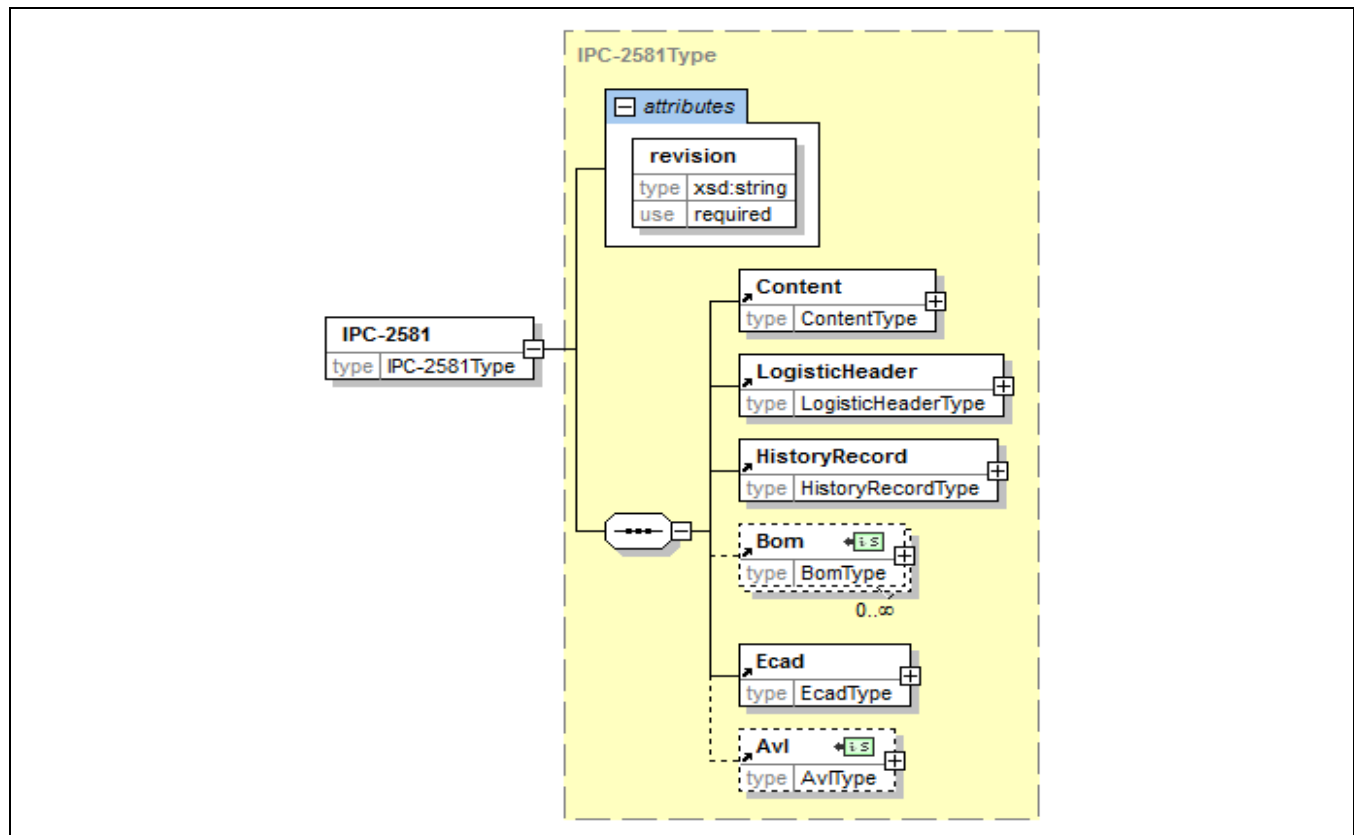
3 REQUIREMENTS

The XML schema contained in this document describes the structure of a generic computer-aided manufacturing 258X exchange format. The document specifies data elements specifically designed to establish the information exchange related to the data needed by printed board manufacturing, and assembly including inspection of those products.

The XML schema defines the configuration of mandatory and optional elements, as well as mandatory and optional attributes. The Top Level (TopElement) of the schema contains six major elements. The schema notation specifies that the 6 top-level elements are required to appear in the order shown in Figure 1. The order of appearance in the file is significant. For instance, the appearance of graphics on a layer is dependent on the order of appearance in the file. The order is also important because elements often reference information that is defined elsewhere in the file in order to eliminate redundancy within the file. The file is structured to allow all references to be resolved in one pass.

An implementation of the XML schema must be able to facilitate the reading and/or writing of all characteristics defined within the requirements stated in the Mode function of this standard. Some tools may have only read capability; some may have only write capability. Some tools may have both read/write capability. All schema defined in the standard as mandatory (1-1, 1-n occurrences) **shall** be executed as appropriate. Tool providers **shall** identify their capability by Mode (e.g. FULL, FABRICATION, ASSEMBLY etc.) plus 2581R; 2581W; or 2581RW.

Each element has a specific function or task. Accordingly, the information interchange for a specific purpose is possible only if that element is populated. The ability to select those characteristics that are appropriate for a given task makes the schema a robust methodology for defining only those areas and characteristics that are necessary to produce a given product. Figure 1 shows the potential children elements of the Top Element (IPC-2581).



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
IPC-2581	IPC-2581Type	The generic name of the file used to describe Design, Fabrication, Assembly, and Test information in accordance with the IPC-2581 standard.	1-1
revision	string	The status of the IPC-2581 file content.	1-1
Content	ContentType	The Content element defines the function of the file, and references the major sections of the product description (i.e., Step, Layer, Bom and Avl). In addition, there are six dictionaries indicated in Content that would contain the pre-described information needed for the file details.	1-1
LogisticHeader	LogisticHeaderType	The LogisticHeader describes information pertaining to ordering and delivery, by identifying individuals and locations responsible for these functions.	1-1
HistoryRecord	HistoryRecordType	The HistoryRecord element provides a sequential change number for the IPC-2581 file. The number is changed every time the controlled version of the IPC-2581 file is modified. Also identified are the change approval conditions.	1-1
Bom	BomType	The Bill of Materials for the board is a list of all the different components to be used for the assembly of the board, arranged by their OEM Design Number (ODN) and the materials used to fabricate the printed board.	0-n
Ecad	EcadType	The Ecad section describes the Computer Aided Design data of the job, including all the graphical description of the layers, component location, panel design, etc.	1-1
Avl	AvlType	The AVL element contains the list of matching manufacturer's part numbers (MPNs) and vendor information of a certain component parts. Approved vendors of printed board materials may also be included.	0-1

Figure 1 IPC-258X Children Element

3.1 Rules concerning the use of XML and XML Schema

The rules required to define syntax and semantics of the 258X file format notation have been simplified by the adoption of the W3C standards for XML Schema and XML file formats. These two standards are well specified by the W3C. The popularity of these standards has led to the development of many commercial and open source software tools and libraries that conform to the W3C standards.

A 2581 file begins with the notation of the revision of the generic standard and the latest revision letter and amendment number, if any amendments exist, e.g. <IPC-2581 revision="B1">, and ends with the </IPC-2581> tag. The content between these tags must match the .xsd definition of the 2581 schema as defined by the IPC-2581 XML. The latest specification and schema release is found at:
<http://webstds.ipc.org/standards.htm#x2580>.

3.1.1 File readability and uniformity

A valid IPC-2581 file must conform to the W3C Canonical XML format. The format is defined by the <http://www.w3.org/TR/xml-c14n> specification. Software tools exist that will take malformed XML and automatically generate Canonical XML.

3.1.2 File markers

An optional checksum can be appended following the </IPC-2581> tag. The checksum is an MD5 message digest algorithm (see Internet RFC 1321: <http://www.ietf.org/rfc/rfc1321.txt>) that is base64 encoded. The checksum starts with the "<" character of the <IPC-2581> tag and ending with the ">" character of the closing </IPC-2581> tag. The checksum follows immediately after the ">" character of the closing </IPC-2581> tag.

The digest provides a 128-bit checksum of the IPC-2581 file contents. The MD5 signature must be base64 encoded (see IETF RFC 1421 for the base64 algorithm) to convert the MD5 signature to a US-ASCII, base64 string. An end of line character will indicate the end of the base64 encoded MD5 signature.

3.1.3 File extension

The file extension for an IPC-2581 file is .xml.

3.1.4 File remarks

The IPC-2581 format permits file remarks using the standard XML commenting notation. They are only to be used to support debugging software. A parser may ignore and discard remarks when reading an IPC-2581 file. File remarks are never to be used to represent design or manufacturing information.

3.1.5 Character set definition

The XML standard uses the Unicode character set. This character set covers the characters used in hundreds of written languages. The XML standard allows several of the Unicode encoding formats to be used in an XML file. IPC-2581 requires the use of the UTF-8 character encoding of the Unicode character set. Although comments and user assigned names may be in any language of choice, all qualified names or enumerated string names **shall** be in English only.

3.1.6 Child Element Order

The IPC-2581 schema uses the <sequence> order indicator throughout, which means that all child elements of a given parent element must be instantiated in an IPC-2581 file in the order in which they are defined in the schema. For example, referring to section 8.2.3.12.3, consider the element Pad, which has four child elements Xform, Location, Feature and PinRef, defined in that order, as shown below:

```

<xsd:element name="Pad" type="PadType"/>
<xsd:complexType name="PadType">
  <xsd:sequence>
    <xsd:element ref="Xform" minOccurs="0"/>
    <xsd:element ref="Location"/>
    <xsd:element ref="Feature"/>
    <xsd:element ref="PinRef" minOccurs="0"/>
  </xsd:sequence>

```

The element Xform, if it exists (as it is optional) must occur first in any instance of Pad, followed by Location, and so on, as shown in the example below:

```

<Pad padstackDefRef="LS32X67">
  <Xform rotation="90.000"/>
  <Location x="7.1500" y="10.1500"/>
  <StandardPrimitiveRef id="RECTCENTER_3"/>
  <PinRef pin="34"/>
</Pad>

```

If the child elements appear in any other order in an IPC-2581 file then the XML schema of that file is strictly invalid.

3.1.7 XML Validation and Schema File Location

An attribute `schemaLocation` can be appended to the attribute list of the initial `<IPC-2581>` tag of any given IPC-2581 file. Its value must be two URIs separated by a space, the first URI, <http://webstds.ipc.org/2581>, being fixed as the namespace location, and the second, <http://www.ipc.org/2581/IPC-2581B1.xsd>, being the location of the IPC-2581 revision B1 schema definition (.xsd) file. This can enable validation tools to be able to validate the file contents against the schema definition. It is highly recommended that validation is done, to ensure your IPC-2581 files conform to the schema. The following is an example of the `schemaLocation` attribute, pointing to the IPC-2581 revision B1 .xsd file on the IPC website.

```

<IPC-2581 revision="B1" xmlns="http://webstds.ipc.org/2581" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://webstds.ipc.org/2581 http://www.ipc.org/2581/IPC-2581B1.xsd">

```

The following is a list of all current and previous revision schema files available:

<http://www.ipc.org/2581/IPC-2581B1.xsd>
<http://www.ipc.org/2581/IPC-2581B.xsd>
<http://www.ipc.org/2581/IPC-2581A.xsd>

3.2 Naming attributes within a IPC-2581 file

There are two types definitions used to name attributes in an IPC-2581 file. The first type of name is `qualifiedNameType`. This type includes an attribute prefix in the name to allow for multiple products (boards, assemblies, or panels) to be defined in the single IPC-2581 file. See section 3.2.1 for more details.

The second type definition is a `shortName`. This type provides syntax restrictions but it does not have the prefix ability of the `qualifiedNameType`. It is used to define the attributes such as `OEMDesignNumber`, `Bom:name` and `Step:name`, which are assumed to be inherently unique across all products within the same file. See Table 3 for further details of the data type definitions.

3.2.1 Data organization and identification rules

To prevent collisions between attributes having the same value used by the different partitions (boards, assemblies, or panels) within a single IPC-2581 file, an attribute prefix in the *qualifiedNameType* attribute type can be used to separate the values into 'namespaces' per partition. (Note that these 'namespaces' alter attribute values, so they should not to be confused with XML namespaces, which alter attribute or element names.) This is necessary because an IPC-2581 file may contain information describing an arbitrary collection of partitions

that are allowed within a single IPC-2581 file. That is, if an attribute is of type `qualifiedNameType` then it can be of the form `A:B`, where `B` is the attribute value, and `A` is a prefix that can make the value unique if repeated across multiple products in the same file. For example, if 3 boards in the same file all have a `refDes` `R1`, then `R1` could be represented by `X:R1`, `Y:R1`, `Z:R1`, where `X`, `Y`, and `Z` are unique prefixes assigned to each of the 3 boards. This mechanism also prepares the way for a distributed database of IPC-2581 design data in which the data can be trusted to be universally unambiguous.

3.2.2 The Use of XML elements and types

A comprehensive overview of XML Schema can be found in the W3C XML Schema Primer. This section briefly describes the decisions that were made in the development of the IPC-258X schema. Reviewing the Primer is recommended prior to reading this section.

The XML Schema defines a namespace mechanism that can be used when defining element names. The W3C also provides a set of general purpose element and attribute types, such as `xsd:string`, `xsd:double`, and `xsd:datetime`. The IPC-2581 format uses these standard types, however the documentation of the IPC-258X standard has been defined without the use of a namespace prefix for element names within an IPC-258X file.

Each of the schema elements has a prefix, “`xsd:`”, which is associated with the XML Schema namespace through the declaration, `xmlns:xsd=`“<http://www.w3.org/2000/08/XMLSchema>”, that appears in the schema element. The prefix `xsd:` is used by convention to denote the XML Schema namespace, although any prefix can be used. The same prefix, and hence the same association, also appears on the names of built-in simple types, e.g. `xsd:string`. The purpose of the association is to identify the elements and simple types as belonging to the vocabulary of the XML Schema language rather than the vocabulary of the schema author.

In XML Schema, there is a basic difference between complex types that allow elements in their content and may carry attributes, and simple types that cannot have element content and cannot carry attributes. There is also a major distinction between definitions that create new types (both simple and complex), and declarations that enable elements and attributes with specific names and types (both simple and complex) to appear in document instances.

New complex types are defined using the `complexType` element and such definitions typically contain a set of element declarations, element references, and attribute declarations. The declarations are not themselves types, but rather an association between a name and constraints that govern the appearance of that name in documents governed by the associated schema. Elements are declared using the “`element`,” and attributes are declared using the “`attribute`.”

3.2.3 Attribute base types (governing templates)

The attribute basic types (`SimpleTypes`) provided by XML Schema are defined by the W3C. They are easy to distinguish from the IPC-258X types because the W3C type is always prefixed with “`xsd:`”. The W3C datatypes are defined in <http://www.w3.org/2000/10/XMLSchema> (XML Schema Part 2).

Table 2 defines those W3C basic types that are used to define attributes in the IPC-258X schema. The `xsd:string` type is constrained to create specific base types for special purpose strings, such as `qualifiedNameType` and `shortName`. The rules for special number types and the date format are also defined. Table 3 defines those basic types that have been standardized for use within the IPC-258X format.

Table 2 Governing Template Basic Types Defined by W3C

xsd:string	A W3C standard data type for a Unicode character string. The characters are from the UTF-8 character set as defined in http://www.ietf.org/rfc/rfc2279.txt .
xsd:double	A W3C standard data type for a binary floating-point number. The W3C definition of xsd:double is in http://www.w3.org/TR/xmlschema-2/. The xsd:double is a number where the value can be positive, negative, integer or floating point, with at least 7 digits of precision. Numbers are assumed to be positive but can be explicitly designated as positive by preceding the number with a '+' (ASCII decimal 43) character. Negative numbers must be explicitly designated as negative by a preceding '-' (ASCII decimal 45) character. An internal representation of an IEEE double precision floating-point number is assumed. This range of values for IEEE doubles is defined as $3.4 \times 10^{-38} \leq \text{value} \leq 3.4 \times 10^{+38}$. The format for representing a double is the same as the format used in the computer languages C, Perl, Python, or TCL. For example, all the following are legal numbers: 1.005 ; 0.01; .01; -2.334e-33; .224e-2
xsd:nonNegativeInteger	A W3C standard data type for non-negative integer numbers. The W3C definition of xsd:nonNegativeInteger is in http://www.w3.org/TR/xmlschema-2/. The range of values allowed are $0 \leq \text{value} \leq 2147483647$ (the non-negative values that fit in a 32 bit signed integer).
xsd:positiveInteger	A W3C standard data type for positive integer numbers. The W3C definition of xsd:positiveInteger is in http://www.w3.org/TR/xmlschema-2/. The range of values allowed are $1 \leq \text{value} \leq 2147483647$ (the positive values that fit in a 32 bit signed integer).
xsd:dateTime	The W3C standard data type for the current date and time is xsd:dateTime. (See http://www.w3.org/TR/NOTE-datetime-970915.html.) The following formats from the W3C specification are recommended for 258X files: Complete date plus hours, minutes and seconds: YYYY-MM-DDThh:mm:ssTZD (e.g. 1997-07-16T19:20:30.4536+01:00) Complete date plus hours, minutes, seconds and a decimal fraction of a second: YYYY-MM-DDThh:mm:ss.sTZD (e.g. 1997-07-16T19:20:30.45+01:00) where: YYYY = four-digit year MM = two-digit month (01=January, etc.) DD = two-digit day of month (01 through 31) hh = two digits of hour (00 through 23) (am/pm NOT allowed) mm = two digits of minute (00 through 59) ss = two digits of second (00 through 59) s = one or more digits representing a decimal fraction of a second TZD = time zone designator (Z or +hh:mm or -hh:mm)
xsd:anyURI	A W3C standard data type for hyperlinks. The W3C definition of xsd:anyURI is in http://www.w3.org/TR/xmlschema-2/ .
xsd:unsignedByte	The W3C standard for an unsigned byte (an unsigned 8 bit integer with a value between 0-255.) The W3C definition of xsd:unsignedByte is in http://www.w3.org/TR/xmlschema-2/ .
xsd:base64Binary	The data is encoded using base64. (see IETF RFC 1421 for the base64 algorithm and http://www.w3.org/TR/xmlschema-2/#base64Binary)

Table 3 Governing Template Basic Types Defined by IPC

qualifiedNameType	The <code>qualifiedNameType</code> data type is a data type defined for the IPC-258X series. The type is a restricted <code>xsd:string</code> data type where the pattern of the string must match the regular expression as shown by the pattern value in the following definition: <pre><xsd:simpleType name = "qualifiedNameType"> <xsd:restriction base = "xsd:string"> <xsd:pattern value="([a-zA-Z0-9_\.+*\&lt;&gt;])*([a-zA-Z0-9_\.+*\&lt;&gt;])*/> </xsd:restriction> </xsd:simpleType></pre> Examples of strings that match the pattern are: "prefix:name", "name", "prefix1-2:name_3". Note that <code>&gt;</code> and <code>&lt;</code> are single entities per UTF-8 definition that represent the characters <code>></code> and <code><</code> respectively. The characters <code>&</code> and <code>;</code> that form the entities are not valid by themselves.
nonNegativeDouble	The <code>nonNegativeDouble</code> data type is defined for the 258X series. The type restricts an <code>xsd:double</code> to positive numbers, inclusive of 0. The non-negative range of values for IEEE doubles is defined as $0.0 \leq \text{value} \leq 3.4 \times 10^{38}$.
shortName	The <code>shortName</code> data type is a data type defined for the 258X series. The type is a restricted <code>xsd:string</code> data type where the pattern of the string must match the regular expression <code>"[a-zA-Z0-9_\.+*\&lt;&gt;]*"</code>. The <code>xsd</code> definition of the <code>shortName</code> data type is: <pre><xsd:simpleType name = "shortName"> <xsd:restriction base = "xsd:string"> <xsd:pattern value="[a-zA-Z0-9_\.+*\&lt;&gt;]*"/> </xsd:restriction> </xsd:simpleType></pre> Examples strings that match the pattern is "bob_24", "2bob#4", "Bob/24" Note that <code>&gt;</code> and <code>&lt;</code> are single entities per UTF-8 definition that represent the characters <code>></code> and <code><</code> respectively. The characters <code>&</code> and <code>;</code> that form the entities are not valid by themselves.

3.2.4 Coordinate system and transformation rules

Any geometry defined in a 258X file is defined in a Cartesian coordinate system. The x coordinates become more positive going from left to right (west to east). The y coordinates become more positive going from bottom to top (south to north). The primary side (TOP) of the board, coupon, or panel is in the x-y plane of the coordinate system with the primary side facing up.

The illustration in Figure 2 provides a perspective drawing of a board and a coordinate system. Each product in a 258X file is defined relative to a local coordinate system for the product. The point of origin of the product is located at (0,0) in the local coordinate system.

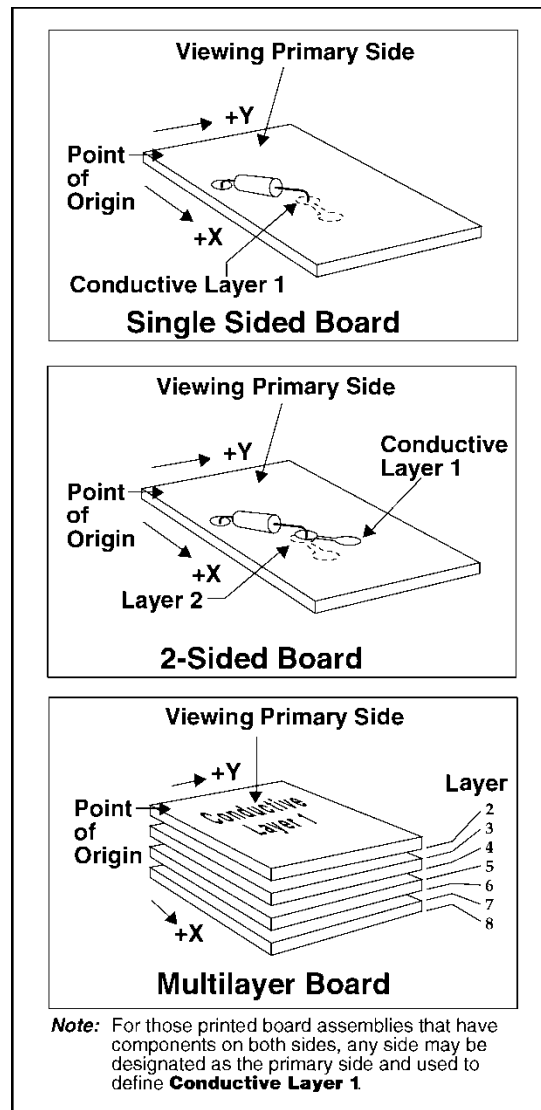


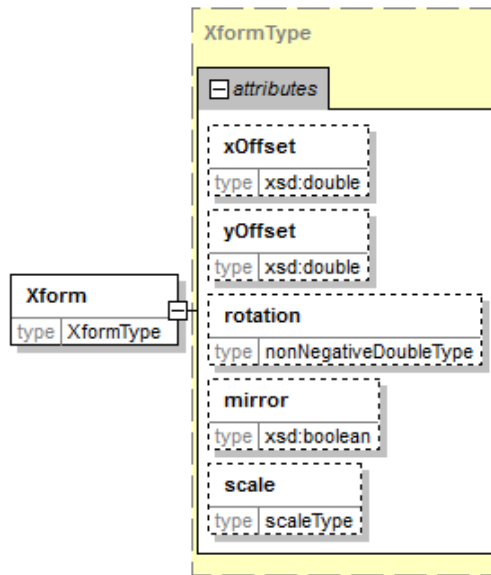
Figure 2 Printed Board Viewing

3.3 Transformation characteristics (Xform)

The Xform element defines a transformation that may be used throughout this specification to define a modification of the original stored data in the Dictionaries, the location and orientation of physical features. Features, Shapes, Primitives or other graphics in the file may manipulate relative to their local Cartesian coordinate system by the values set in the transform. The Xform element can define a modification of the pre-defined feature's point of origin, and then apply rotation, mirroring, scaling and location (x and y) of the image.

The units of measure are defined in the CadHeader as an attribute that describes the details of all the features in the Ecad section. In addition, units of measure are also defined in each of the Dictionaries that contain graphical information. These are DictionaryStandard, DictionaryUser, DictionaryFont, and DictionaryLineDesc. When a pre-defined image from one of the dictionaries is used in the Ecad section, the units of measure must match.

The order of the transformation **shall** always follow the order of the fields in the description. This would be to 1) modify the origin; 2) apply rotation; 3) mirror image; and 4) scale.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
Xform	XformType	An element that provides the ability to reset the point of origin of a predefined target that can be scaled, mirror imaged or rotated.	0-1
xOffset	double	The x offset of a previously defined shape relative to its original point of origin in order to define a new origin used for rotation, mirroring, scaling and location. The CadHeader defines the units of measure.	0-1
yOffset	double	The y offset of a previously defined shape relative to its original point of origin in order to define a new origin used for rotation, mirroring, scaling and location. The CadHeader defines the units of measure.	0-1
rotation	nonNegativeDouble Type	Defines the rotation of a shape about the local origin in degrees. Positive rotation is always counter-clockwise as viewed from the board TOP (primary side). Degree accuracy is expressed as a two place decimal i.e., 45.15; 62.34	0-1
mirror	boolean	An enumerated string is either "true" or "false" (part of the 3WC standard – must be lower case). True indicates that the image is mirrored; False indicates that it is not. Mirror is implemented as "false" by default and MUST be set to "true" if the features/components are to be inverted.	0-1
scale	scaleType	An attribute that defines a "double" dimension whose minExclusiveValue=0.0 representing the multiplication factor of all x and y dimensions. The scale factor does not apply to angular values.	0-1

3.3.1 The x and y Offset attributes

The `xOffset` and `yOffset` attributes are of type `xsd:double`. They define the x and y offset of a shape relative to the origin of a Cartesian coordinate system. The definition of shapes can be nested and the x and y attributes are always interpreted relative to the local coordinate system of the shape to which the transformation applies. The default value for x and y is 0.0.

3.3.2 The rotation attribute

The `rotation` attribute is of type `nonNegativeDouble` that defines the rotation of a shape about the local origin. The interpretation of the value is set globally in the file to units of degrees. The range of the rotation parameter for DEGREES is 0.00 to 360.00 expressed as a `nonNegativeDouble` with an accuracy of a two place decimal. Positive rotation is always counter-clockwise as viewed from the board TOP (primary side), even if the component that is being rotated is on the board BOTTOM (secondary side). Rotation defaults to 0.0, and can be applied to text, or any physical shape.

3.3.3 The mirror attribute

The `mirror` attribute is of data type `boolean`. This type is an enumerated string indicating true or false. The default value for `MIRROR` is “false”. When `mirror` is set to “true” it indicates that all x dimensions are set to a –x value. The proper interpretation of the mirror and rotate attributes are shown in Figure 3. The example shows a unique artwork (14-pin DIP device) placed on the top and bottom of a board at 90.00° rotations.

3.3.4 The scale attribute

The `scale` attribute is of data type `scaleType`. The `scale` attribute is a “double” that must have a value greater than zero. All x and y dimensions of a geometry are multiplied by the scale attribute. The scale factor does not apply to angular values. The default value is 1.0.

3.3.5 The x and y Location attributes

The `xLocation` and `yLocation` attributes are of type `xsd:double`. They define the x and y position where a feature, component, text or other shape is placed. The `xLocation` and `yLocation` coordinate positions a shape by its original origin or its modified origin (x and y Offset) relative to the origin of the image upon which the feature, component etc. is to be located. Mirroring, rotating, and scaling may all apply to the location of the shape as indicated by the `Xform`.

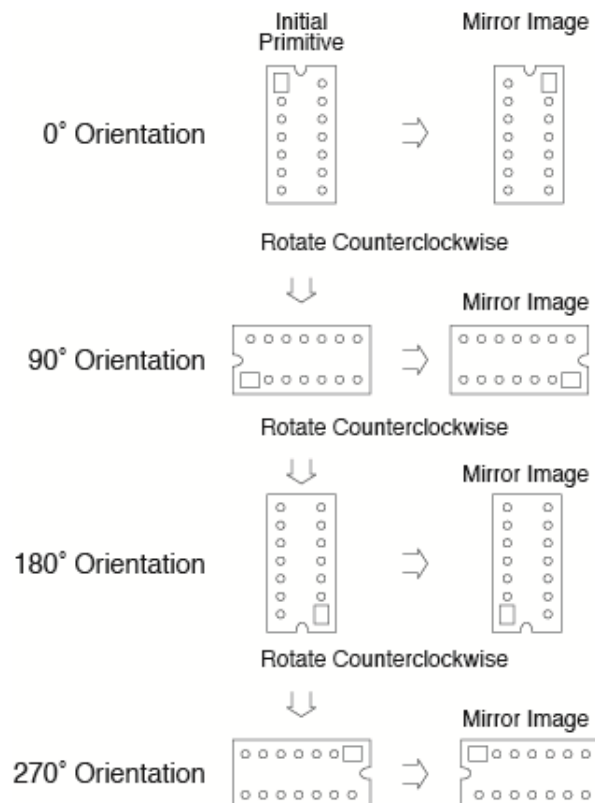


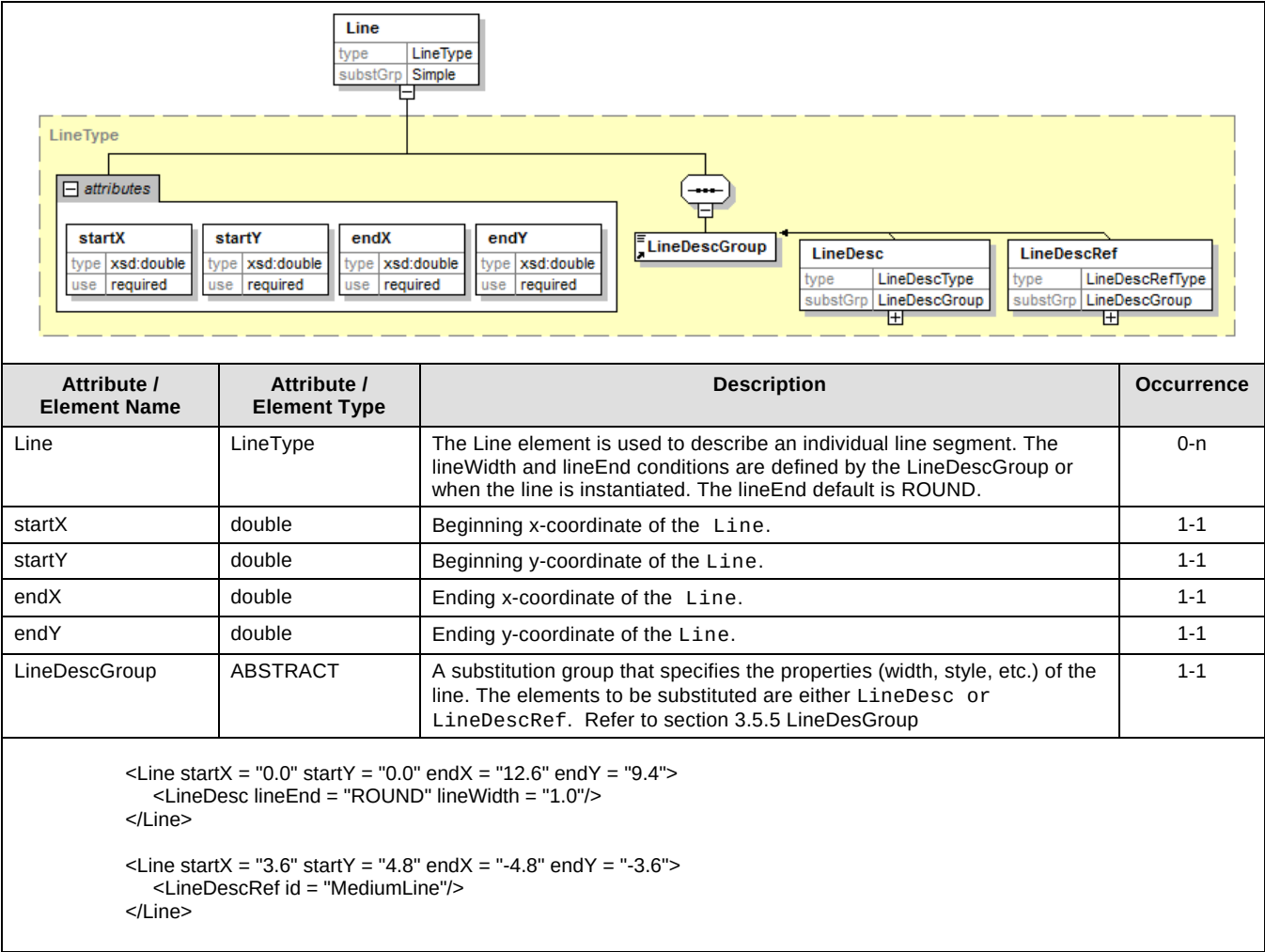
Figure 3 Rotation and Mirror Diagram

3.4 Base Elements

The following elements are used throughout the IPC-2581 schema, either standalone or as 'building block' child elements of more complex elements.

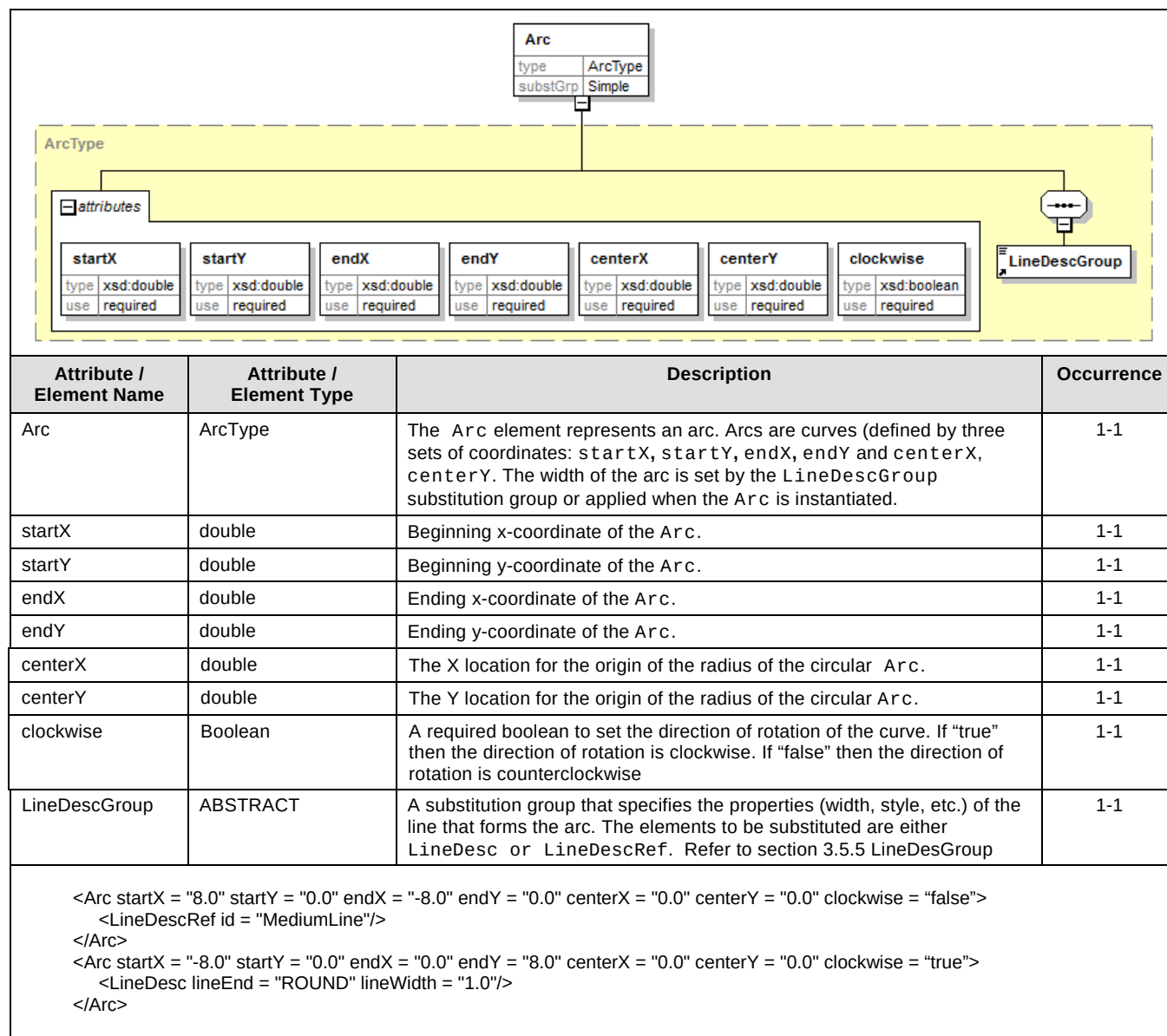
3.4.1 Line

The Line element defines an individual line segment. The start and end points and description of the line are defined.



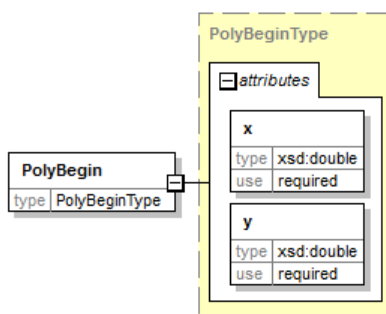
3.4.2 Arc

The Arc element describes a curved line relative to a center of rotation. The start and end points of the line forming the arc are defined, as well as the center of rotation and description of the line.



3.4.3 PolyBegin

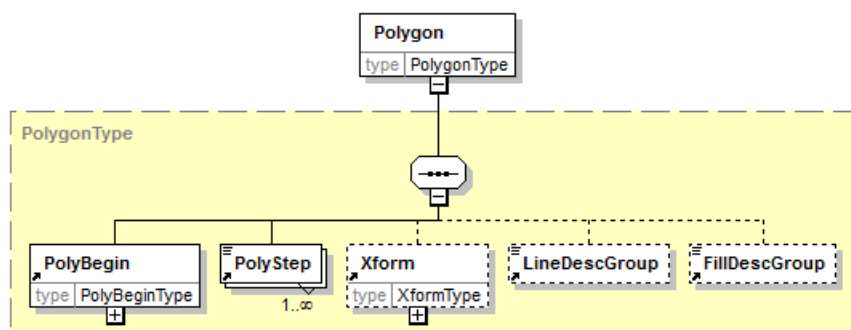
The PolyBegin element defines the starting point of Polygon or Polyline element.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
PolyBegin	PolyBeginType	The PolyBegin element defines the starting point of a polygon or polyline.	1-1
x	Double	The X co-ordinate starting point.	1-1
y	Double	The Y co-ordinate starting point.	1-1

3.4.4 Polygon

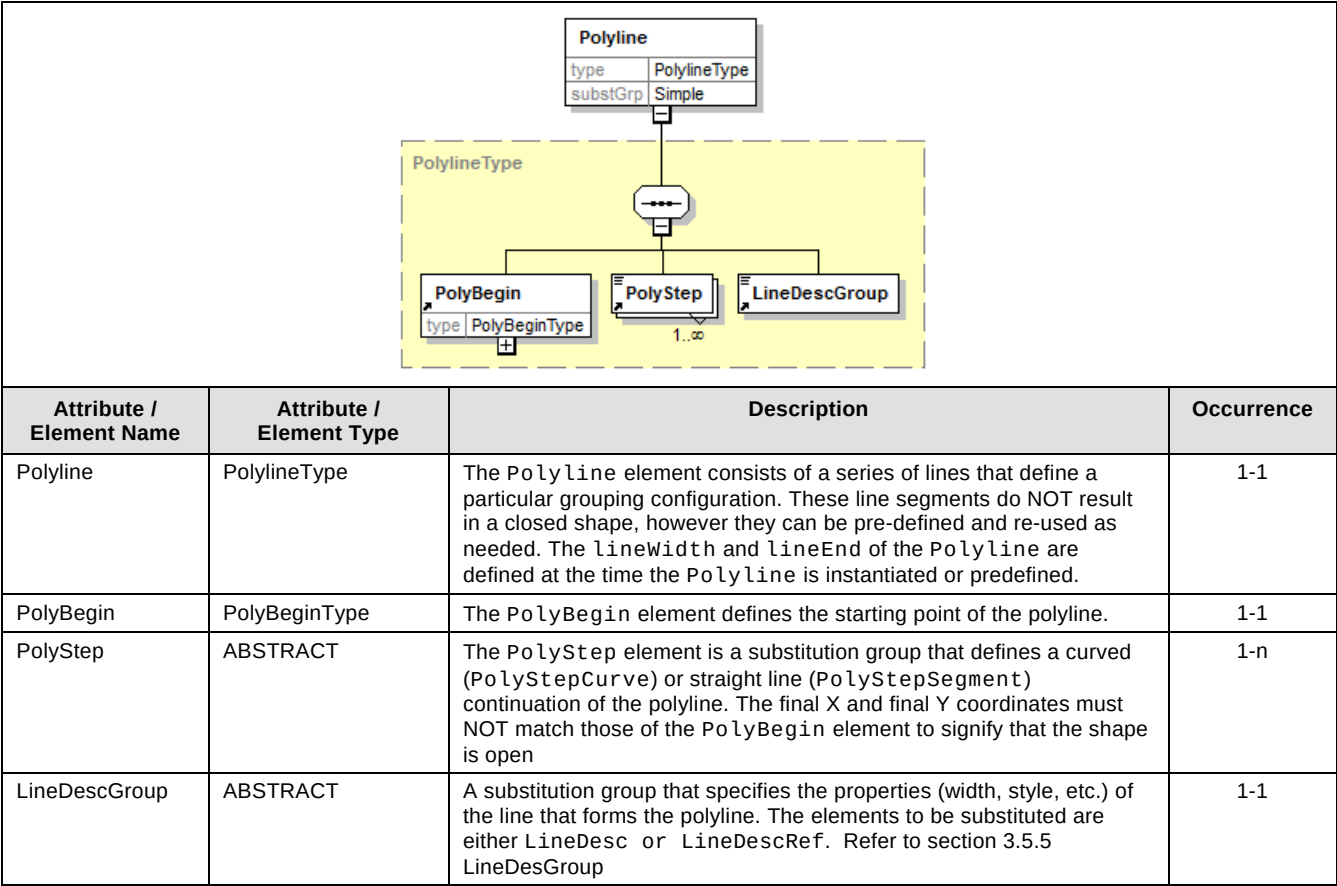
A polygon is a closed shape whose edges do not cross.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
Polygon	PolygonType	A closed shape whose edges do not cross, the coordinates of which are defined relative to the local coordinate system of the polygon.	1-1
PolyBegin	PolyBeginType	The PolyBegin element defines the starting point of the polygon.	1-1
PolyStep	ABSTRACT	The PolyStep element is a substitution group that defines a curved (PolyStepCurve) or straight line (PolyStepSegment) continuation of the polygon. Refer to section 3.5.7 PolyStep. The final X and final Y coordinates must match those of the PolyBegin element to signify that the shape is closed.	1-n
Xform	XformType	An element that provides the ability to reset the point of origin of a predefined target that can be scaled, mirror imaged or rotated. <i>Note: the use of this is not recommended, as it may be removed in the next revision. It is better to assign the Xform to the shape instance.</i>	0-1
LineDescGroup	LineDescGroupType	A substitution group that specifies either LineDesc or LineDescRef. Refer to section 3.5.5 LineDescGroup	0-1
FillDescGroup	FillDescGroupType	A substitution group that specifies either FillDesc or FillDescRef. Refer to section 3.5.6 FillDescGroup	0-1

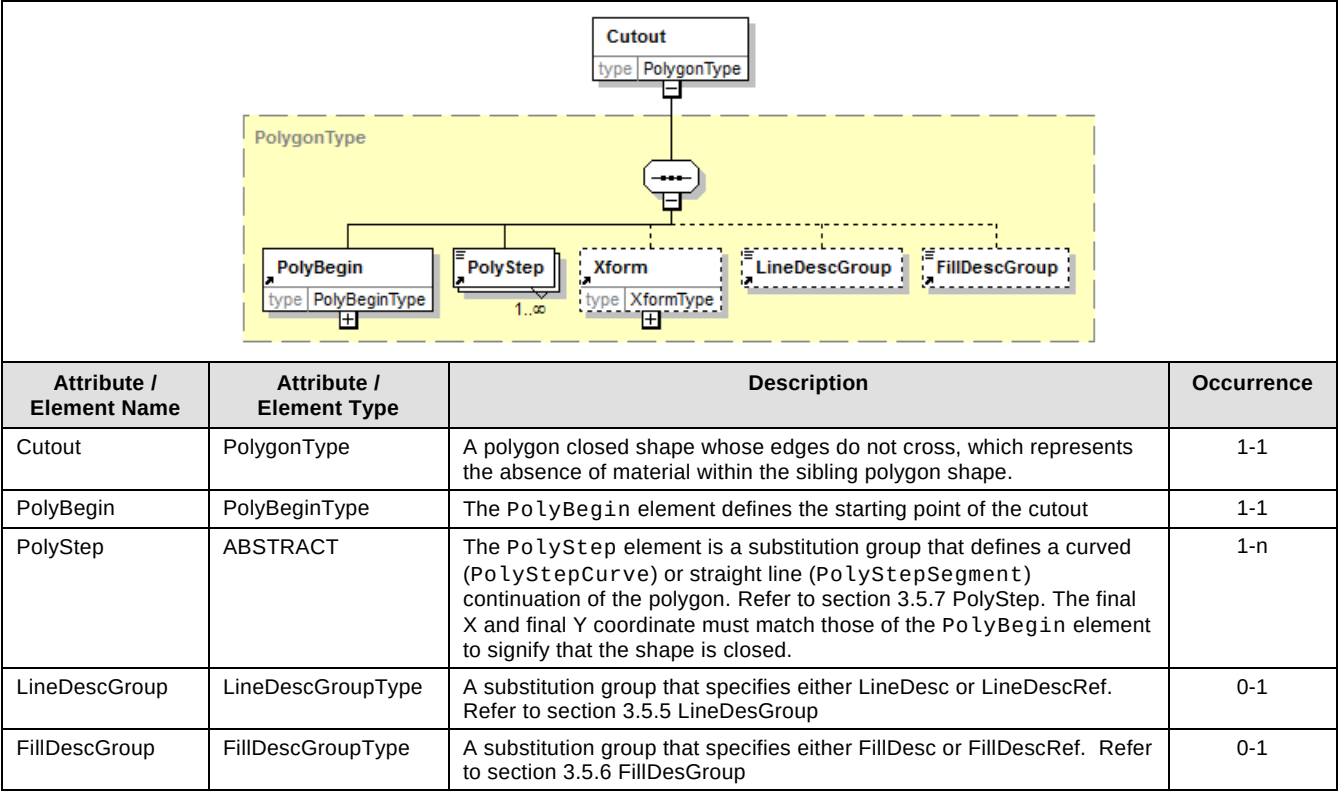
3.4.5 Polyline

The Polyline element consists of a series of lines that define a particular grouping configuration. These line segments do NOT result in a closed shape, however they can be pre-defined and re-used as needed.



3.4.6 Cutout

A Cutout element defines a polygon closed shape whose edges do not cross. Cutout elements must always be paired with a sibling Polygon element, with co-ordinates and size such that the cutout shapes are fully enclosed within the boundry of the polygon, so that the cutouts represent the absence of material within that polygon.



3.4.7 Outline

The Outline element consists of the following characteristics using a Polygon shape to represent a closed shaped group of lines.

Outline type OutlineType substGrp Simple OutlineType Polygon type PolygonType LineDescGroup			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
Outline	OutlineType	An element that has as its sub elements the Polygon and LineDesc elements in order to define a closed shape that has a line width.	1-1
Polygon	PolygonType	The standard description for the Polygon characteristic must be a closed shape. The point of origin of the shape is identified as the 0:0 coordinate. The element includes the start of the polygon definition (PolyBegin) and the appropriate number of PolySteps to complete the closed shape. The lineWidth is through the LineDesc substitution group or defined at a time when the Polygon is instantiated.	1-1
LineDescGroup	LineDescGroupType	A substitution group that specifies the properties (width, style, etc.) of the line that forms the outline. The elements to be substituted are either LineDesc or LineDescRef. Refer to section 3.5.4.1 LineDescGroup	1-1
<Outline> <Polygon> <PolyBegin x = "0.0" y = "6.4"/> <PolyStepSegment x = "12.6" y = "6.4"/> <PolyStepCurve x = "14.8" y = "4.2" centerX = "12.6" centerY = "4.2" clockwise = "true"/> <PolyStepSegment x = "14.8" y = "0.0"/> <PolyStepSegment x = "0.0" y = "0.0"/> <PolyStepSegment x = "0.0" y = "6.4"/> </Polygon> <LineDescRef id = "FineLine"/> </Outline> <Outline> <Polygon> <PolyBegin x = "-10.4" y = "-2.0"/> <PolyStepSegment x = "-10.4" y = "0.0"/> <PolyStepCurve x = "10.4" y = "0.0" centerX = "0.0" centerY = "0.0" clockwise = "true"/> <PolyStepSegment x = "10.4" y = "-2.0"/> <PolyStepSegment x = "0.0" y = "-6.0"/> <PolyStepSegment x = "-10.4" y = "-2.0"/> </Polygon> <LineDesc lineEnd = "ROUND" lineWidth = "0.2"/> </Outline>			

3.4.8 PinRef

PinRef is an element that can be used to associate a layer feature, such as a pad or fiducial, to a component or an attribute of a component

PinRef	PinRefType	An individual Pin related to the place where a component attaches to the net. This description facilitates location of lands on the surface of the board or places where through-hole components are mounted.	1-n
componentRef	qualifiedNameType	The qualifiedNameType that identifies the reference designator used as the attribute refDes of the Component element in Step It is the reference to the component that is connected by the particular Pin and becomes a part of the electrical description of the net. The componentRef attribute is not required when used for a PinRef element under Pad elements which are under LandPattern elements. Otherwise, the componentRef attribute is required for PinRef elements under Pad, LayerPad and LogicalNet elements.	0-1
pin	string	An identification of the component pin, such as the pin number, that becomes a part of the electrical description.	1-1
title	string	An alternate method of relating the pin information providing characteristics of the component lead or termination description.	0-1

3.4.9 Location

The Location element defines the location of an object in terms of the x and y coordinates of its origin relative to the CAD origin. The CadHeader element defines the units of measure.

Location	LocationType	The location of an object, with reference to its origin. Its origin may have been reorientated by the Xform.	1-1
x	double	The x coordinate of the object, in accordance with the Cartesian coordinate system. The CadHeader defines the units of measure.	1-1
y	double	The y coordinate of the object, in accordance with the Cartesian coordinate system. The CadHeader defines the units of measure.	1-1

3.5 Substitution Groups

The IPC-2581 uses the concept of substitution within the XML schema. Various groups of elements have been identified in the body of the standard and have been designated as having a specific focus or purpose. Within the schema, these substitution groups are provided with a name. When a group exists and if they are required according to the instances of the schema, it is mandatory that the substitution name be replaced by one of the acceptable descriptions identified within the group.

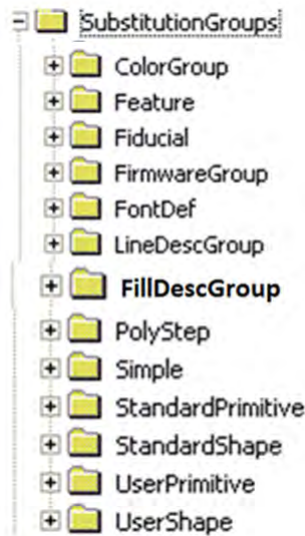
Often a schema needs to specify that one of several different XML Elements can be used with equal validity. For example, in every case where a `Triangle` can be used, it is also permissible to use a `Diamond`, `Hexagon`, `Octagon`, `Oval`, or one of several others: even though these shapes are quite different, they are equivalent as far as the schema is concerned. IPC-2581 handles this by using “substitution groups.”

A substitution group consists of two types of elements: a “head” and elements which may substitute for the head. Furthermore, when the head is denoted as `ABSTRACT`, the substitution is required, rather than optional. In IPC-2581, the heads of all substitution groups are `ABSTRACT`. Thus, it means that a valid instance document is not allowed to contain a `StandardPrimitive` element, but instead, (where `StandardPrimitive` is called for in the schema) a `Triangle`, `Diamond`, `Hexagon`, etc. must be used.

It should be noted that the head of one substitution group may be used within a different substitution group. As an example, the `StandardPrimitive` element is part of the `StandardShape` substitution group, which in turn is part of the `Feature` substitution group. This means that a `Triangle`, `Diamond`, `Hexagon`, etc. may be used wherever a `Feature` or `StandardShape` is called for, as well as wherever a `StandardPrimitive` is called for.

IPC-2581 features several dictionaries that permit specifying some type of information (such as a `StandardPrimitive` or a `LineDesc`) one time, and to reuse that definition as often as necessary. Some substitution groups in IPC-2581 are present to enable specifying either a dictionary entry or the same kind of information defined in specific detail within the body of the file. Any predefined image contained in the Dictionaries must have a unique “id”. It is the “id” name that is used to instantiate information from any of the dictionaries.

There are 13 substitution groups within the IPC-2581 schema. These are shown in the following table.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
ColorGroup	ABSTRACT	A substitution group that permits the substitution of the Color element when it is a child of the parent Set, or Text Elements.	3
Feature	ABSTRACT	A substitution group that permits the substitution of the Feature element when it is a child of the parent Set element.	1
Fiducial	ABSTRACT	A substitution group that permits the substitution of the Fiducial element when it is a child of the parent Set element.	1
FirmwareGroup	ABSTRACT	A substitution group that permits the substitution of the FirmwareGroup element when it is a child of the parent Firmware element.	1
FontDef	ABSTRACT	A substitution group that permits the substitution of the FontDef element when it is a child of the parent EntryFont element.	1
LineDescGroup	ABSTRACT	A substitution group that permits the substitution of the LineDescGroup element when it is a child of the parent of most StandardPrimitive, Outline, Polyline, or Set elements.	16
FillDescGroup	ABSTRACT	A substitution group that permits the substitution of the FillDescGroup element when it is a child of the parent of most StandardPrimitive, Outline, Polyline, or Set elements.	16
PolyStep	ABSTRACT	A substitution group that permits the substitution of the PolyStep element when it is a child of the parent Polyline or Polygon elements.	2
Simple	ABSTRACT	A substitution group that permits the substitution of the Simple element when it is a child of the parent DfxMeasurement, Glyph, or Slot elements.	3
StandardPrimitive	ABSTRACT	A substitution group that permits the substitution of the StandardPrimitive element when it is a child of the parent EntryStandard element.	1
StandardShape	ABSTRACT	A substitution group that permits the substitution of the StandardShape element when it is a child of the parent LayerPad or Pad elements.	2
UserPrimitive	ABSTRACT	A substitution group that permits the substitution of the UserPrimitive element when it is a child of the parent EntryUser element.	1
UserShape	ABSTRACT	A substitution group that permits the substitution or classification of a higher level substitution group. The UserShape element may be used to further classify Feature. In so doing, UserShape can be substituted by a UserPrimitive or UserPrimitiveRef.	0

3.5.1 ColorGroup

The ColorGroup substitution group consists of three elements, Color, ColorRef and ColorTerm, with which it is possible to define a color in the body of the IPC-2581 standard. ColorGroup is used by the FillDesc, Set, Layerfeature, Text, and General (specification) elements.

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
Color	ColorType	The description of a specific color using the three attributes of red, green, and blue.	0-n
ColorRef	ColorRefType	The id of a previously defined 'rgb' color stored in the DictionaryColor.	0-n
ColorTerm	ColorTermType	A color name which is one of an enumerated list of basic color terms.	0-n

There is no default color. If no color is defined anywhere within the XML hierarchy of a given object then the implementing system's color setting shall be used. If different colors are set at different levels within the XML hierarchy of a given object then the value set at the lowest level shall take precedence. For example in the XML hierarchy below the color defined at the lower level, green, shall be used for the Circle.

```

<Set>
  <Color r="255" g="0" b="0"/>
  <Pad>
    <Location x="277.6" y="0"/>
    <Circle diameter="1.0">
      <FillDesc fillProperty="FILL">
        <Color r="0" g="255" b="0"/>
      </FillDesc>
    </Circle>
  </Pad>
</Set>

```

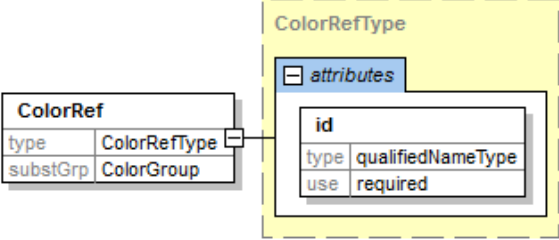
3.5.1.1 Color

Color is defined by three values that represent the red, green and blue components of the composite color. If r, g, and b are all set to 0, the color is black. If all values are 255 then the color is white. The attributes of a Color element are defined as follows:

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
Color	ColorType	The specific description of the color represented by the three attributes.	1-1
r	unsignedByte	Defines the red color intensity as a value between 0 and 255.	1-1
g	unsignedByte	Defines the green color intensity as a value between 0 and 255.	1-1
b	unsignedByte	Defines the blue color intensity as a value between 0 and 255.	1-1

3.5.1.2 ColorRef

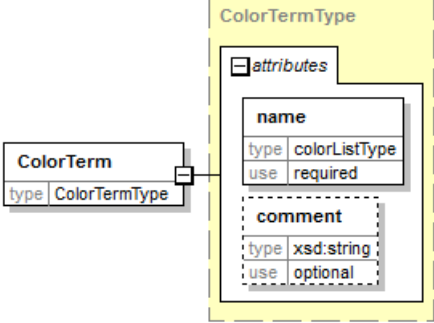
The ColorRef element is used throughout the 2581 file to establish the relationship to a previously defined Color. The ColorRef definition is according to the following characteristics.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
ColorRef	ColorRefType	The specific reference to a predefined color represented by the three attributes contained in a DictionaryColor (see section Error! Reference source not found.)	1-1
id	qualifiedNameType	The qualified description name assigned as an id standard for XML schema color substitution.	1-1

3.5.1.3 ColorTerm

The ColorTerm element is used to describe a color as one of 11 basic color terms, from an enumerated list. It can be used by a customer to describe to a supplier the required color of a material, or physical object, when the precise hue of the color can be left up to the supplier. A 12th enumerated list member OTHER can be used if an additional color term is required, in which case a description must be given in the comment attribute. The ColorTerm definition is according to the following characteristics.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
ColorTerm	ColorTermType	An element to describe a color as one of an enumerated list of basic color terms	1-1
name	colorListType	An enumerated list consisting of basic color terms. The list consists of BLACK, WHITE, RED, GREEN, YELLOW, BLUE, BROWN, ORANGE, PINK, PURPLE, GRAY, OTHER.	1-1
comment	string	Required if OTHER is chosen as the name, or can be used to describe any other information about the color requirement	0-1

3.5.2 Fiducial

The `Fiducial` substitution group consists of four elements that may be used to replace the `Fiducial` element. When the `Fiducial` element is substituted, it **shall** be by either a `BadBoardMark`, `GlobalFiducial`, `GoodPanelMark`, or `LocalFiducial` fiducial type. The `Fiducial` elements contain an `Xform` and a substitution capability to a `StandardShape`. An optional `Pin` attribute allows reference to a component pin.

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
BadBoardMark	FiducialType	A set of Standard Shapes used as an aid to the board assembler by defining those boards in a panel that did not pass inspection or electrical test. The fiducial shape is positioned near each board in the assembly panel array and covered over to signify that the board is defective.	0-n
GlobalFiducial	FiducialType	A set of Standard Shapes used in the description, arrangement or positioning of a group of features on an individual board, assembly, or panel.	0-n
GoodPanelMark	FiducialType	A single Standard Shape used to define a panel where all boards on the panel are good. The fiducial is positioned once on the panel and enables reduction in inspection time.	0-n
LocalFiducial	FiducialType	A set of fiducials (usually a pair) used in the description and arrangement of features related to a specific component on a board, assembly, or panel which aid in the location/positioning process.	0-n

For further details refer to section 8.2.3.12.4 `Fiducial`.

3.5.3 FirmwareGroup

The `FirmwareGroup` substitution group consists of the description element for the firmware that defines the data to be added to a component through the `RefDes` element of a particular `BomItem`. The information may be provided as a `CachedFirmware` element or as a reference to the firmware which has been stored and identified with an "id" in the `DictionaryFirmware`.

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
CachedFirmware	CachedFirmwareType	An element that identifies the firmware needed for a particular component in the Bill of Material. The information is in a <code>hexEncodeBinary</code> format.	0-n
FirmwareRef	FirmwareType	An element that references <code>CachedFirmware</code> stored in the <code>DictionaryFirmware</code> through the callout of the firmware "id"	0-n

3.5.3.1 CachedFirmware

The CachedFirmware element is used to describe firmware that will be contained in the DictionaryFirmware. Refer to section 0 DictionaryFirmware. The details are in accordance to the following characteristics.

CachedFirmware

type	CachedFirmwareType
substGrp	FirmwareGroup

CachedFirmwareType

attributes

hexEncodedBinary

type	xsd:string
use	required

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
CachedFirmware	CashedFirmwareType	The firmware description needed by a particular component that becomes part of the predefined firmware in the DictionaryFirmware.	1-1
hexEncodedBinary	string	An attribute defining the binary code that shall be added to a particular component and which is contained in the DictionaryFirmware.	1-1

3.5.3.2 FirmwareRef

The FirmwareRef element is used throughout the 2581 file to establish the relationship to a previously defined CashedFirmware. The FirmwareRef definition is according to the following characteristics.

FirmwareRef

type	FirmwareRefType
substGrp	FirmwareGroup

FirmwareRefType

attributes

id

type	qualifiedNameType
use	required

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
FirmwareRef	FirmwareRefType	The specific reference to firmware previously defined and contained in the DictionaryFirmware.	1-1
id	qualifiedNameType	The qualified name of CachedFirmware contained in the DictionaryFirmware.	1-1

3.5.4 FontDef

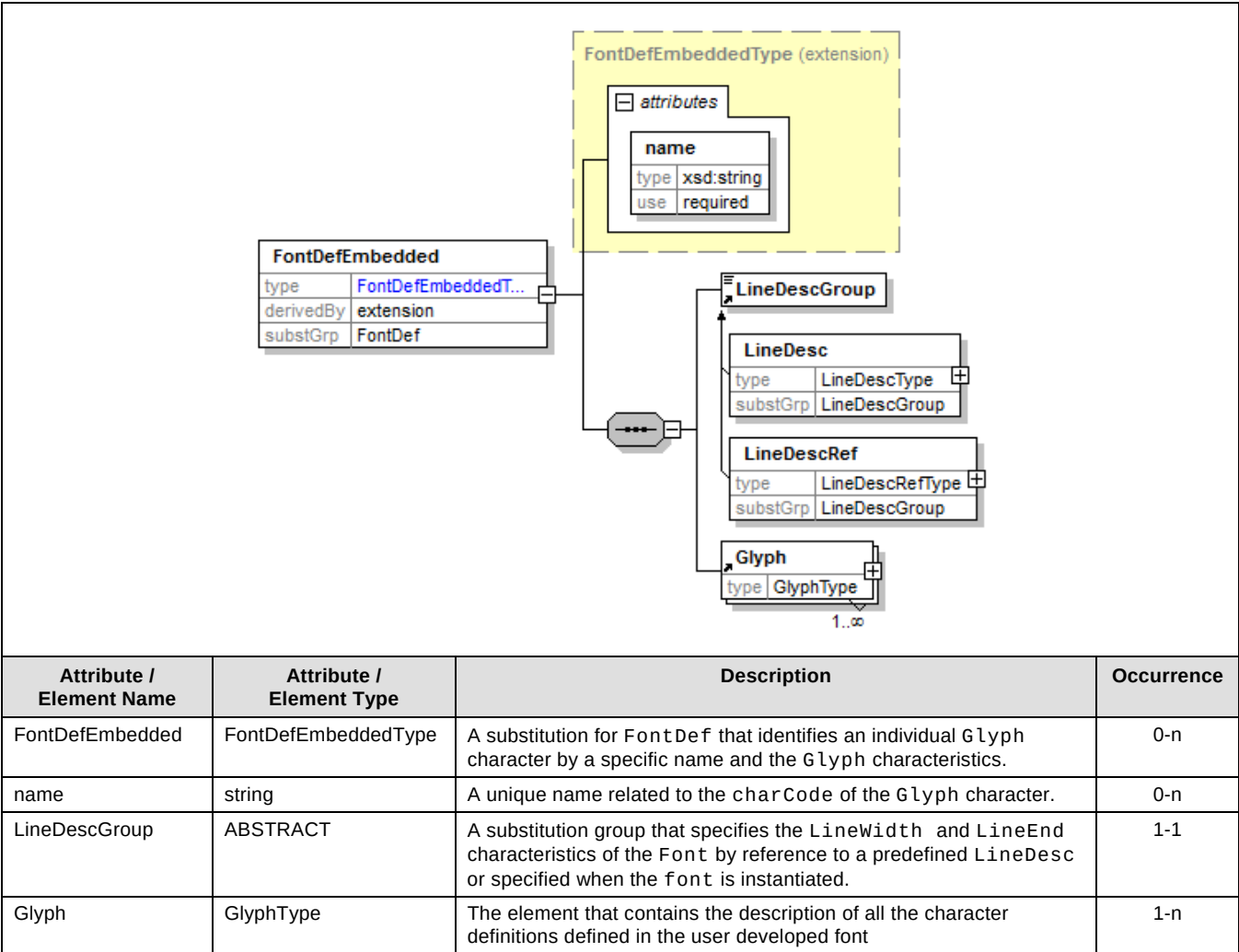
The `FontDef` substitution group consists of the description of a font that is different than the standard Helvetica and which is contained in the `DictionaryFont`. Fonts in the dictionary have an “id” which is called out when a `FontRef` is instantiated. `FontRef` is used by the element `Text`, which is called for in `SilkScreen` and `AssemblyDrawing`. `Text` can also be substituted whenever a `UserPrimitive`, `UserShape` or `Feature` is called for.

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
FontDefEmbedded	FontDefEmbedded Type	A description of a font using individual characters that are defined in accordance with the <code>Glyph</code> element and are contained in the <code>DictionaryFont</code> . Fonts must be predefined and may not be instantiated as part of the data in the file.	0-n
FontDefExternal	FontDefExternal Type	A reference to an external font description through the instantiation of a URN. The font is named and the reference is contained in the <code>DictionaryFont</code> . The appropriate character set is defined by the URN. External Fonts must be predefined and may not be instantiated as part of the data in the file.	0-n

The term "Uniform Resource Name" (URN) refers to the subset of URI that are required to remain globally unique and persistent even when the resource ceases to exist or becomes unavailable. A URI can be further classified as a locator, a name, or both. The term "Uniform Resource Locator" (URL) refers to the subset of URI that identify resources via a representation of their primary access mechanism (i.e., their network "location"), rather than identifying the resource by name or by some other attribute(s) of that resource.

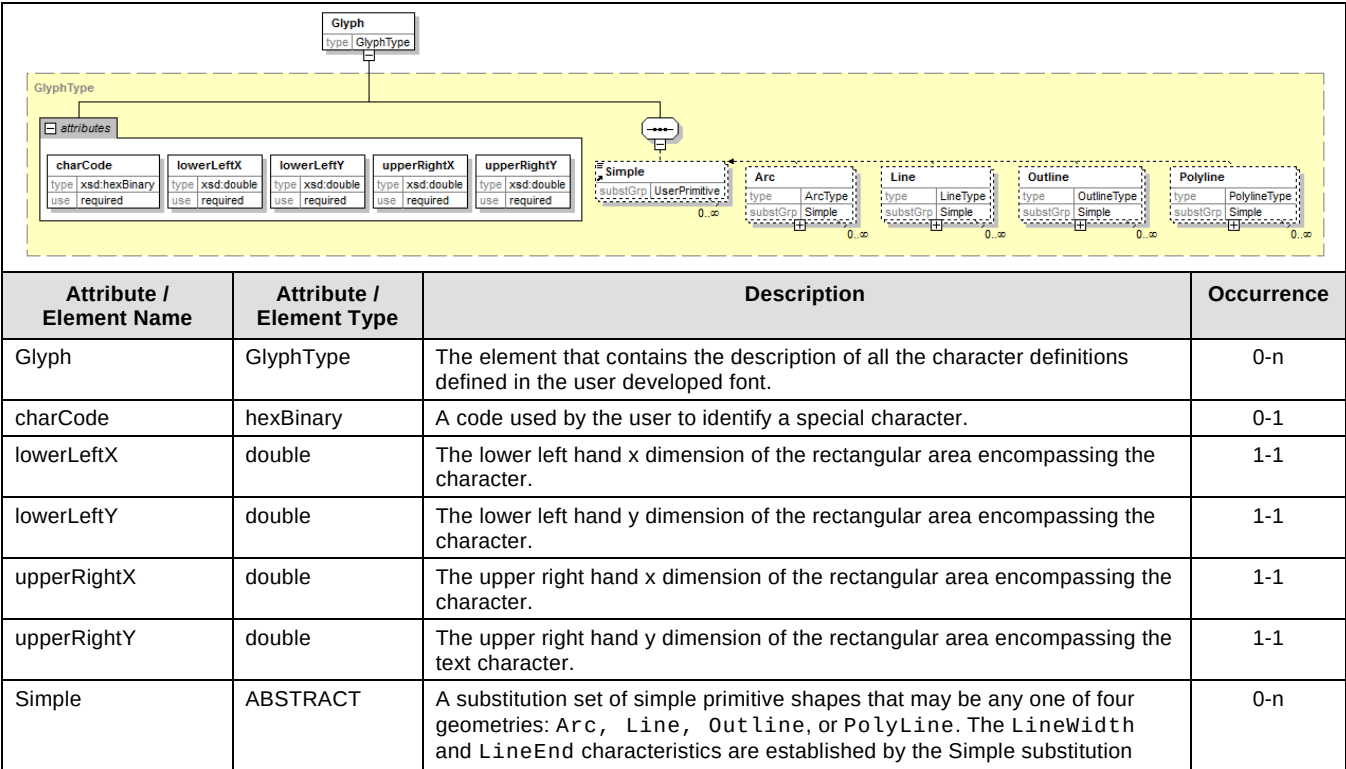
3.5.4.1 FontDefEmbedded

The FontDefEmbedded element is used to capture individual characters and store them in the DictionaryFont. Refer to section 4.8 DictionaryFont.



3.5.4.1.1 Glyph

The Glyph character set is a group of user defined characters that will be reference by the text command in the file. Glyph permits the user to define a special set of characters that need description as a part of the IPC-2581 file. Each character is in a BoundingBox that contains all the line strokes needed to completely define each character in the set. The point of origin is the lower left hand corner of the BoundingBox. The lowerLeftX and lowerLeftY point of origin will be used to position, rotate or mirror image all Glyph characters.



3.5.4.1.2 Glyph combination

The developers of individual Glyph characters are encouraged to consider the manner in which the characters will be used. Since the BoundingBox surrounding the character must entirely encompass the Glyph, it is important to leave room in the BoundingBox so that the spacing between characters is consistent with the character style.

Using individual Glyph characters does not present a problem, however Glyph combinations should match the style of Glyph chosen by the user. Since it is mostly the Text element that instantiates fonts, the Glyph BoundingBox must fit into the Text BoundingBox. This is a simple strategy when all the Glyph characters are of a similar height. In this instance the “Y” dimensional differences between Glyph characters bounding boxes and Text bounding boxes should be identical in order to keep the Glyph characters within the Text box. Under those circumstances, only the spacing between characters needs to be considered.

As an example consider the word simple instantiated in capital letters or lower case. When instantiating a Text string, the Glyph for “SIMPLE” would only require equal bounding boxes in the character height even though the character “l” would have a smaller character width than the character “M”. A different strategy for Glyph development must be used if the Text string were to call for “Simple”. Since character height is different, it is recommended that the Glyph BoundingBox consider its location position in a Text BoundingBox according to the rule that all characters must be inside the Text box.

Figure 4 shows an example of Glyph bounding boxes related to the Text BoundingBox. The characters line up even though they are positioned on the lower y-coordinate. They were designed along a construction line to have this condition occur.

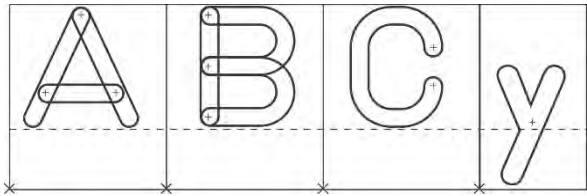


Figure 4 Glyph bounding rectangles to Text bounding box relationships

3.5.4.2 FontDefExternal

The FontDefExternal element is used to capture known font characters and store the reference in the DictionaryFont.

FontDefExternal

type	FontDefExternalType
substGrp	FontDef

FontDefExternalType

attributes

name

type	xsd:string
use	required

urn

type	urn
use	required

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
FontDefExternal	FontDefExternalType	A substitution for FontDef that identifies a known group of characters through their font style.	0-n
name	string	A unique name related to the character set of a known font.	0-n
urn	urn	A specific urn that makes reference to a known font that has the appropriate permission to use the font substitution in a 2581 file.	1-1

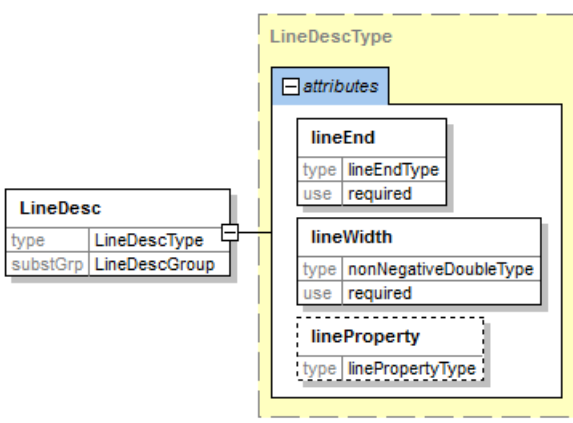
3.5.5 LineDescGroup

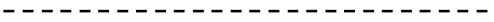
The LineDescGroup substitution specifies the LineDesc or LineDescRef. The LineDesc specifies the LineWidth, LineEnd and LineProperty characteristics of any feature that requires that definition. Line descriptions are a part of the Outline, Polyline and Set element definitions. The substitution is also instantiated by the substitution group Simple which calls for Arc, Line, Outline and Polyline. In addition to UserPrimitive, StandardPrimitive also makes use of the line descriptions when required. The LineDescRef requires an "id" that must be unique within the 2581 file.

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
LineDesc	LineDescType	An element that identifies the LineEnd, LineWidth and LineProperty characteristics	0-n
LineDescRef	LineDescRefType	A reference to a LineDesc that is contained in the DictionaryLineDesc and identified by its unique "id". The units for the dictionary are defined and must be consistent with the units of the CadHeader when referenced from the Ecad section.	0-n

3.5.5.1 LineDesc

The LineDesc element is used throughout the 2581 file to establish the characteristics of lineEnd lineWidth, and lineProperty descriptions. The LineDesc definition is according to the following characteristics.

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
LineDesc	LineDescType	An element that defines the characteristics of a line. The Units are defined when the LineDesc is instantiated; Units is a part of the characteristics when LineDesc is contained in the DictionaryLineDesc.	1-n
lineEnd	lineEndType	A description of the line ends used in any graphic description. The lineEnd description is an enumerated string that may be ROUND SQUARE NONE. The default is ROUND.	1-1

ROUND SQUARE NONE		This diagram illustrates the LineDesc/@lineEnd attribute	
lineWidth	nonNegativeDouble Type	A dimensional characteristic that defines the nominal lineWidth of a Line, Polyline, or Arc element. The dimensions are in the same category as all dimensions contained in the IPC-2581 file.	1-1
lineProperty	linePropertyType	The line property is one of SOLID, DOTTED, DASHED, CENTER, PHANTOM or (solid) ERASE. The default is SOLID.	0-1
SOLID DOTTED DASHED CENTER PHANTOM ERASE		This diagram illustrates the lineProperty attribute. lineProperty attributes require the following rules to define width, end, and spacing characteristics for the line segments, dashes and dots.	
SOLID Lines		Requires lineWidth with lineEnd default as ROUND	
DOTTED Lines		Dot Diameter 1X lineWidth; Space is 2X lineWidth as the default	
DASHED Lines		Dash Length 3X lineWidth; space 3X line Width as the default	
CENTER Lines		Length 6X lineWidth; space is 2X lineWidth; Dot =1X lineWidth	
PHANTOM Lines		Length 6X lineWidth; Space is 2X lineWidth; Dots =1X lineWidth	
ERASE Lines		Requires lineWidth with lineEnd default as NONE	

3.5.5.2 LineDescRef

The LineDescRef element is used throughout the 2581 file to establish the relationship to a previously defined LineDesc. The Units of the predefined LineDesc must match the Units of the Ecad section in which it is instantiated. The LineDescRef definition is according to the following characteristics.

LineDescRef typeLineDescRefType substGrpLineDescGroup LineDescRefType attributes id typequalifiedNameType userequired			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
LineDescRef	LineDescRefType	An element used to reference a previously defined LineDesc, contained in the DictionaryLineDesc, described in section 4.9	1-n
id	qualifiedNameType	The identification of the LineDesc being referenced from the DictionaryLineDesc.	1-1

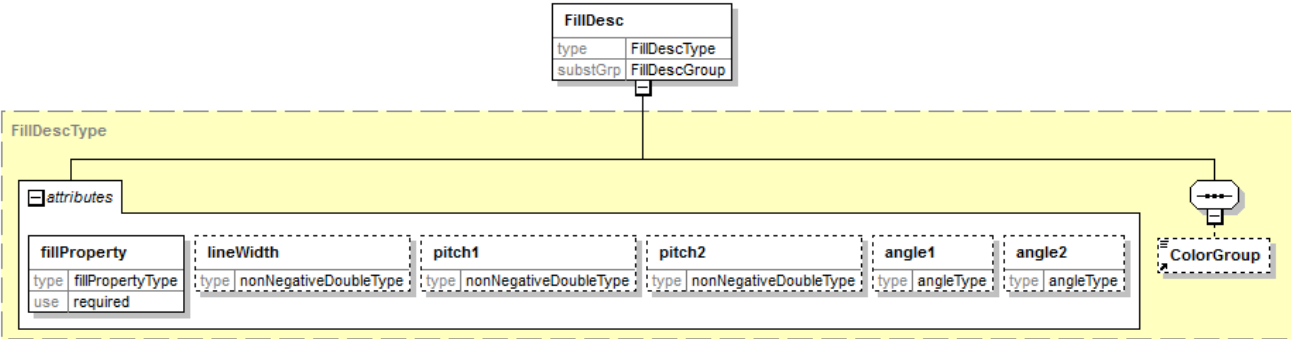
3.5.6 FillDescGroup

The FillDescGroup substitution specifies either FillDesc or FillDescRef. The FillDesc element provides the FillProperties, LineDesc and Color characteristics of any feature that requires that definition. Fill descriptions are a part of the StandardPrimitives, any UserPrimitive closed shape and Set element definitions. The FillDescRef requires an “id” that must be unique within the 2581 file

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
FillDesc	FillDescType	An element that identifies the FillProperties, LineDesc and Color characteristics	0-n
FillDescRef	FillDescRefType	A reference to a FillDesc that is contained in the DictionaryFillDesc and identified by its unique “id”. The units for the dictionary are defined and must be consistent with the units of the CadHeader when referenced from the Ecad section.	0-n

3.5.6.1 FillDesc

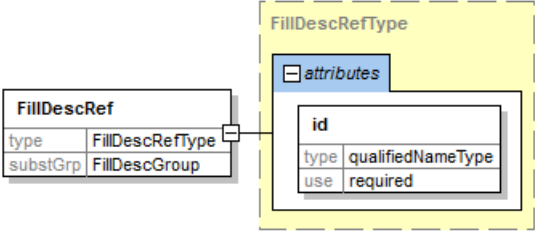
The FillDesc element is used throughout the 2581 file to establish the characteristics of FillProperty and Color descriptions. The FillDesc definition is according to the following characteristics.

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
FillDesc	FillDescType	An element that defines the characteristics of a fill property. The Units are defined when the LineDesc is instantiated; Units is a part of the characteristics when FillDesc is contained in the DictionaryFillDesc.	1-1
fillProperty	fillPropertyType	The texture of the fill is one of HOLLOW, HATCH, MESH, FILL or VOID. The value VOID is reserved for Cutout elements – refer to section 3.4.5 Cutout. The default is FILL.	1-1
lineWidth	nonNegativeDouble Type	The width of the lines used in a HATCH or MESH. (This attribute only applies to the fill types HATCH and MESH.)	0-1
pitch1	nonNegativeDouble Type	The distance between the first set of lines in a HATCH or MESH. The default pitch between the hatch lines is 4 times the lineWidth. (This attribute only applies to the fill types HATCH and MESH.)	0-1
pitch2	nonNegativeDouble Type	The distance between the second set of lines in a MESH. The default pitch between the hatch lines is 4 times the lineWidth. (This attribute only applies to the fill type MESH.)	0-1

angle1	angleType	The angle of a set of lines in a HATCH or a MESH. The angle is measured relative to the x-axis of the local coordinate system. The range of values is limited to between 0 and 180 degrees. The default is 45 degrees. (This attribute only applies to the fill types HATCH and MESH.)	0-1
angle2	angleType	The angle of the second set of lines in a MESH. The angle is measured relative to the x-axis of the local coordinate system. The angle must be between 90 and 180 degrees. The default is 135 degrees. (This attribute only applies to the fill type MESH.)	0-1
ColorGroup	ABSTRACT	An element that is a substitution group which can be used to define a particular Color for the shape being filled. In the absence of this element the default color is black, but this can be overridden by a color set at the parent element, or above, level.	0-1
HOLLOW FILL MESH HATCH VOID     		This diagram illustrates the FillDesc type Attribute. The LineDesc lineProperty is set to SOLID in the example shapes to provide an outline around the shapes that are painted with MESH, HATCH, or VOID. The VOID rectangle feature is within a previously defined rectangle that has been defined with HATCH lines. As shown all of the VOID characteristics are within the boundaries of the larger HATCH rectangle feature	
circle – FILL rectangle – MESH     rectangle – HATCH polygon with VOID		This diagram illustrates the FillDesc type Attribute The LineDesc lineProperty is set to SOLID in the example shapes to provide an outline around the shapes that are painted with MESH, HATCH, or VOID. The VOID within the polygon is completely contained within its boundaries as required.	
 Bottom Pad (Land)		The example may be defined as a UserPrimitive using a Circle fill description with a circular VOID in its center. The User Primitive name may be used in the padstack descriptions.	
The fillProperty VOID means the element shape clears an area with respect to contours. Elements (and references to elements) with fillProperty VOID can appear only as elements in a Set element or a UserSpecial element shape. Elements with fillProperty VOID apply only to the elements that appear before the VOID element in the Set element or in the UserSpecial element. Elements with fillProperty VOID only clear filled contours. A contour is a polygon which is a closed shape whose edges do not cross. The edges of elements with fillProperty VOID do not cross other elements with fillProperty VOID and do not cross the edges of the owning contour element. The order of voids and polygons within Contours and Sets determines their containment order. The outermost polygon comes first. Polygons are before voids that are contained in them. Voids are before polygons that are contained inside them.			

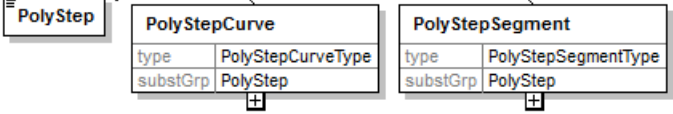
3.5.6.2 FillDescRef

The FillDescRef element is used throughout the 2581 file to establish the relationship to a previously defined FillDesc. The Units of the predefined FillDesc must match the Units of the Ecad section in which it is instantiated. The FillDescRef definition is according to the following characteristics.

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
FillDescRef	FillDescRefType	An element used to reference a previously defined FillDesc, contained in the DictionaryFillDesc.	1-n
id	qualifiedNameType	The identification of the FillDesc being referenced from the DictionaryFillDesc.	1-1

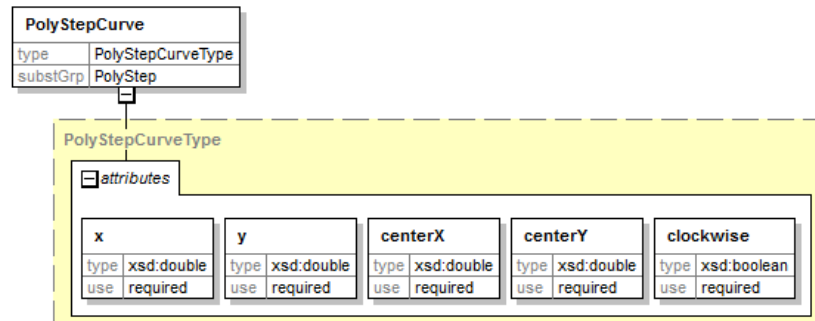
3.5.7 PolyStep

The Polystep substitution consists of defining either a Line or an Arc as the continuation of a Polyline or Polygon description. The location information is interpreted as being the end point to which the curve (Arc), or segment (Line) is drawn. Straight or curved line segments must not cross. The substitution may take place anywhere within the file where the elements Polyline and Polygon occur. This action includes the dictionaries where graphic descriptions are predefined. The Units of measure must be consistent with the Units parameter of the three dictionaries where this substitution can take place; DictionaryStandard, DictionaryUser, and DictionaryFont.

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
PolyStepCurve	PolyStepCurveType	The continuation of the linear description of a Polyline or Polygon if the next portion to be defined is an arc. The end point of the arc is defined as well as the center of rotation	0-n
PolyStepSegment	PolyStepSegmentType	The continuation of the linear description of a Polyline or Polygon if the next portion to be defined is a line segment. The end point of the line is defined.	0-n

3.5.7.1 PolyStepCurve

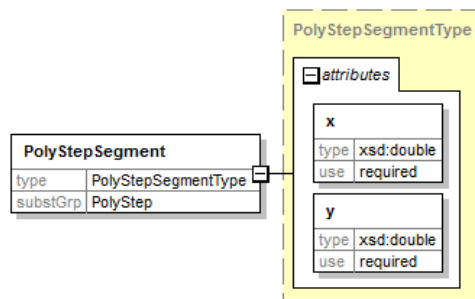
A PolyStepCurve is a continuation of the linear description of a PolyLine or Polygon if the next portion to be defined is an arc. The end point of the arc is defined as well as the center of rotation. The start point of the arc is defined by either a PolyBegin element or the previous PolyStep.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
PolyStepCurve	PolyStepCurveType	The continuation of the linear description of a PolyLine or Polygon if the next portion to be defined is an arc. The end point of the arc is defined as well as the center of location.	1-1
x	Double	The X end point of the curve.	1-1
y	Double	The Y end point of the curve.	1-1
centerX	Double	The X point that is the center of rotation of the curve.	1-1
centerY	Double	The Y point that is the center of rotation of the curve.	1-1
clockwise	Boolean	A required boolean to set the direction of rotation of the curve. If "true" then the direction of rotation is clockwise. If "false" then the direction of rotation is counterclockwise	1-1

3.5.7.2 PolyStepSegment

A PolyStepSegment is a continuation of the linear description of a PolyLine or Polygon if the next portion to be defined is a line segment. It defines the end point of the line. The start point of the line is defined by either a PolyBegin element or the previous PolyStep.



Attribute / Element Name	Attribute / Element Type	Description	Occurrence
PolyStepSegment	PolyStepCurveType	The continuation of the linear description of a PolyLine or Polygon if the next portion to be defined is an arc. The end point of the arc is defined as well as the location of the radius.	1-1
x	Double	The X end point of the line.	1-1
y	Double	The Y end point of the line.	1-1

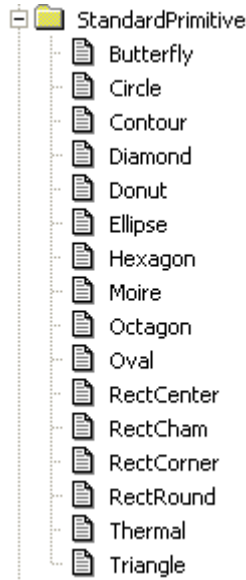
3.5.8 Simple

The Simple substitution consists of base elements Arc, Line, Outline or Polyline. The Simple substitution is called for in the DfxMeasurement, Glyph, and Slot elements. Simple is also identified as a UserPrimitive, UserShape, or Feature and the four elements may be substituted when called for in conjunction with those descriptions. When predefined in the DictionaryUser, or DictionaryFont the Units must match those of the dictionary.

Attribute / Element Name	Attribute / Element Type	Description	Occurrence
Arc	ArcType	Arc elements are curves (defined by three sets of coordinates: startX, startY, endX, endY and centerX, centerY. The width of the arc is applied when the arc is instantiated or predefined.	0-n
Line	LineType	Line elements are individual line segments. The lineWidth and lineEnd conditions are defined when the line is instantiated or predefined. The lineEnd default is ROUND.	0-n
Outline	OutlineType	Outline has Polygon and LineDescGroup as children elements. The characteristics of the Polygon must be a closed shape. The point of origin of the shape is identified as the 0:0 coordinate. The element includes the start of the polygon definition (PolyBegin) and the appropriate number of PolySteps to complete the closed shape. The lineWidth is defined at a time when the Outline is instantiated or predefined.	0-n
Polyline	PolylineType	The Polyline element consists of a series of lines that define a particular grouping configuration. These line segments do NOT result in a closed shape, however they can be pre-defined and re-used as needed. The lineWidth and lineEnd of the Polyline are defined at the time the Polyline is instantiated or predefined.	0-n

3.5.9 StandardPrimitive

There are sixteen Standard Primitives defined in the IPC-2581 structure. Any of the primitive shapes may be a candidate for substitution when StandardPrimitive is called for in the schema. The names of the various shapes indicate their type; each has its attributes that identify the physical requirements. Any StandardPrimitive may be predefined, provided a unique “id”, and contained in the DictionaryStandard. All StandardPrimitive shapes are developed in accordance with their description requirements in the preferred orientation of this standard.

			
Attribute / Element Name	Attribute / Element Type	Description	Occurrence
Butterfly	ButterflyType	A primitive shape (either ROUND or SQUARE) that has two quadrants of the Cartesian coordinate system removed (0 to 90° and 180 to 270°).	0-n
Circle	CircleType	A primitive shape that defines a round object where the circular shape consists of a diameter.	0-n
Contour	ContourType	A closed primitive shape that has as its children a polygon and optional cutout(s). The sequence of connected edges that form the polygon may be straight or circular.	0-n
Diamond	DiamondType	A primitive shape with four equal sides that are extended from its horizontal center to its vertical center. The lines converge into a point both horizontally and vertically. The overall description of the shape is controlled by the width (distance between vertical point) and the height (distance between horizontal point).	0-n
Donut	DonutType	A round, square, hexagon, or octagon shaped primitive consisting of an outer diameter and inner diameter that define physical characteristics of the donut.	0-n
Ellipse	EllipseType	An elliptical primitive shape that follows the standard ellipse characteristics and is defined by a width and height dimension, establishing the overall limits of the feature.	0-n
Hexagon	HexagonType	A six-sided primitive shape with all sides being equal in length and with all angles between adjacent sides being equal. The orientation of the Hexagon is with one of its points facing North. Only the dimension across the points is required.	0-n
Moire	MoireType	A series of circles, each consisting of a smaller diameter than the previous. The details of the Moire is defined by the number of rings, their center line spacing, and the ring width. The pattern may also contain a crosshair representing its point of origin. Restrictions apply so that ringwidth is smaller than ringGap.	0-n

Octagon	OctagonType	An eight-sided primitive shape with all sides being equal in length and with all angles between adjacent sides being equal. The orientation of the Octagon is with one of its points facing north. Only the dimension across the points is required.	0-n
Oval	OvalType	A rectangular primitive shape with a complete radius (180° arc) at each end. The limits of the feature are controlled by the length and width of the oval across the outer extremities.	0-n
RectCenter	RectCenterType	The characteristics of a rectangle defined by a width and height dimension consistent with a horizontal position on the Cartesian coordinate system. The center point is the point of origin and is used to locate the RectCenter. A "square" is a RectCenter with the width and height equal.	0-n
RectCham	RectChamType	A rectangle with one or more corners chamfered. The user has the option to define any of the corners as containing the chamfer as well as the chamfered dimensions. All chamfers (or opportunities for chamfers) must be identical in size.	0-n
RectCorner	RectCornerType	A constraining rectangular area (bounding box) that describes a rectangle consistent with a horizontal position on the Cartesian coordinate system. The point of origin is the lower left corner. A Square positioned by its corners is a RectCorner that is defined by having the X and Y offset be equal.	0-n
RectRound	RectRoundType	A rectangle with one or more corners rounded. The user has the option to define any of the corners as containing the radius as well as the radiused dimensions. All corners (or opportunities for corners) must be identical in size.	0-n
Thermal	ThermalType	A primitive shape consisting either of ROUND, SQUARE, HEXAGON, or OCTAGON configuration that historically defines the removal of material from a plane or conductive fill area in accordance to the shape attribute description.	0-n
Triangle	TriangleType	A primitive isosceles triangular shape that has two equal sides and a base. The feature is defined by a base and height dimension.	0-n

3.5.9.1 Butterfly

A Butterfly is a StandardPrimitive shape that may have the external periphery be either round or square with two quadrants of the Cartesian coordinate system removed (0 to 90° and 180 to 270°). The round shape is defined by its diameter; the square shape is defined by an equal side dimension. The Butterfly is positioned by its point of origin, which is at the center of the Butterfly.