JEDEC PUBLICATION

Part Model Guidelines for Electronic-Device Packages – XML Requirements

JEP30E

(Revision of JEP30D, August 2023)

August 2024

JEDEC SOLID STATE TECHNOLOGY ASSOCIATION

NOTICE

JEDEC standards and publications contain material that has been prepared, reviewed, and approved through the JEDEC Board of Directors level and subsequently reviewed and approved by the JEDEC legal counsel.

JEDEC 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 use by those other than JEDEC members, whether the standard is to be used either domestically or internationally.

JEDEC standards and publications are adopted without regard to whether or not their adoption may involve patents or articles, materials, or processes. By such action JEDEC does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the JEDEC standards or publications.

The information included in JEDEC standards and publications represents a sound approach to product specification and application, principally from the solid state device manufacturer viewpoint. Within the JEDEC organization there are procedures whereby a JEDEC standard or publication may be further processed and ultimately become an ANSI standard.

No claims to be in conformance with this standard may be made unless all requirements stated in the standard are met.

Inquiries, comments, and suggestions relative to the content of this JEDEC standard or publication should be addressed to JEDEC at the address below, or refer to www.jedec.org under Standards and Documents for alternative contact information.

Published by
©JEDEC Solid State Technology Association 2024
3103 10th Street North
Suite 240S
Arlington, VA 22201

.

PRICE: Contact JEDEC

Printed in the U.S.A.

DO NOT VIOLATE
THE
LAW!

For information, contact:

JEDEC Solid State Technology Association
3103 10th Street North
Suite 240S
Arlington, VA 22201
https://www.jedec.org/contact

This page intentionally left blank.

PART MODEL GUIDELINE FOR ELECTRONIC-DEVICE PACKAGES - XML REQUIREMENTS

Contents

PART MODEL GUIDELINE FOR ELECTRONIC-DEVICE PACKAGES - XML REQUIREMENTS

Contents (cont'd)

Tables

Figures

Foreword

This standard establishes requirements for the generation of electronic digital part models for electrical and electronic products for the JEDEC Solid State Technology Association. The standard defines a digital secured framework, that enables component manufacturers to submit the respective part content in a secured software consumable format that can be digitally signed off, providing Trust and Traceability to its original source. This standard specifically focuses on the parental structure, under which several sub-sections defined by their respective standards, can be contained and linked together within the Part Model parent structure. The requirements herein are intended to ensure that such part information is presented in as uniform a manner as practicable.

Where applicable, references are made to standardization documents of the following organizations throughout the Part Model hierarchy:

American Mathematical Society.
American National Standards Institute, Inc. (ANSI)
Association Connecting Electronics Industries (IPC)
Chips Alliance
Electronics Components Industry Association (ECIA)
Electronic Industries Association (EIA)
High Definition Multimedia Interface (HDMI)
Institute of Electrical and Electronics Engineers (IEEE)
International Electrotechnical Commission (IEC)
International Committee for Information Technology Standards (INCITS)
MIPI Alliance (mipi)
National Institute of Standards and Technology (NIST)
Optical Internet Working Forum (OIF)
Open Compute Project (OCP)
Peripheral Component Interconnect Special Interest Group (PCI-SIG)
SD Association (SDA)
System Management Interface Forum (SMIF)
The American Society of Mechanical Engineers (ASME)
Universal Chiplet Interconnect Express (UCIe)
Universal Serial Bus Implementers Forum (USB-IF)

These JEP30 standards represent a standard XML structure to enable Component Manufacturers to provide part data to their customers, utilizing these definitions herein in a digital model.

Introduction

This standard provides a structured part identity framework that enables the individual technical to be connected via their sub-sectional schemas as defined in the array of JEP30 related publications.

This page intentionally left blank.

PART MODEL GUIDELINE FOR ELECTRONIC-DEVICE PACKAGES - XML REQUIREMENTS

(From JEDEC Board Ballot JCB-24-29, JCB-23-33 formulated under the cognizance of the JC-11 Committee on Mechanical Standardization.)

1 Scope

This standard establishes the requirements for exchanging part data between part manufacturers and their customers for electrical and electronic products. This standard applies to all forms of electronic parts. It covers several sub-sections such as assembly process classification, electrical, environmental including material deceleration, physical, thermal, and supply chain data along with materials and substances that may be present in the supplied product or sub-products. This Guideline specifically focuses on the parental structure, under which several sub-sections listed above, can be contained and linked together within the Part Model parent structure.

All releases of any sub-schema must be under the umbrella of the Part model Schema to ensure that the Part Model schema is referencing the correct version of the sub-schema. This will enable the sub-schema to connect into the identification of the Manufacturer Part Number and the Manufacturer of the Part.

Example of how this standard can be used, is in defining the part in sufficient detail to enable process efficiencies during the part and product life cycles, i.e., design, purchasing, manufacturing, quality control, test, etc. This release includes the standardization of part data within a structured framework, to provide this support to the industry. The standard is designed to be scalable insofar that it should cover as many components as possible that are available in the market. It should also be scalable to encompass the emergence of new packages in the future.

Although this standard is considered to have international standardization implications, a complete comparison between the JEDEC standard and the international documents has not been made.

This standard applies to business-to-business transactions. The standard is not a compliance guide. As revisions to various Regulatory Regulations are released, this standard will be updated.

1.1 Purpose

This standard is intended to benefit part manufacturers and their customers by providing consistency and efficiency to the transfer of part data from part manufacturer to customers. It establishes standard electronic data exchange formats that will facilitate and improve data transfer along the entire global supply chain, at every stage in the product life cycle. A key aspect therefore is the structure of the content that is contained in this format, which the committee believes should be based on the following two principals:

1) Data that is required to be consumed by software tools, and

1.1 Purpose (cont'd)

2) Data that is not required to be consumed by software tools but is provided for informational purpose.

2 Applicable Documents

The following documents form a part of this standard to the extent specified herein. The revision of the document in effect at the time of solicitation shall take precedence.

2.1 JEDEC (www.jedec.org)

JEP30-A100, Part Model Assembly Process Classification Guidelines for Electronic-Device Packages – XML Requirements

JEP30-E100, Part Model Electrical Guidelines for Electronic-Device Packages – XML Requirements

JEP30-P100, Part Model Package Guidelines for Electronic-Device Packages – XML Requirements

JEP30-S100, Part Model Supply Chain Guidelines for Electronic-Device Packages – XML Requirements

JEP30-T100, Part Model Thermal Guidelines for Electronic-Device Packages – XML Requirements

JESD99, Terms, Definitions, and Letter Symbols for Microelectronic Devices

JEP95, JEDEC Registered and Standard Outlines for Solid State Products

2.1.1 JEDEC Part Model Schema and Sub-Schemas

JEP30-10, Part Model Schema

JEP30-A101, Part Model Assembly Process Classification Schema

JEP30-E101, Part Model Electrical Schema

JEP30-P101, Part Model Package Schema

JEP30-S101, Part Model Supply Chain Schema

JEP30-T101, Part Model Thermal Schema

JEP30-D10, *Part Model Schema Types Dictionary* (Required to support the Part Model Schema and each of its sectional sub-schemas.)

2.2 IEC (std.iec.ch)

IEC 62474, Material Declaration for Products of and for the Electrotechnical Industry

2.3 IPC (www.ipc.org)

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

2.4 ISO/IEC (www.iec.ch)

IPC/IEC 6523, Information technology – Structure for the identification of organizations and organization parts

2.5 American Mathematical Society

"Short Math Guide for LaTeX", Version 1.09 (2002-03-22), currently available at

http://www.ams.org/tex/short-math-guide.html

3 Terms and Definitions

The following terms and definitions are applicable to this XML Schema.

All definitions and terms associated with any sub-section schema are in accordance with the Guidelines for that sub-schema.

All other definitions and terms necessary to define the Part Model parent schema, are defined in this document.

3.1 Data Terms and Definitions

device: A piece of equipment, a mechanism, or another entity designed to serve a special purpose or perform a special function.

NOTE 1 In JEDEC documents, the word "device" is often used as an abbreviated reference to the type or types of solid-state devices that are within the scope of those documents. Context could indicate otherwise; e.g., in the phrase "the device used to hold the device under test", the first usage of the word "device" refers to a mechanism; the second to a solid-state device.

NOTE 2 Contrast with "component".

Component: A constituent part.

NOTE 1 Examples include source and drain regions as components of transistors, terminal frames and dice/dies as components of packaged integrated circuits, resistors and integrated circuits as components of printed circuit boards, motherboards as components of computers, LCD screens as components of monitors, ac and dc components of complex waveforms, and loops and algorithms as components of software programs.

NOTE 2 Unless the context identifies the thing of which a component is a part, a descriptive prepositional phrase identifying the thing should follow the word "component".

3.1 Data Terms and Definitions (cont'd)

Part: As used within the context of the Part Model, a Part is the same as a device. Each Part has an identity. But there are many variations of these identities; hence the following definitions will clarify these variations.

NOTE A **Part** is a unique device that is manufactured by a specific manufacturer and has been assigned a specific manufacturer part number (MPN). It can only be produced by one manufacturer.

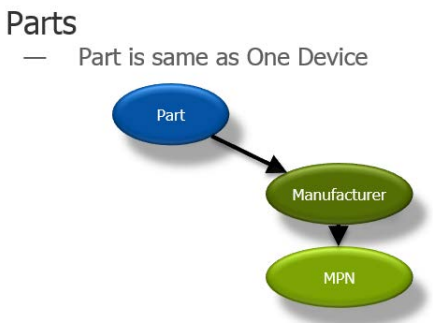


Figure 1 - Part

Manufacturer Alternative Part Numbers: When the part manufacturer ships these Parts in different supply forms, such as a reel, tube, box, or tray, the part manufacturer will assign different manufacturer part numbers to this Part to represent the part in a specific supply form. These numbers are defined as "**Manufacturer Alternative Part Numbers**". All Alternative Manufacturer Part Numbers that represent the same Part, in different Supply Forms, must have EXACTLY the same specification, because they all represent exactly the same device.

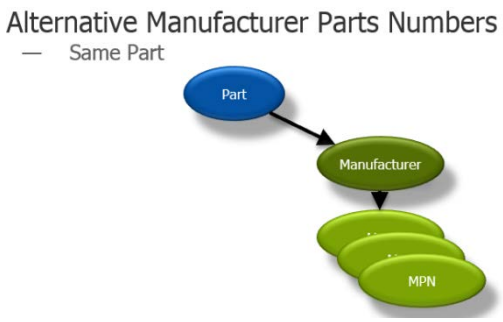


Figure 2 - Alternative Manufacturer Part Numbers

Alternative Supplier Part Numbers: When a Supplier distributes Parts or resells Parts, or acts as an Agent for a Manufacturer, the supplier usually assigns a Supplier Part Number to the Part. Just like the Manufacturer, the Supplier can have many different Supplier Part Numbers for the same Part. In addition, there can be many different Suppliers that can distribute this Part from the part manufacturer. These numbers are defined as "**Alternative Supplier Part Numbers**".

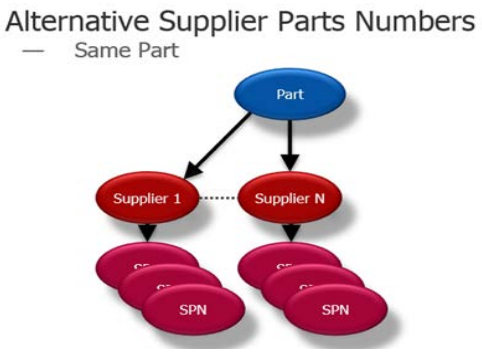


Figure 3 - Alternative Suppliers Part Numbers

Alternative Parts: To reduce risk of Product supply, multiple sources of equivalent Parts are desired, which is typically done via an Approved Manufacturers List (AML) that accompanies a Bill of Materials (BOM) for a given Product design. A Customer Part Number (CPN) or sometimes defined as an Internal Part Number (IPN) is assigned to a group of Parts that can be placed on the same Reference Designator in a Product Design.

3.1 Data Terms and Definitions (cont'd)

The Specification of the CPN/IPN represents the envelope of the specification of each **Part** that is contained within that group. All Parts connected to the same Customer / Internal Part Number are interchangeable in order for the end Product to still function as normal. These Parts are now referred to as "**Alternative Parts**".

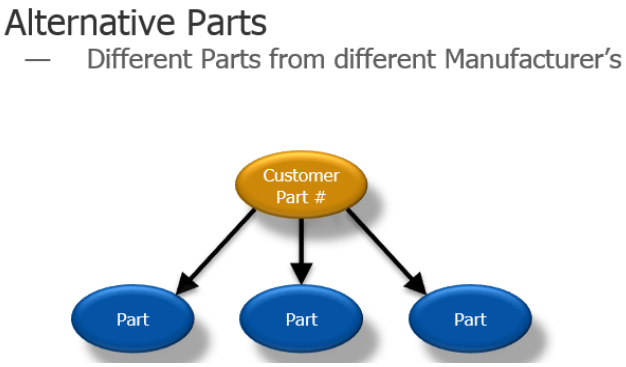


Figure 4 - Alternative Parts

EDA Part: An "**EDA Part**" represents the Design representation of a Part. It includes the following: 1) Terminal List, 2) Footprint, and 3) Schematic Symbol.

NOTE It does not include the values that can be propagated to the Schematic Symbol, as these can be inherited from the CPN/IPN as opposed to be inherited directly from the Part.

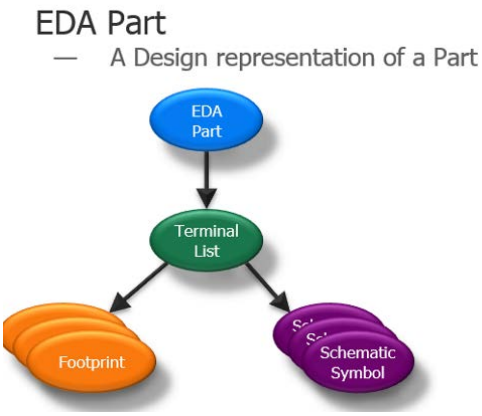


Figure 5 - EDA Part

An **EDA Part** must be connected to a **Part**. There can be many EDA Parts connected to the same Part, however all EDA Parts that are connected to the same Part must be interchangeable.

3.1 Data Terms and Definitions (cont'd)

NOTE 1 The same EDA Part can be connected to many different Parts.

NOTE 2 The same EDA Part can be reused by many different Parts, so long as the specification of these Parts maintain some commonality from the perspective of Symbol and the Footprint. This is possible since the Part specific values are excluded from the EDA Part. For example, all Resistors of different Values, Wattage and Tolerance can inherit the same Symbol. All Parts contained within the same Package can be mapped to the same Footprint.

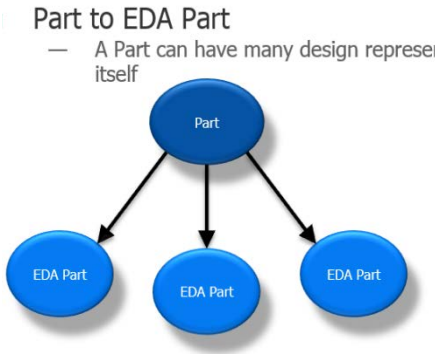


Figure 6 - Part Mapped to Multiple EDA Parts

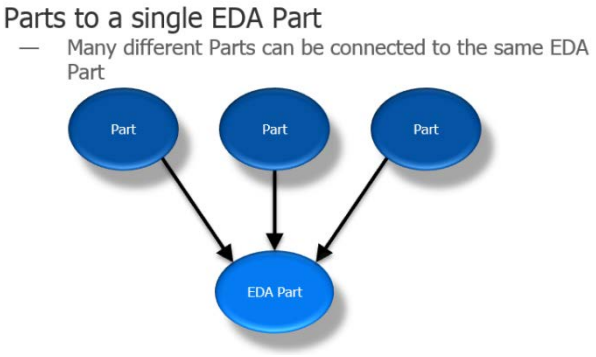


Figure 7 - Parts Relationship to a Single EDA Part

Part Model: A Part Model is a data representation described in an XML file that conforms to the rules and structure of the Part Model XML Schema.

NOTE 1 Companies who use the Part Model XML Files and claim compliance to JEDEC, must ensure that their Part Model XML file conforms to the specific released version of the Part Model XML Schema released by JEDEC.

NOTE 2 Section 4 will define the outline of the structure of the Part Model XML Schema. Specific components of the XML Schema and their hierarchy are specifically controlled by the Standards Committee who retain the expertise for these structures.

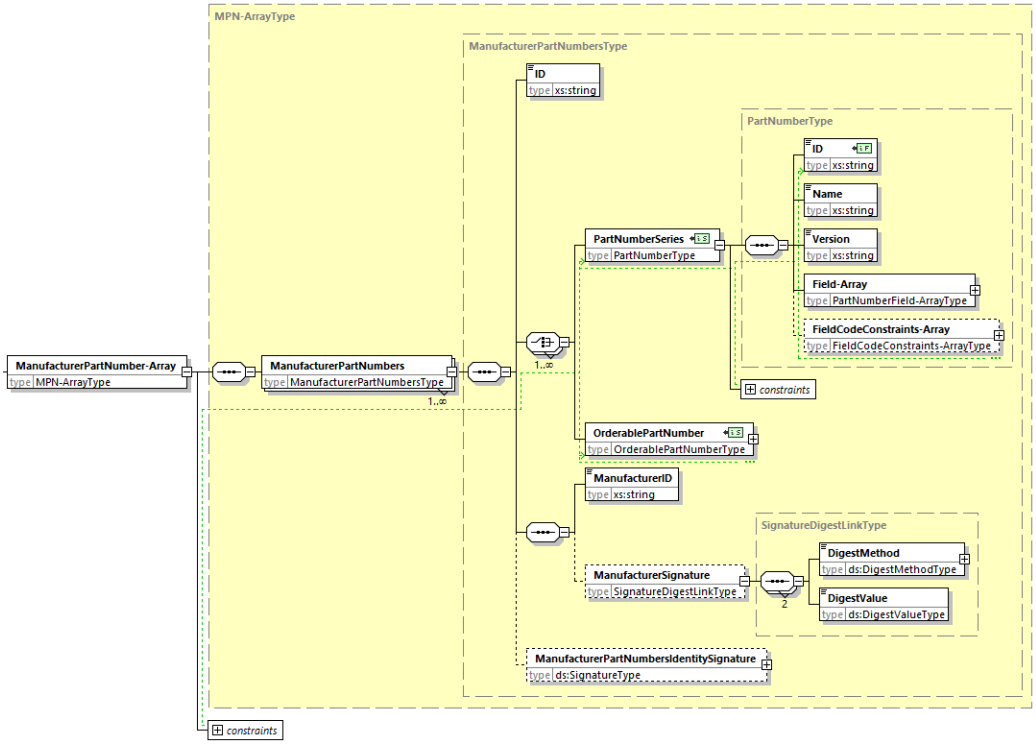
Array: The use of Arrays within the XML Schema enables a list of items to be captured under the array.

Family: The use of Family within the XML Schema means that the object can represent a family of parts each with identical characteristics. The family sections are typically referenced from another branch (typically MPN). This provided the ability of linking a large set of data (e.g. 100 data elements) that is represented once, to many other records (e.g. a 1000 MPN records) by means of an ID, instead of having to duplicate the 100 data elements a 1000 times.

3.2 XML Schema Key Terms and Definitions

XML structure is simply a set of rules to define how to organize text into a standard structure within a file. This structure is best represented by images. The key definition specified in Table 1 below helps you to understand how to interpret the images used to document the schema throughout this document. Software tools that provide schema viewing capability can enable you to see this structure as well and walk up and down the structured representation of the data. There is significant detailed information on XML that can be found online.

3.2.1 Sample XML Schema Structure

path	**PartModel/ManufacturerPartNumber-Array**
diagram	
type	**MPN-ArrayType, ManufacturerPartNumbersType, PartNumberType, PartNumberField-ArrayType, FieldCodeConstraints-ArrayType, OrderablePartNumber-ArrayType, JEP30-D10:SignatureDigestLinkType, ds:DigestMethodType, ds:DigestValue, SignatureType.**

The *ManufacturerPartNumber-Array* is the top-level element shown in this diagram. However, you can see a horizontal line entering the element from the left-hand side, indicating that this Element has a parent. This is also denoted by the *path* and is shown here in the top line of the table as *PartModel/ManufacturerPartNumber-Array*.

The *ManufacturerPartNumber-Array* element has a *type* called *MPN-ArrayType*. This element has a structure below it, as shown by the boxes to the right-hand side of the element.

An Element is a named structural entity in the XML. This would translate inside an xml file to

3.2.1 Sample XML Schema Structure (cont'd)

```
<ManufacturerPartNumber-Array>
        <ID>MPNs ID 1</ID>
        <PartNumberSeries>
            <…….>
        < PartNumberSeries >
        < ManufacturerID>Manufacturer ID 1</ManufacturerID>
    </ManufacturerPartNumber-Array>
```

where … can represent any number of additional elements contained within the *PartNumberSeries* element. The elements to the right of *ManufacturerPartNumber-Array* are its children such as *PartNumberSeries*. The element to the left of *Field-Array* is its parent which is the *PartNumberSeries* element.

ID fields throughout the schema and sub-schemas are used to allow one section of the XML to reference another section of the XML. They do not have any relevance or meaning outside of the XML.

The *Field-Array* has a solid line meaning that it is a required element, however, it is only required if its parent is also present in the XML. This means that if any data under the *PartNumberSeries* element is populated, then the following data elements must be present: - *ID*, *Name*, *Version*, and *Field-Array*. The *FieldCodeConstraints-Array* is not mandatory, since that is an optional element as denoted by the dotted line surrounding the element. The XML is still considered valid in that is complies with the schema when optional elements are excluded, however, if the information is applicable it should be populated to benefit the User of the data. In this example, if there were no constraints that needed to be applied to the *PartNumberSeries* element, then there would be no need to specify any data under this *FieldCodeConstraints-Array* branch.

The *path* shown in the top row of the tables, is the path within the xml schema from the parent to the respective element in the schema. Each *path* is unique within the schema. Paths have limitations, insofar that there can be no spaces in the path name, and no section of the path can begin with a numerical. Non-alphanumeric characters are not-allowed except for "-" and underscores.

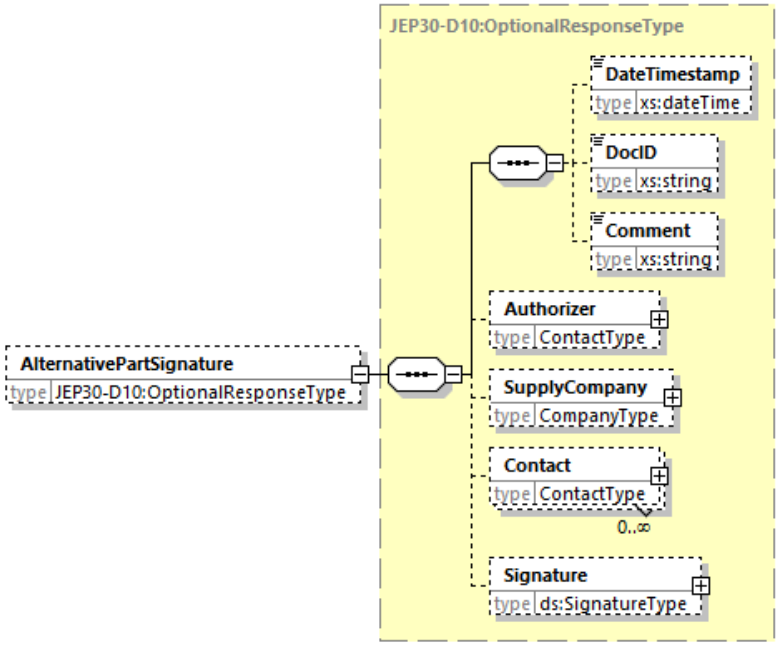
The *diagram* shows a graphical representation of the structure of the data elements within the schema. XML files that conform to this schema will have its data structured in accordance with this diagram.

Throughout the schema, there are various digital signatures such as in this example, *ManufacturerPartNumbersIdentitySignature*, which has a *type ds:SignatureType*. This is to provide the capability to apply a digital signature to all the content that falls in under a single representation of *ManufacturerPartNumbers* element. All parts listed via the unbounded element of *PartNumberSeries* or the unbounded element of *OrderablePartNumber* should belong to the same group of Manufacturers Parts Numbers (typically what is represented on a datasheet) for which this PartModel and its technical content is intended to represent.

Since the Manufacturers Parts Numbers come from a specific Manufacturer, whose details are represented in another array, a reference to that manufacturer is made via the *ManufacturerID*, which is also accompanied by an optional *ManufacturerSignature* link. This signature has a type called *SignatureDigestLinkType* and its purpose is to bring into this array the *DigestMethod* and the *DigestValue* from the *Manufacturer-Array* that accompanies that *ManufacturerID*.

3.2.1 Sample XML Schema Structure (cont'd)

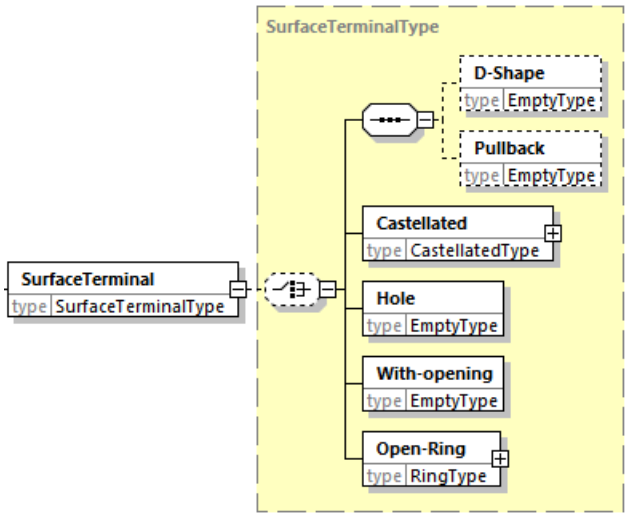
While digital signatures are optional, the provision of digital signatures through the document helps the consumer validate upon receipt that the PartModel contents are not maliciously altered or tampered with. Since the PartModel structure spans so many distinct technical and marketing structures, the application of digital signatures throughout the schema offers consumers of the PartModel confidence that each top-level sections are valid at the time of consolidating into the overall and final PartModel file. These signatures also provide efficient methods for software tools to effectively join data from multiple different Part Model files into a single Part Model file for the same set of *ManufacturerPartNumbers*.

sample path	**PartModel/SupplyChainSection/SupplyChain-Array/AlternativePart-Array/AlternativePart/AlternativePartSignature**
diagram	
type	**JEP30-D10:OptionalResponseType, ContactType, CompanyType, ds:SignatureType.**

In some cases where a supplier accumulates many sections of the PartModel from different companies, the digital signature may not be adequate. In this scenario, a new *type JEP30-D10:OptionalResponseType* enables the digital signatures to be accompanied with some additional details such as authorizer, supply company and contacts for the submission of that document (or portion of the PartModel) before it is merged to form a more complete Part Model file.

An example of this could be where the Die is manufactured by a foundry who produces a file containing electrical representation data for the part. An OSAT may have outsourced the design of a package that will host that die. The OSAT or their outsourced design house may provide the package details of the PartModel. The component manufacturers may now want to merge these two sources of partial part models and join them together to provide a complete part model to their customer.

3.2.2 Sample XML Schema Types

path	**PartModel/PackageSection/Package-Array/Package/PackageTerminalCode/SurfaceTerminal**
diagram	
type	**SurfaceTerminalType, CastellatedType, RingType.**

Many elements will contain a type. Simple types convey valuable information about what you are allowed to put under the element, such as *xs:string*, *xs:Boolean*, *xs:integer*, *xs:decimal*, *xs:dateTime*, *xs:date* etc. Elements with types beginning with xs: are standard available types that limits the format of the data, and hence improves the integrity of the data within the xml. Elements that have types excluding xs: represent a structure of data for that element such as an enumerated list of values, or another hierarchal structure below that element.

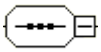
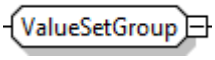
An element defined as a string expresses that a free form string must be defined with that element, such as the name of “SMT Ceramic Chip Capacitor” under the *PartNumberSeries* element would be defined as

```
<PartNumberSeries >
        <ID>Part Construct 1</ID>
        <Name>SMT Ceramic Chip Capacitor</Name>
        ….
```

However, the type called *EmptyType* such as seen on the elements *D-Shape*, *Pullback*, *Hole*, and *WithOpening* simply defines the presence of that feature on the *SurfaceTerminal*. Having the type called *EmptyType* as opposed to having no type defined, allows for better recognition of the data by software tools.

3.2.2 Sample XML Schema Types (cont'd)

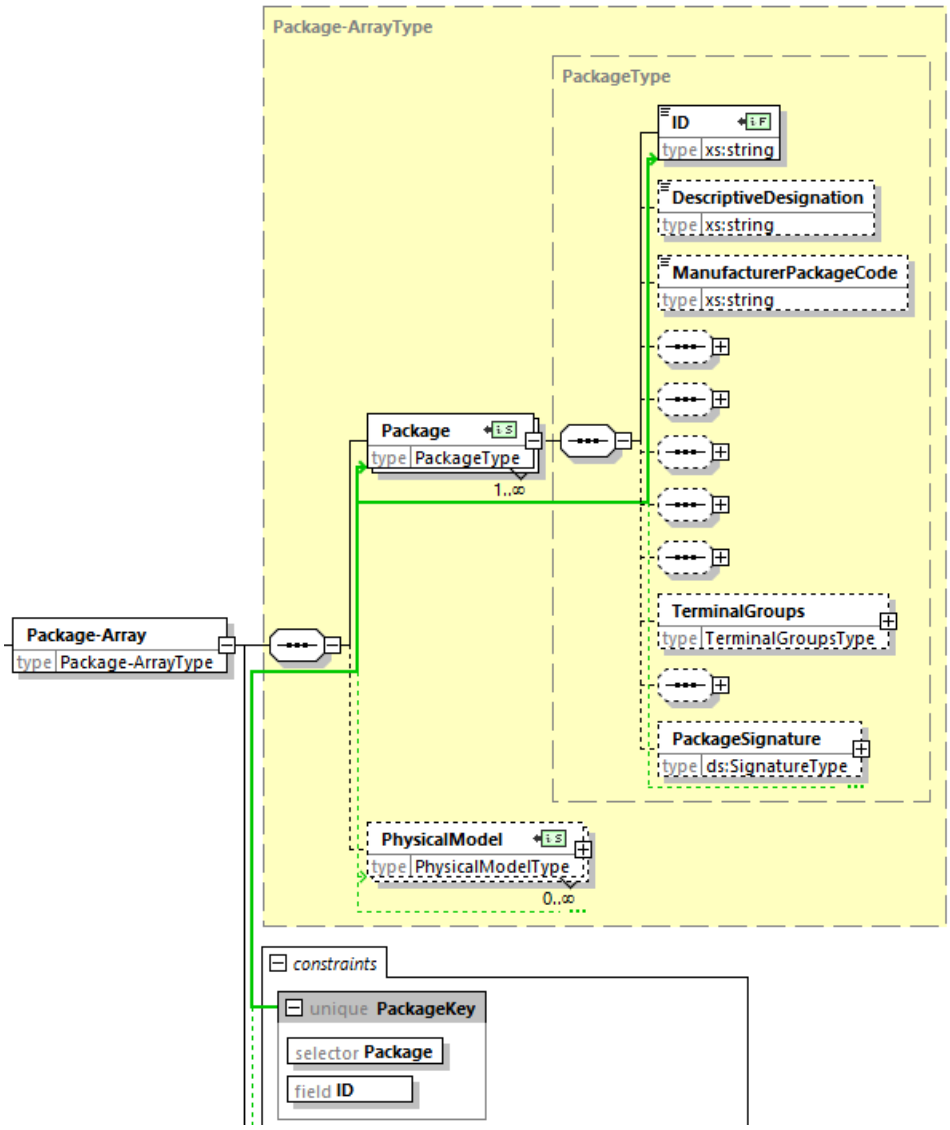
Table 1 - XML Schema Key Definition

Graphic Descriptor	Type	Description
	CARDINALITY	SHOWN BELOW AN ELEMENT. THE CARDINALITY DEFINES IF THE ELEMENT CAN BE REPEATED OR NOT. IN THIS CASE THERE MUST BE AT LEAST 1 AND AS MANY AS NEEDED. WHEN 0 IS THE FIRST VALUE IT MEANS THAT THIS ELEMENT IS OPTIONAL AND CAN BE EXCLUDED. WHEN THERE IS NO CARDINALITY THERE SHOULD ONLY BE ONE ELEMENT OF THAT TYPE UNDER THE PARENT ELEMENT.
	SEQUENCE	ANY OF THE ELEMENTS TO THE RIGHT OF THE SEQUENCE CAN BE ADDED UNDER THE ELEMENT TO THE LEFT OF THE NODE. A SEQUENCE FURTHER RETRAINS THE XML BY REQUIRING THE ELEMENT BE PROVIDED IN THE ORDER SPECIFIED BY THE SCHEMA WHICH IS ALSO THE ORDER SHOWN IN THE IMAGES.
	CHOICE	ONLY A SINGLE ELEMENT TO THE RIGHT OF THE CHOICE CAN BE ADDED UNDER THE ELEMENT TO THE LEFT OF THE NODE.
	ALL	ALL ELEMENTS TO THE RIGHT OF THE NODE MUST BE ADDED UNDER THE ELEMENT TO THE LEFT OF THE NODE, UNLESS THE ELEMENT TO THE RIGHT IS OPTIONAL. THE NODE "ALL" REDUCES THE CONSTRAINTS ENFORCED BY SEQUENCE, INSOFAR THAT THERE IS NO RESTRICTION ON THE ORDER OF THE ELEMENT WITHIN THE XML FILE.
	GROUP	THE GROUP ELEMENT IS USED TO DEFINE A GROUP OF ELEMENTS TO BE USED IN COMPLEX TYPE DEFINITIONS

3.2.3 Sample XML Schema Unique keys and Key Refs

In addition to defining the required structure of an XML document, schemas provide mechanisms for associating identifiers to sections of the document, and for cross-referencing those identifiers from other parts of it. Identifiers are enforced to be unique, and references are validated to point to identifiers that exist.

A *unique* key constraint is the association of an identifier element to a section of the document within a scope. The following diagram shows how the schema declares that, within a *Package-Array*, a single *Package* is identified by the value of its *ID* element:

path	**PartModel/PackageSection/Package-Array/Package**
diagram	
type	**Package-ArrayType, PackageType, TerminalGroupsType, PhysicalModelType**

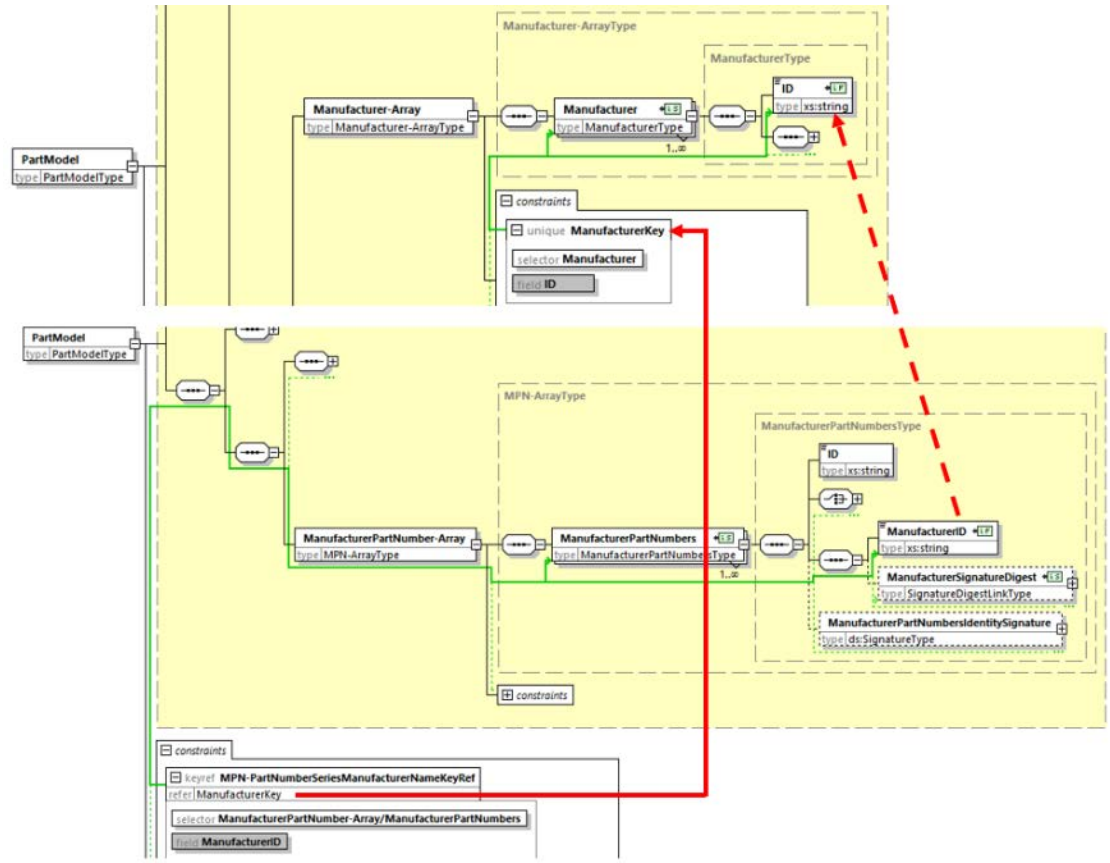
The *PackageKey* constraint enforces that no different packages from the same *Package-Array* can have the same *ID*.

3.2.3 Sample XML Schema Unique keys and Key Refs (cont'd)

A *keyref constraint* is a declaration that, within a scope, a value refers to another element in the document, which has an associated key to it.

Keyrefs must be declared at a level under which both the key and the reference to it are visible. Therefore, many keyrefs end up being declared under *PartModel*, the top-level element.

The following diagram, in which most elements have been omitted for clarity, shows how a specific *Manufacturer* within the *Manufacturer-Array* is linked to the set of parts under the *ManufacturerPartNumbers* element.

path	**PartModel**
diagram	
type	**PartModelType, MPN-ArrayType, PartDetailsType**

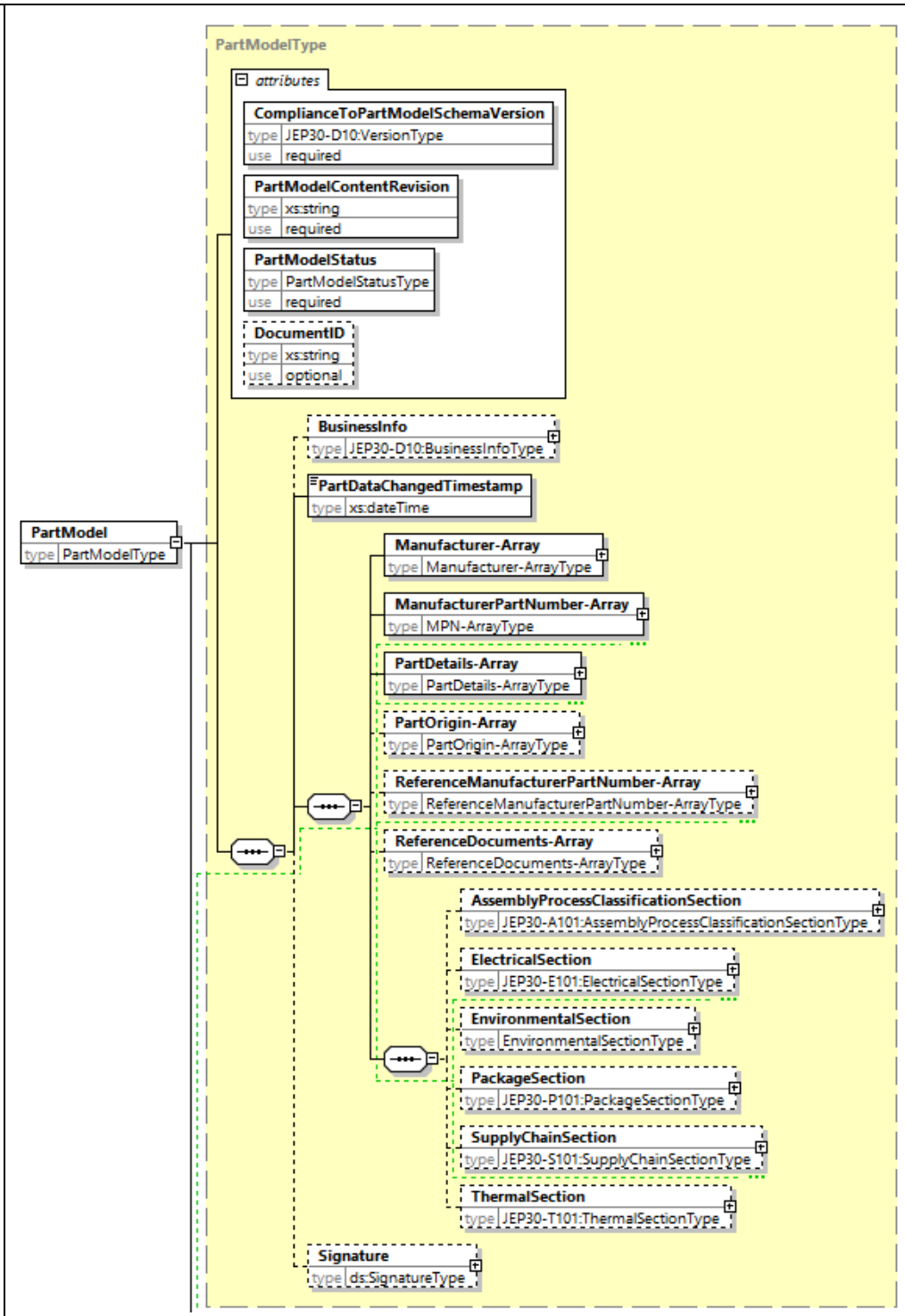
The *Manufacturer/ID* has a constraint *unique* key with the name *ManufacturerKey*. The *ManufacturerPartNumbers/ManufacturerID* has a constraint *keyref* with its own unique name, but it also includes a reference back to the key named *ManufacturerKey*. Both the key and keyref are unique within the entire schema.

4 Part Model Schema Definition

The following section describes the XML Schema structure.

4.1 Part Model

diagram



type **PartModelType.**

4.1 Part Model (cont'd)

The primary purpose of the Part Model Schema is to provide the structure for identifying unique parts (Manufacturer and MPNs), and the structure to include the sub schemas which define the part details. The Part Model is the high-level parent containing:-

1) Compliance to Part Model Schema Version
2) Part Model Content Revision
3) PartModel Status
4) Business Info
5) Part Data Change Timestamp
6) Manufacturer - Array
7) Manufacturer Part Number – Array
8) Part Details - Array
9) Part Origin - Array
10) Reference Manufacturer Part Number – Array
11) Reference Document - Array
12) References to Sub-schemas
    i. Assembly Process Classification
    ii. Electrical
    iii. Environmental
    iv. Package
    v. Supply Chain
    vi. Thermal

This document covers the parental structure of the part model, and its reference to its sub-schemas. The content under any sub-schema is tied to the Manufacturer's name and Manufacturer's part number as seen in the following section. The same concept applies to all the sub-schemas, thus enabling the data to be connected from one sub-schema to another sub-schema, to make a complete set of data for any given part or set of parts.

The ComplianceToPartModelSchemaVersion *indicates the version of the Schema to which the XML file is to be validated against.*

The *PartModelContentRevision* indicates the revision of the data for the Part that is submitted in the XML file. This enables the Component Manufacturer to provide a new XML file for a Part each time they wish to upgrade a new set of data for a part, in any of the child sections.

The *PartModelStatus* is a mandatory attribute that determines the status of the PartModel XML file. It has enumerated values of Pre-Release, Released, Superseded, and Withdrawn.

Both the part model Schema and each of the sub-schemas include a reference to the Part Model Types Library, which enables the xml data types to remain consistent from one sub-schema to another sub-schema.

4.1.1 Schema Release and Versioning

Release and versioning are synchronized between each sub-schema and its part model parent schema. The versioning concept follows the logic outlined below:-

1) The format of the version number is X.Y.Z, where
    i. X increases when there is a change in the released standards document, that requires a change to the XML Schema to support that document, e.g.
        a. Updates to the J-STD-075 or to the J-STD-020, that drive a change in the *"AssemblyProcessClassificationSection"* of the Part Model Schema, or
        b. Updates to the JESD30, that drive a change in the *"PackageSection"* of the Part Model Schema, or
        c. Updates to the JESD77, JESD99, etc. , that drive a change in the *"ElectricalSection"* of the Part Model Schema, or
        d. Updates to the J-STD-046 or to the J-STD-048, that drive a change in the *"SupplyChainSection"* of the Part Model Schema, or
        e. Updates to the JESD51-XX, that drive a change in the *"ThermalSection"* of the Part Model Schema, or
        f. Etc.,
    ii. Y increases when there is a backwards incompatible change, such as
        a. A previous optional element now being made mandatory,
        b. A new mandatory element being added, or
        c. Changing the structure of a branch which breaks the backward incompatibility of the schema.

Backward incompatible changes may require clients to update their systems that use the part model.

    iii. Z increases when there is a backwards compatible change, such as but not limited to:-
        a. A previous mandatory element now being made optional,
        b. A new optional element being added, or
        c. An existing element being made unbounded.

Each time that a sub-schema gets updated, then the part model version also gets updated to release that sub-schema under the umbrella of the Part Model. This is because the Part Model must now reference the new version of sub-schema, since all sub-schemas have their own version number. The parent schema includes them by referring to a precise version, so a version bump in the sub-schema requires a version bump in the parent schema only at the time of release of the Parent.

XML Files that are used for communicating part data to the User, can be validated against any version of the *PartModel* schema to ensure that it is valid according to that Schema. Validation of a specific parts XML against the schema does not ensure correctness of content. It only ensures the structure is correct and ensures some portions of the content use an acceptable value.

4.1.1 Schema Release and Versioning (cont'd)

Table 2 - Schema Versioning

Part Model	Sub-Schemas					
	Assembly Process Classification	Electrical	Package	Supply Chain	Thermal	Part Model Types Library
1-0-0	1-0-0	1-0-0	1-0-0		1-0-0	1-0-0
2-0-0	2-0-0	2-0-0	2-0-0	1-0-0	2-0-0	2-0-0
3-0-0	2-0-1	2-0-1	3-0-0	1-0-1	2-0-1	3-0-0
4-0-0	2-0-1	3-0-0	4-0-0	1-0-1	2-0-1	3-0-0
5-0-0	2-0-2	3-0-1	5-0-0	1-0-2	2-0-2	3-0-1

The purpose of this organization is to enable the Part manufacturer to focus their data transfer to any specific sub-set of data that is required by the consumer of the data, without having to define the entire set of content applicable to the Part. This means that a part manufacturer can create two separate files containing

1. *Manufacturer-Array, ManufacturerPartNumber-Array, PartDetails-Array* details plus the *PackageSection* details,
2. *Manufacturer-Array, ManufacturerPartNumber-Array, PartDetails-Array* details plus the *AssemblyProcessClassificationSection* details.

The consumer of those files can then extract out the data from each of these files and merge them together to form a more complete data set for a specific part. Validation that the two files contain the same *Manufacturer-Array, ManufacturerPartNumber-Array*, can be done by comparing the *Digest Method* and *Digest Value contained within the ManufacturerSignature* and the *ManufacturerPartNumbersIdentitySignature.*

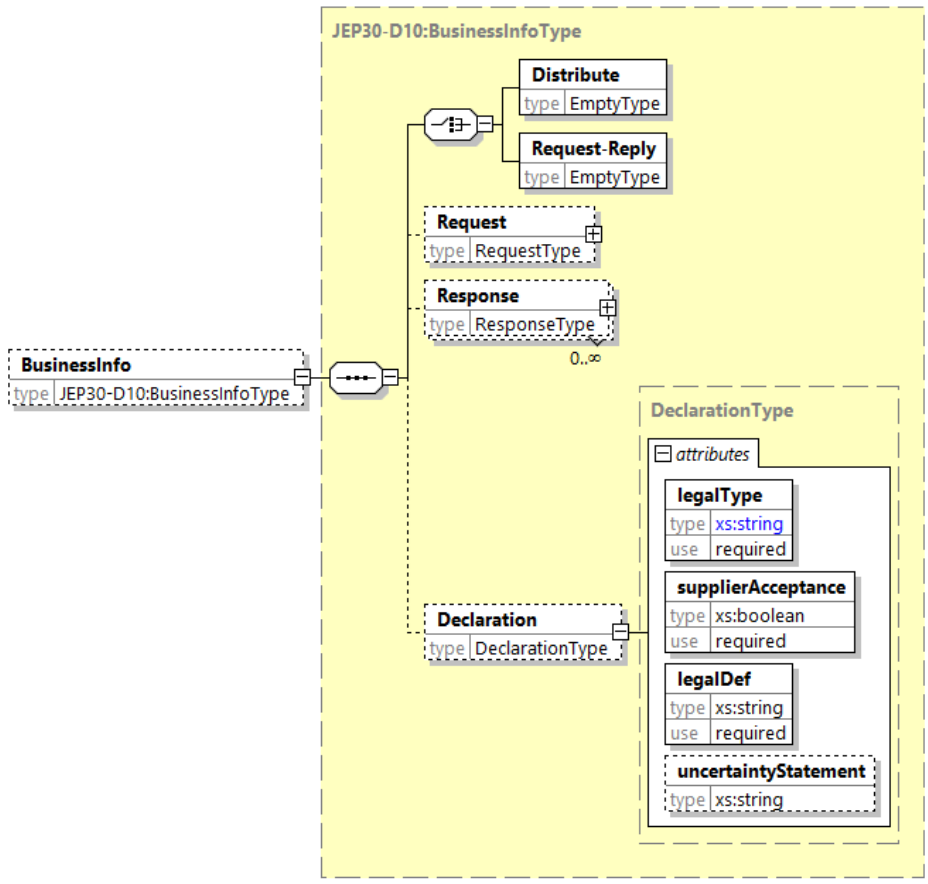
Alternatively, the part manufacturer can decide to include all these branches of data into the one file. The file can support data across all the components of the xml schema, for just one Part, or for multiple Parts.

The concepts of separating out the MPN from the Family details is based on the following concepts

1. The list of MPN data can be very big.
2. There can be many Alternative MPN's that can be inserted into the file under the Array

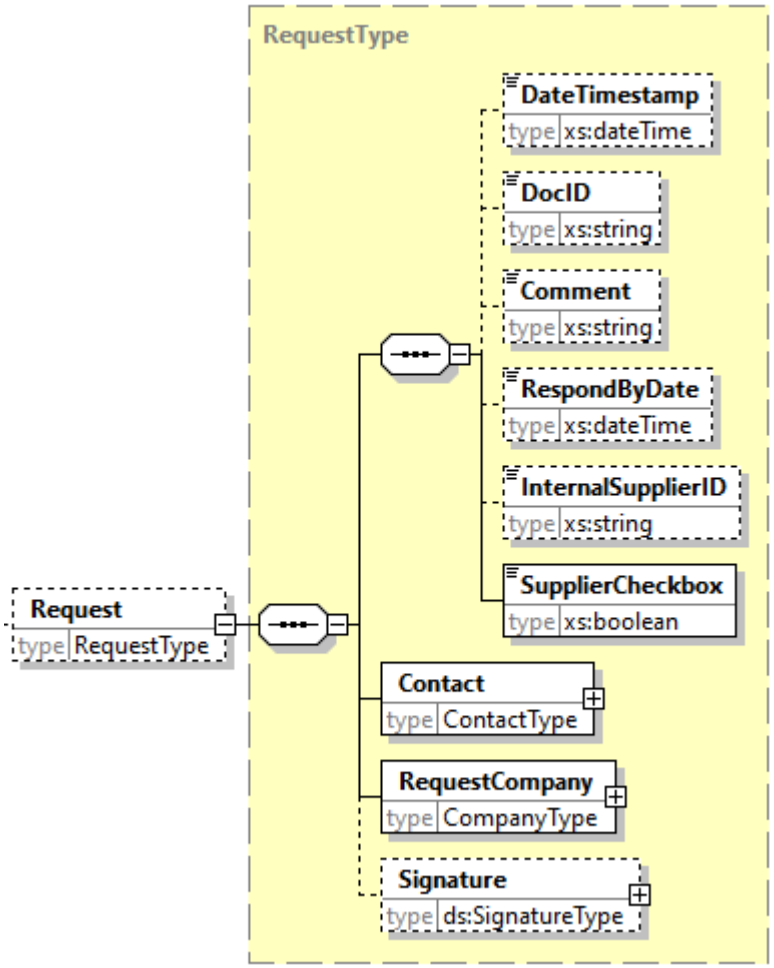
Since many of these MPN's, and Alternative MPN's (each represented by a different string of alphanumeric characters), can map to a single XML component representation, e.g. the Package component of the XML Schema, the XML file can contain just one representation of the Package Data, while having multiple records of the MPN data. This concept eliminates data duplicity, minimizes data file size, and makes the transfer of data highly efficient. The linking of different xml components of data is enabled using ID's.

4.2 Business Info

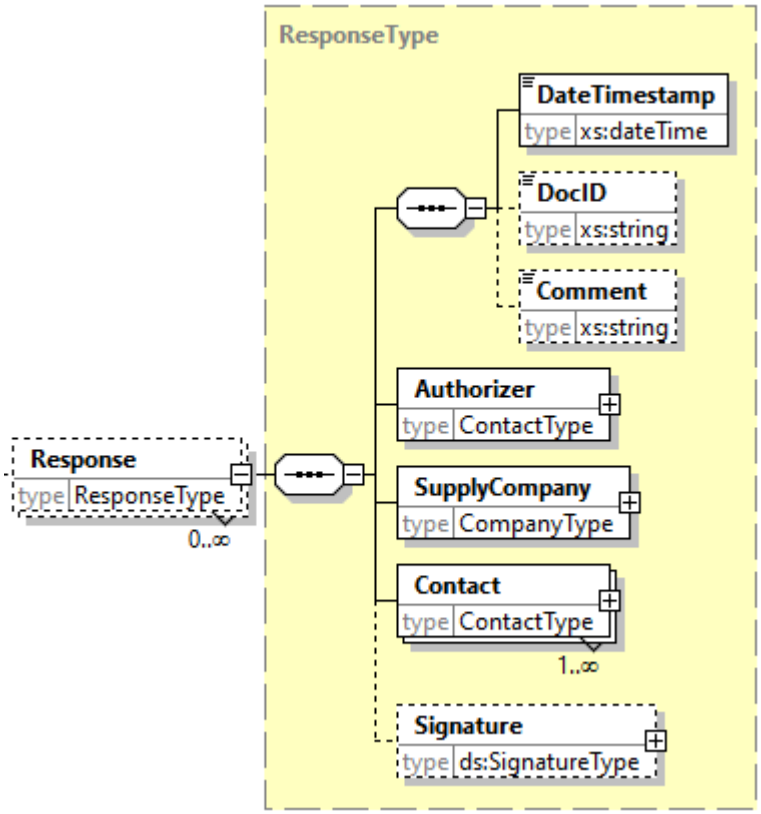
path	**PartModel/BusinessInfo**
diagram	
type	**JEP30-D10:BusinessInfoType, RequestType, ResponseType, DeclarationType.**

The Business Information section contains a structure to capture the details of the person who is requesting the data from the supplier, plus the details of the person and authorizer who is responding to that request.

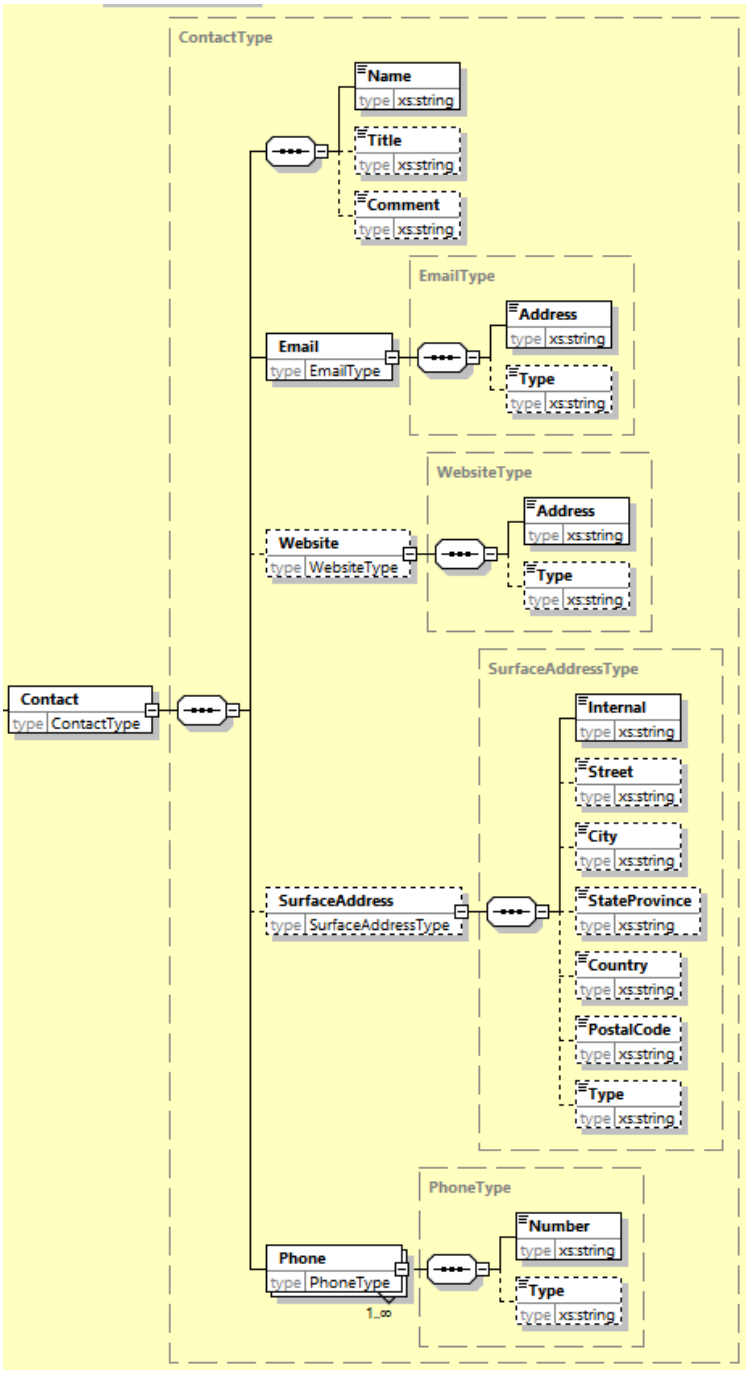
4.2.1 Request

path	**PartModel/BusinessInfo/Request**
diagram	
type	**RequestType, ContactType, CompanyType, ds:SignatureType**

4.2.2 Response

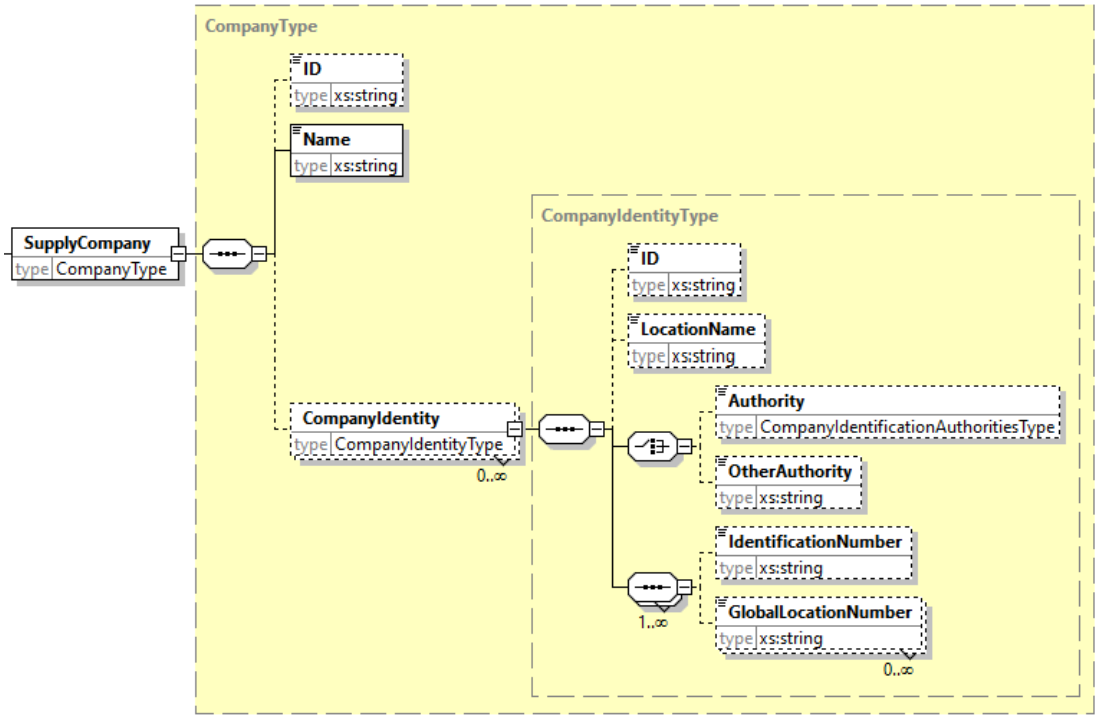
path	**PartModel/BusinessInfo/Response**
diagram	
type	**Response Type, ContactType, CompanyType, ds:SignatureType**

4.2.2.1 Contact Type

paths	**PartModel/BusinessInfo/Request/Contact, PartModel/BusinessInfo/Response/Authorizer, PartModel/BusinessInfo/Response/Contact, PartModel/Manufacturer-Array/Manufacturer/Authorizer, PartModel/Manufacturer-Array/Manufacturer/Contact.,.....**
diagram	
type	**ContactType, EmailType, WebsiteType, SurfaceAddressType, PhoneType**

Contact details should be company specific contact details and should not have any Personal Identifiable information contained within the file.

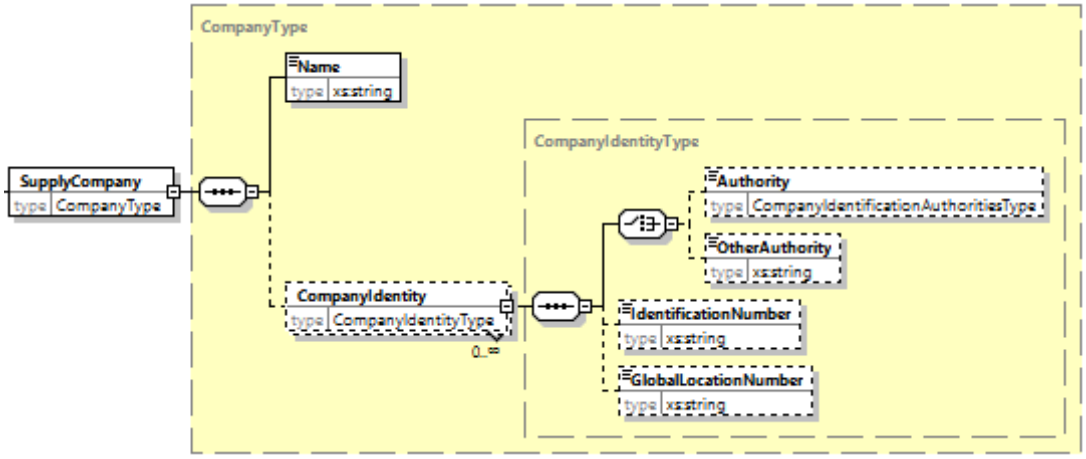
4.2.2.2 Company Type

path	**PartModel/BusinessInfo/Response/RequestCompany**
diagram	
type	**CompanyType, CompanyIdentityType, CompanyIdentificationAuthoritiesType.**

The *RequestCompany* represents the company seeking the information of the part or parts in an xml format that complies with this schema.

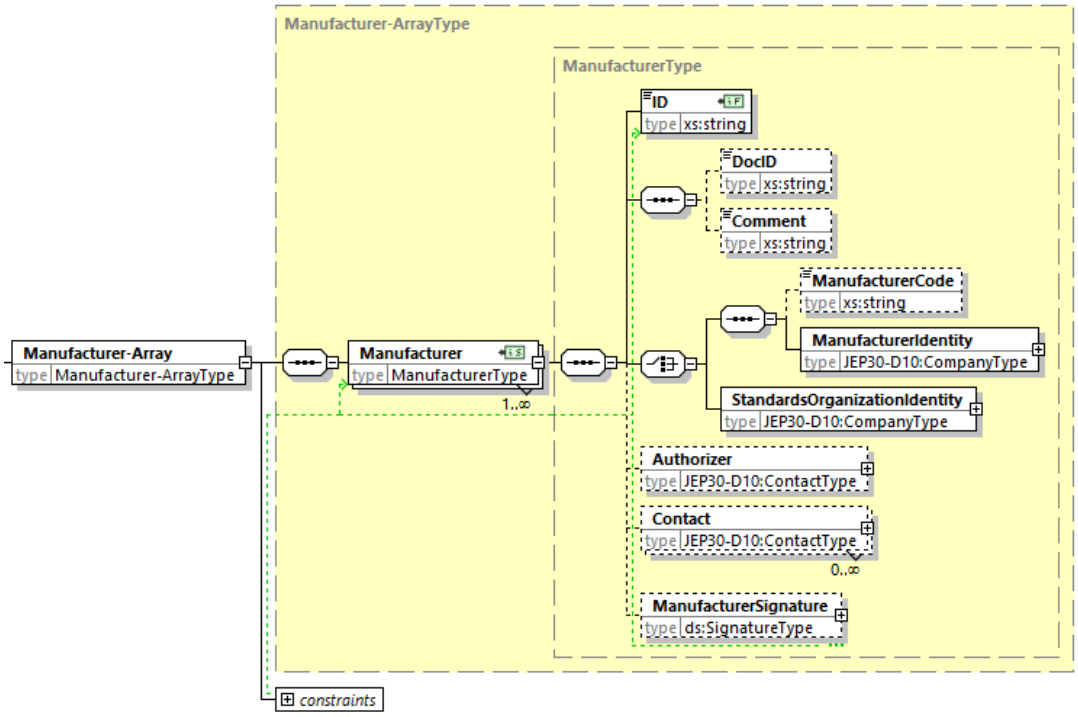
Section 4.4 Company Type below provides details for the *CompanyIdentity*.

4.2.2.3 Supply Company

path	**PartModel//BusinessInfo/Response/SupplyCompany**
diagram	
type	**CompanyType, CompanyIdentityType, CompanyIdentificationAuthoritiesType.**

The *SupplyCompany* represents the company providing the information of the part or parts in an xml format that complies with this schema. The supply company can be a distributor of the parts manufactured by a different company. If the supply company and the manufacturer of the parts represented in the xml file is the same company, then the *BusinessInfo/Response* does not have to be populated, as the manufacturer details will be captured under the *Manufacturer-Array* branch, as outlined in section 4.3 Manufacturer – Array below, while Section 4.3.1 Manufacturer Identity below provides the details for the *CompanyIdentity*.

4.3 Manufacturer – Array

path	**PartModel/Manufacturer-Array**
diagram	
type	**Manufacturer-ArrayType, ManufacturerType, CompanyType, ContactType, ds:SignatureType**

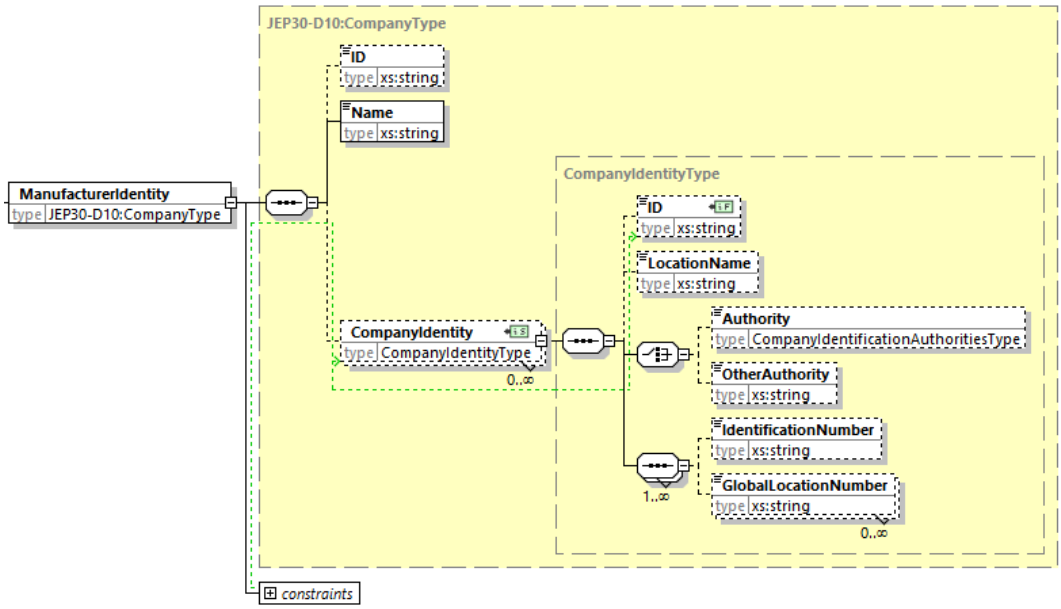
The Manufacturer Array section is a separate section that contains the data for the identity of the Manufacturer that manufacturers the part. Since many parts from the same manufacturer can be contained within the same xml file, the part references the manufacturer via the *Manufacturer/ID*.

The *ManufacturerIdentity* represents the company that manufacturers the part or parts represented in this xml. The manufacturer is usually also the supplier of the parts, but they can also have their parts distributed by other companies. This section under *Manufacturer-Array/Manufacturer* is mandatory to be provided in the XML file.

The addition of the *StandardsOrganizationIdentity* as a choice to the *ManufacturerIdentity*, provide the capability to now enable Standards Organizations to also create PartModel content for their published part related standards. An example of this is the JEDEC Standard packages could now be represented under the *PackageSection*.

If the supply company and the manufacturer of the parts represented in the xml file are different companies, and if this xml file is being provided by the supply company, then the *BusinessInfo/Response* section should also be populated with the supplier details, as outlined 4.2 above.

4.3.1 Manufacturer Identity

path	**PartModel/Manufacturer-Array/Manufacturer/ManufacturerIdentity**
diagram	
type	**CompanyType, CompanyIdentityType, CompanyIdentificationAuthoritiesType.**

4.4 Company Type

The *CompanyIdentity* structure is optional and can follow the standards as outlined in the ISO/IEC 6523 Information technology – Structure for the identification of organizations and organization parts. This is an international standard that defines a structure for uniquely identifying organizations and parts thereof in computer data interchange and specifies the registration procedure to obtain an International Code Designator (ICD) value for an identification scheme. The standard consists of two parts:

1. Part 1: Identification of organization identification schemes defines a structure for the identification of organizations and parts thereof. The components of this structure are the following:
   i. An International Code Designator (ICD) value, which uniquely identifies the authority (*CompanyIdentity/Authority*) which issued the code to the organization.
   ii. An organization identifier (*IdentificationNumber*), up to a maximum of 35 characters.
2. Part 2: Registration of organization identification schemes defines the registration procedure for ICD values. The registration authority for ICD values is Farance Inc. on behalf of the American National Standards Institute (ANSI).

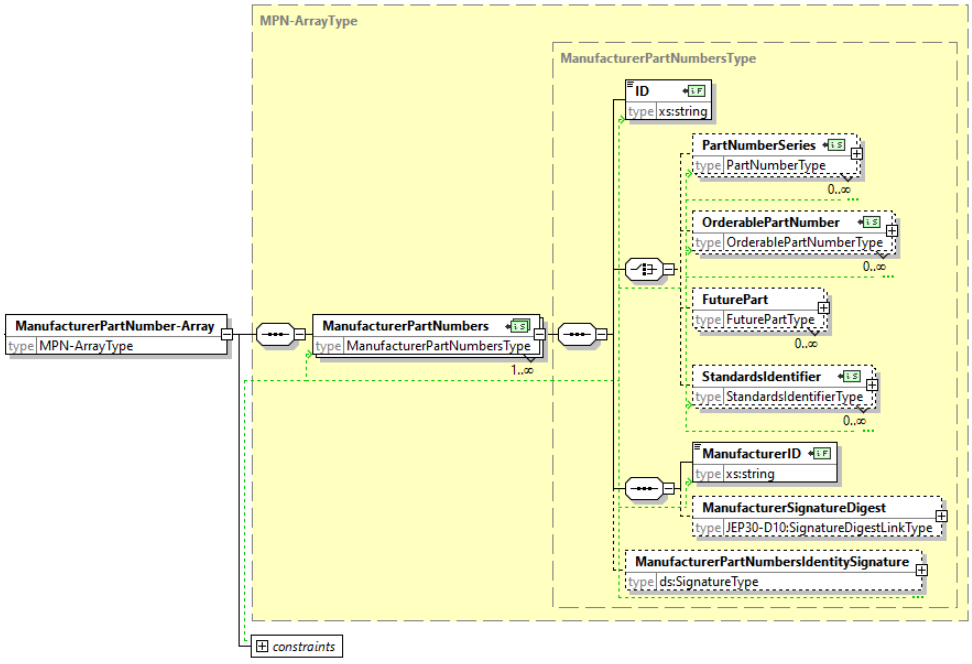
The most widespread standard compliant with ISO 6523 norm is the identifier called "Global Location Number" (GLN), developed by GS1 company members. In B2B exchanges, it is widely used by companies to identify locations or functions within a location (i.e., a factory, accounting department of a company, an administration, a warehouse, a delivery address). It has become a key to exchange business messages (orders, invoices, etc.) using UN/EDIFACT specifications.

4.4 Company Type (cont'd)

The *CompanyIdentificationAuthoritiesType* has the following enumerated values.

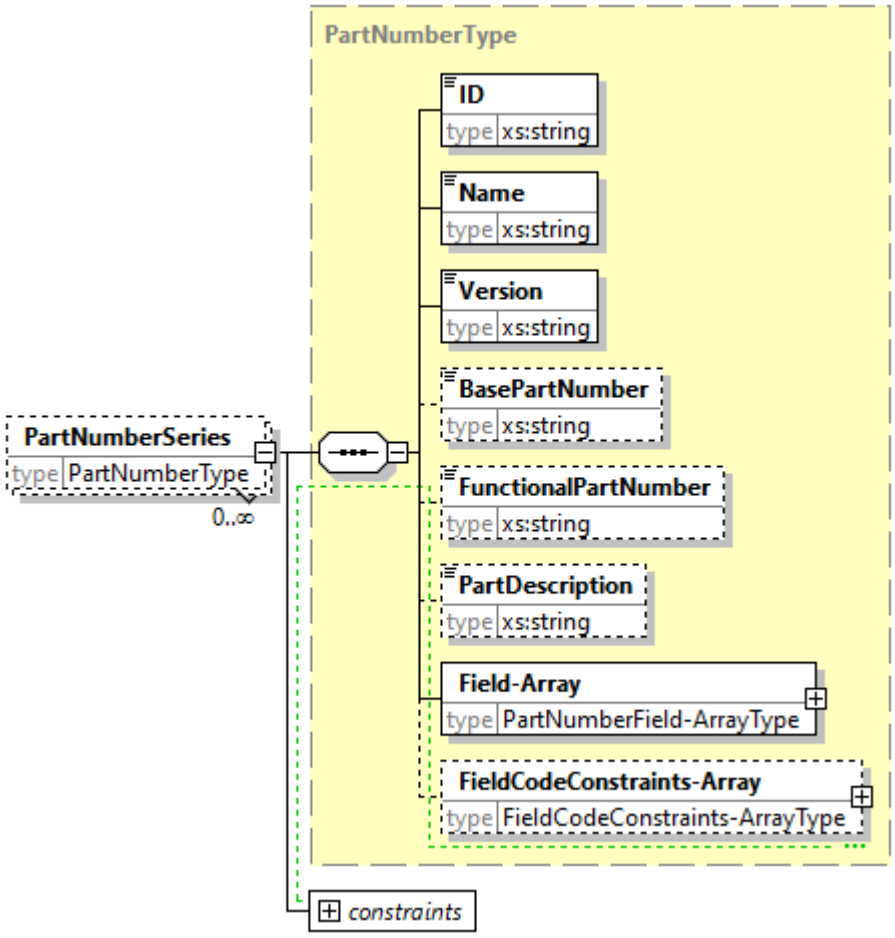
1. GS1,
2. CAGE,
3. GLEIF,
4. SAM UEI,
5. DUNS,
6. Tax Identification Number,
7. Other National Authority Identifiers can also be recorded under *OtherAuthority*.

4.5 Manufacturer Part Number – Array

path	**PartModel/ManufacturerPartNumber-Array**
diagram	
type	**MPN–ArrayType, ManufacturerPartNumbersType, PartNumberType, OrderablePartNumberType, FuturePartType, JEP30-D10:SignatureDigestLinkType, ds:SignatureType.**

The *ManufacturerPartNumber-Array/ManufacturerPartNumber* provides the definition of the part number or a specific Standard, so that it can be connected to the technical specification as shown in section 4.6 Part Details below. The part number definitions are explained in the sub-sections below. All Parts which are intended to be communicated to the consumer of the data should be captured either under the *PartNumberSeries, OrderablePartNumber-Array* or the *FuturePart*. Alternatively, applicable Standards should be captured either under *StandardsIdentifier*. These can be connected to the technical specifications of the Part via the *PartDetails* section. Parts which are referenced but not intended to have their technical details provided in full are captured under the *ReferenceManufacturerPartNumber-Array* as outlined in 4.7 Reference Manufacturer Part Number – Array below.

4.5.1 Part Number Series

path	**PartModel/ManufacturerPartNumber-Array/ManufacturerPartNumbers/PartNumberSeries**
diagram	
type	**PartNumberType, PartNumberField-ArrayType, FieldCodeConstraints-ArrayType.**

The *PartNumberSeries* branch is a structure to enable the capture of a part number series similar to the one shown in Table 3 - Part Number Ordering Information.

The *BasePartnumber* is the common section of the part numbers that are represented by this *PartNumberSeries*.

The *FunctionalPartNumber* is an expanded version of the *BasePartNumber* that represents the functionality of the part. This *FunctionalPartNumber* may omit characters from the full orderable part number, for example, characters that define the physical package of the intended device, the material composition of the device or the packing format in which the device is shipped. However, the *FunctionalPartNumber* must contain the characters to sufficiently define the part from a functionality perspective.

The *PartDescription* is a description that defines the PartNumberSeries. It is not intended to define the description of an individual orderable part number within the *PartNumberSeries.*

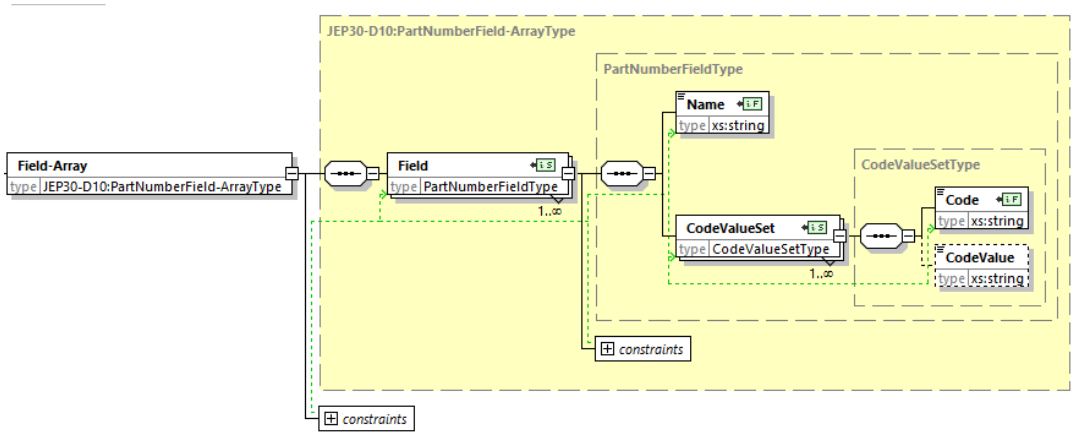
4.5.1 Part Number Series (cont'd)

Table 3 - Part Number Ordering Information

Package Body GC0402	Dielectric	Capacitance Value	Tolerance	Terminal Finish	DC Voltage Rating	Marking	Delivery Mode
0402 0603 0805 1206 1210 1812 NOTE Size ≥ 1206 have min capacitance value of 390 pF	A = COG (NPO) Y = X7R H = X8R	Expressed in picofarads (pF). The first two digits are significant, the third is a multiplier. An "R" indicates a decimal Point. Examples 102 = 1000 pF Ex. 1R0, 6R8, 100, 121, 685, etc..	B = ± 0.10 Pf C = ± 0.25 pF D = ± 0.5 pF F = ± 1 % J = ± 5 % K = ± 10 % NOTE C0G (NP0): B, C, D < 10 Pf; F, J, K, ≥ 10pF; X7R/X8R: J, K	B: SnPb over nickel. barrier N: SnAg over nickel. barrier G: Gold over nickel barrier	A = 16 V B = 25 V C = 50 V D = 100 V	A = unmarked B = marked.	T = 7" reel / plastic tape. O = 7" reel / paper tape. R = 11.25" / 13" reel / plastic tape. I = 11.25" / 13" reel / plastic tape.
6	3	5	6	3	4	2	4

NOTE 1 Delivery Mode - "I" and "O" are restricted for size 0402 / 0603 / 0805.
NOTE 2 Marking is not allowed for Package Body sizes smaller than or equal to 0805.
NOTE 3 Part number GC1206A222J-BCAT is not allowed.
NOTE 4 Exception to the above constraints, part number GC1812A100F-GDAT is allowed via special order.

4.5.1.1 Field – Array

path	**PartModel/ManufacturerPartNumber-Array/ManufacturerPartNumbers/PartNumberSeries /Field-Array**
diagram	
type	**PartNumberField-ArrayType, PartNumberFieldType, CodeValueSetType.**

The *Field-Array* of the *PartNumberSeries* branch captures all the different field data that makes up the combination of the values of each heading in Table 3 above. The XML files shows a sample of this content below.

Each *Field/Name* should be labeled as the same name as the heading in the table. Each *Field/Code* should contain one of the values of the enumerated list of values under that specific column name in the table. The *CodeValue* is an optional element added to assist the user when defining the Part Number Series, since it provides an explanation of the code or a numerical value representation of the code.

Any part number can be constructed by concatenating one of the Field/Code for each Field/Name in sequence, to give for example GA0402Y8R2C-BAAT.

Assuming that there were no constraints on this construction, then the total number of concatenation variations would be calculated as the multiplication of:

1. *BeginTag* = 1
2. *PackageBody* = 6
3. *Dielectric* = 3
4. *CapacitanceValue* = 5 (This number represents the number of values in the series)
5. *Tolerance* = 6
6. *TerminalFinish* = 3
7. *DC-VoltageRating* = 4
8. *Marking* = 2
9. *DeliveryMode* = 4

Total combinations = 51,480

4.5.1.1 Field - Array (cont'd)

```
<PartNumberSeries >
        <ID>Part Construct 1</ID>
        <Name>SMT Ceramic Chip Capacitor</Name>
        <Version>A</Version>
        <Field-Array>
                <Field>
                        <Name>BeginTag</Name>
                        <CodeValueSet>
                                <Code>GC</Code>
                        </CodeValueSet>
                </Field>
                <Field>
                        <Name>PackageBody</Name>
                        <CodeValueSet>
                                <Code>0402</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>0603</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>0805</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>1206</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>1210</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>1812</Code>
                        </CodeValueSet>
                </Field>
                <Field>
                        <Name>Dielectric</Name>
                        <CodeValueSet>
                                <Code>A</Code>
                                <CodeValue>C0G (NPO)</CodeValue>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>Y</Code>
                                <CodeValue>X7R</CodeValue>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>H</Code>
                                <CodeValue>X8R</CodeValue>
                        </CodeValueSet>
                </Field>
                <Field>
                        <Name>CapacitanceValue</Name>
                        <CodeValueSet>
                                <Code>1R0</Code>
                                <CodeValue>1.0 pF</CodeValue>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>6R8</Code>
```

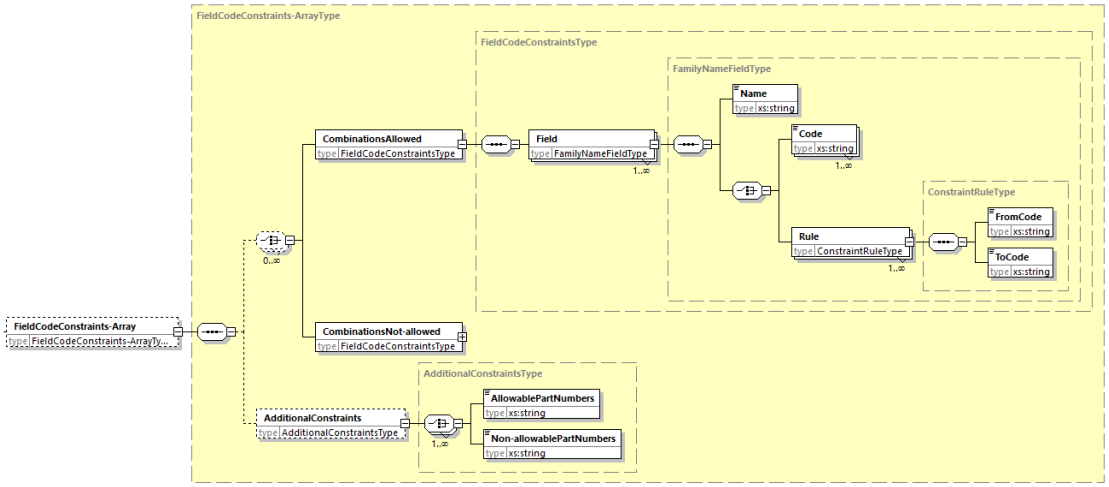
4.5.1.1 Field - Array (cont'd)

```
                <CodeValue>8.2 pF</CodeValue>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>100</Code>
                        <CodeValue>10 pF</CodeValue>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>121</Code>
                        <CodeValue>120 pF</CodeValue>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>685</Code>
                        <CodeValue>6.8 µF</CodeValue>
                </CodeValueSet>
        </Field>
        <Field>
                <Name>Tolerance</Name>
                <CodeValueSet>
                        <Code>B</Code>
                        <CodeValue>± 0.10 pF</CodeValue>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>C</Code>
                        <CodeValue>± 0.25 pF</CodeValue>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>D</Code>
                        <CodeValue>± 0.5 pF</CodeValue>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>F</Code>
                        <CodeValue>± 1 %</CodeValue>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>J</Code>
                        <CodeValue>± 5 %</CodeValue>
                </CodeValueSet>
        </Field>
        <Field>
                <Name>Separator</Name>
                <CodeValueSet>
                        <Code>-</Code>
                </CodeValueSet>
        </Field>
        <Field>
                <Name>TerminalFinish</Name>
                <CodeValueSet>
                        <Code>B</Code>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>N</Code>
                </CodeValueSet>
                <CodeValueSet>
                        <Code>G</Code>
                </CodeValueSet>
```

4.5.1.1 Field – Array (cont'd)

```
                </Field>
                <Field>
                        <Name>DC-VoltageRating</Name>
                        <CodeValueSet>
                                <Code>A</Code>
                                <CodeValue>50 V</CodeValue>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>B</Code>
                                <CodeValue>100 V</CodeValue>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>C</Code>
                                <CodeValue>200 V</CodeValue>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>D</Code>
                                <CodeValue>250 V</CodeValue>
                        </CodeValueSet>
                </Field>
                <Field>
                        <Name>Marking</Name>
                        <CodeValueSet>
                                <Code>A</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>B</Code>
                        </CodeValueSet>
                </Field>
                <Field>
                        <Name>Packaging</Name>
                        <CodeValueSet>
                                <Code>T</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>O</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>R</Code>
                        </CodeValueSet>
                        <CodeValueSet>
                                <Code>I</Code>
                        </CodeValueSet>
                </Field>
        </Field-Array>
</PartNumberSeries >
```

4.5.1.2 Field Code Constraints – Array

path	**PartModel/ManufacturerPartNumber-Array/ManufacturerPartNumbers/PartNumberSeries/FieldCodeConstraints-Array**
diagram	
type	**FieldCodeConstraints-ArrayType, FieldCodeConstraintsType, FamilyNameFieldType, ConstraintRuleType, AdditionalConstraintsType.**

Constraints are added to the *PartNumberSeries* branch via the *FieldCodeConstraints-Array* section. A constraint can be any of several types

1. Combinations allowed
2. Combinations that are not allowed
3. Additional Constraints

Combinational constraints that are allowed (*FieldCodeConstraints-Array/CombinationsAllowed*) or not allowed (*FieldCodeConstraints-Array/CombinationsNot-allowed*) can be defined through a series of field (*Field/Name*) combinations, each with a sub-set of values (*Field/Code*). The list of values under

PartNumberSeries /FieldCodeConstraints-Array/CombinationsAllowed/Field/Code

or

PartNumberSeries /FieldCodeConstraints-Array/CombinationsNot-allowed/Field/Code

must be a sub-set of the list of values under *PartNumberSeries /Field-Array/Field/Code*.

Constraints can also be rule based and can be applied to both the *CombinationsAllowed* and the *CombinationsNot-allowed* via the *Field/Rule* branch. Provided that the codes are properly sequentially ordered in the *PartNumberSeries/Field-Array/Field/Code* element in the XML file, then a subset of these codes can be identified using the *Rule/FromCode* and the *Rule/ToCode* elements.

4.5.1.2 Field Code Constraints – Array (cont'd)

Since multiple instances of *CombinationsAllowed* and *CombinationsNot-allowed* structures are allowed within the xml file, some of which may overlap with others, then the order of insertion into the xml file is important, whereby each subsequent entry overrides the previous entry for the portion that is overlapped. The following shows some examples of how this is used.

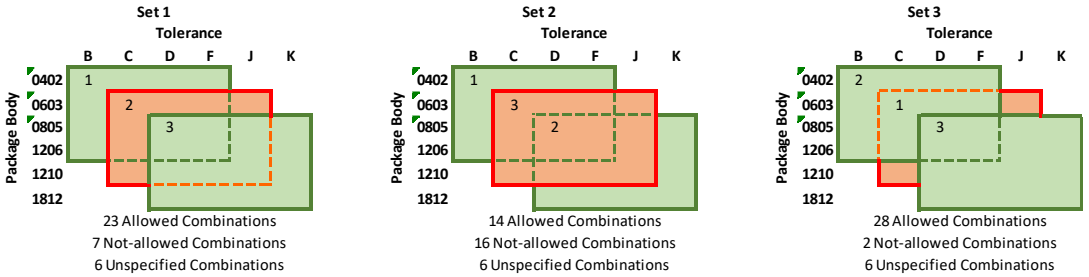


Figure 8 – Examples of Combinations Allowed and Not-allowed

Each set has the same number of parts within each constraint (i.e., 4 criteria for Field 1 and 4 criteria for Field 2). The order of placement within the xml file is denoted by the number in the top left box namely 1, 2 and 3. The green boxes represent the combinations allowed while the red box represents the combinations not allowed. Un-specified combinations specifically for fields listed in that specific constraint can be treated as not allowed. On the other hand, fields that are not specified in a constraint means that all combinations for that field are allowed.

This means that Dielectric, Capacitance Value, Terminal Finish, DC Voltage Rating, Marking and Delivery mode do not impact the constraint and therefore all their values are allowed.

Set 1 first entry (1) represents combinations B(0402) to F(1206) from the first entry in the constraints allowed sequence, followed by entry (2) which represents combinations C(0603) to J(1210). Notice that parts C(0603) to F(1206) overlap with entry (1), but the 2nd entry takes precedence, therefore these combinations are not allowed. The 3rd entry represents combinations D(0805) to K(1812). Notice that D(0805) to J(1210) overlaps that of the previous entry (2), and therefore this combination becomes allowed again. The result therefore are combinations that are allowed are B(0402) to B(1206), C(0402) to F(0402) and D(0805) to K(1812), giving a total of 23 allowed combinations and 7 not-allowed combinations of C(0603) to C(1210), and D(0603) to J(0603).

This reduces the total combinations from 51,840 to 33,120 in this example.

1. *BeginTag* = 1
2. *PackageBody* x *Tolerance* less the constraints = 23
3. *Dielectric* = 3
4. *CapacitanceValue* = 5
5. *TerminalFinish* = 3
6. *DC-VoltageRating* = 4
7. *Marking* = 2
8. *DeliveryMode* = 4

Total combinations = 33,120

4.5.1.2 Field Code Constraints – Array (cont'd)

Set 2 has the order of combinations different whereby entry (1) represents combinations B(0402) to F(1206), followed by entry (2) which represents combinations D(0805) to K(1812), followed by entry (3) which represents combinations C(0603) to J(1210). Since the 3rd entry is the last entry and represents the combinations not allowed, then the combinations allowed are B(0402) to B(1206), C(0402) to F(0402), D(1812) to K(1812), and K(0805) to K(1210), giving a total of 14 allowed combinations and 16 not-allowed combinations of C(0603) to J(1210).

This reduces the total combinations from 51,840 to 20,160 in this example.

1. *BeginTag* = 1
2. *PackageBody* x *Tolerance* less the constraints = 14
3. *Dielectric* = 3
4. *CapacitanceValue* = 5
5. *TerminalFinish* = 3
6. *DC-VoltageRating* = 4
7. *Marking* = 2
8. *DeliveryMode* = 4

Total combinations = 20,160

Set 3 provides a third example of an alternative order of combinations whereby entry (1) represents combinations B(0402) to F(1206), followed by entry (2) which represents combinations C(0603) to J(1210), followed by entry (3) which represents combinations D(0805) to K(1812). In this example, the resultant combinations allowed are B(0402) to C(1206), D(0402) to F(0603) and D(0805) to K(1812) giving a total of 28 allowed combinations of C(1210) and J(0603).

This reduces the total combinations from 51,840 to 40,320 in this example.

1. *BeginTag* = 1
2. *PackageBody* x *Tolerance* less the constraints = 28
3. *Dielectric* = 3
4. *CapacitanceValue* = 5
5. *TerminalFinish* = 3
6. *DC-VoltageRating* = 4
7. *Marking* = 2
8. *DeliveryMode* = 4

Total combinations = 40,320

These examples only show the interaction of two fields but, there is typically more than two interacting fields that define the constraining list of part numbers, and never more than the number of fields that makes up the full part number string (i.e., 8 fields as defined in Table 3 above). When a field is not entered into a constraint, then it is assumed that all the enumerated choices of that field are allowed.

4.5.1.2 Field Code Constraints – Array (cont'd)

Care should be taken to look at the combinations allowed and not allowed before constructing the constraints. Assume that the constraints only had the interaction of two fields as in figure 8 above, then Set 1 should have been constructed using allowed constraints B(0402) to F(0402), B(0603) to B(1206) and D(0805) to K(1812), since it is simpler to understand than the original construct of allowed and non-allowed combinations. Similarly, in Set 3, assuming that these were the only two fields involved, then the first non-allowed constraint is redundant.

Additional constraints via the *AllowablePartNumbers* and the *Non-allowablePartNumbers* can be used to override the combinational constraints.

1. Note 3 in Table 3 indicates a Part that is not allowed, possibly because the market pricing on this is higher due to demand than on the higher voltage rating GC1206A222J-BDAT which is in much higher demand.
2. Note 4 in Table 3 indicates a Part GC1812A100F-GDAT that is allowed, possibly because of a specific customer demand for that unique combination.

The constraints listed in Table 3 — Part Number Ordering Information is defined in the following xml sample:

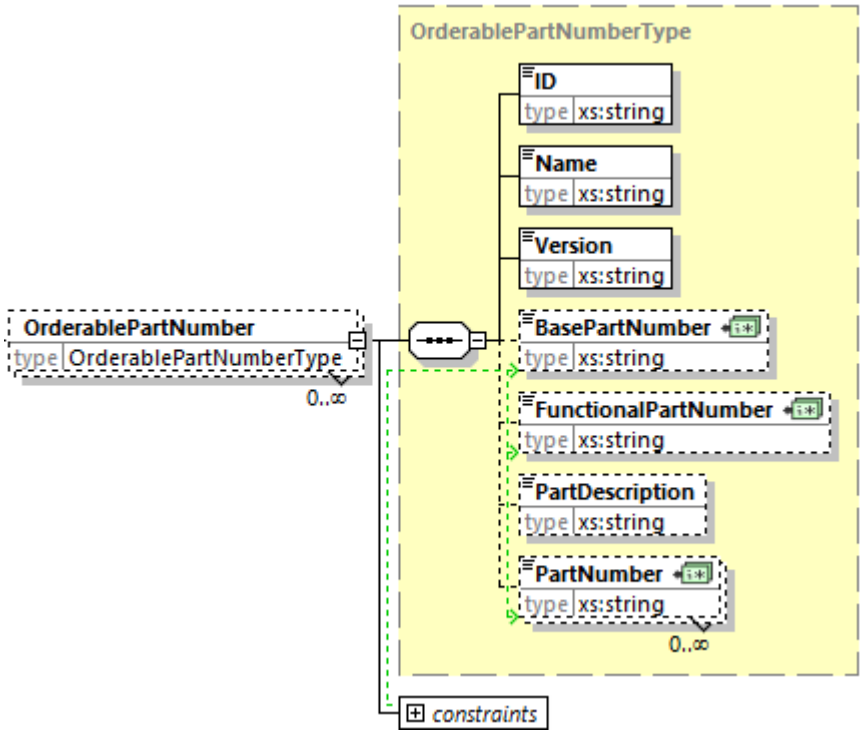
4.5.1.2 Field Code Constraints – Array (cont'd)

```
<FieldCodeConstraints-Array>
 <CombinationsAllowed>
    <Field>
        <Name>Package Body</Name>
      <Rule>
        <FromCode>1206</FromCode>
        <ToCode>1812</ToCode >
        </Rule>
    </Field>
    <Field>
      <Name>Capacitance Value</Name>
      <Rule>
        <FromCode>391</FromCode>
        <ToCode >685</ToCode >
        </Rule>
    </Field>
 </CombinationsAllowed>
 <CombinationsAllowed>
    <Field>
        <Name>Tolerance</Name>
        <Code>B</Code>
        <Code>C</Code>
        <Code>D</Code>
    </Field>
    <Field>
      <Name>Capacitance Value</Name>
      <Rule>
        <FromCode>1R0</FromCode>
        <ToCode >8R2</ToCode >
        </Rule>
    </Field>
 </CombinationsAllowed>
 <CombinationsAllowed>
    <Field>
        <Name>Tolerance</Name>
        <Code>F</Code>
        <Code>J</Code>
        <Code>K</Code>
    </Field>
    <Field>
      <Name>Capacitance Value</Name>
      <Rule>
        <FromCode>100</FromCode>
        <ToCode >685</ToCode >
        </Rule>
    </Field>
 </CombinationsAllowed>
 <CombinationsAllowed>
    <Field>
        <Name>Dielectric</Name>
        <Code>Y</Code>
        <Code>H</Code>
    </Field>
    <Field>
        <Name>Tolerance</Name>
        <Code>J</Code>
        <Code>K</Code>
    </Field>
 </CombinationsAllowed>
```

4.5.1.2 Field Code Constraints – Array (cont'd)

```
<CombinationsNot-allowed>
    <Field>
        <Name>Package Body</Name>
        <Code>0402</Code>
        <Code>0603</Code>
        <Code>0805</Code>
    </Field>
    <Field>
        <Name>Marking</Name>
        <Code>B</Code>
    </Field>
 </CombinationsNot-allowed>
 <AdditionalConstraints>
    <AllowablePartNumbers>GC1812A100F-GDAT</AllowablePartNumbers>
    <Non-allowablePartNumbers>GC1206A222J-BCAT</Non-allowablePartNumbers>
 </AdditionalConstraints>
</FieldCodeConstraints-Array>
```

4.5.2 Orderable Part Number

path	**PartModel/ManufacturerPartNumber-Array/ManufacturerPartNumber/OrderablePartNumber**
diagram	
type	**OrderablePartNumberType.**

This section simply lists out the orderable part numbers for a series. If the *PartNumberSeries* contains enough information to fully define the list of all possible orderable part numbers, then this section *OrderablePartNumber* does not have to be populated. This section is therefore more typically suited to a series in which it:

4.5.2 Orderable Part Number – Array (cont'd)

1. Only has a small number of orderable part numbers, or
2. The part number is not definable in a structured way like that described under *PartNumberSeries*.

The *BasePartnumber* is the common section of the part numbers that are represented by this group of *OrderablePartNumbers/PartNumbers*.

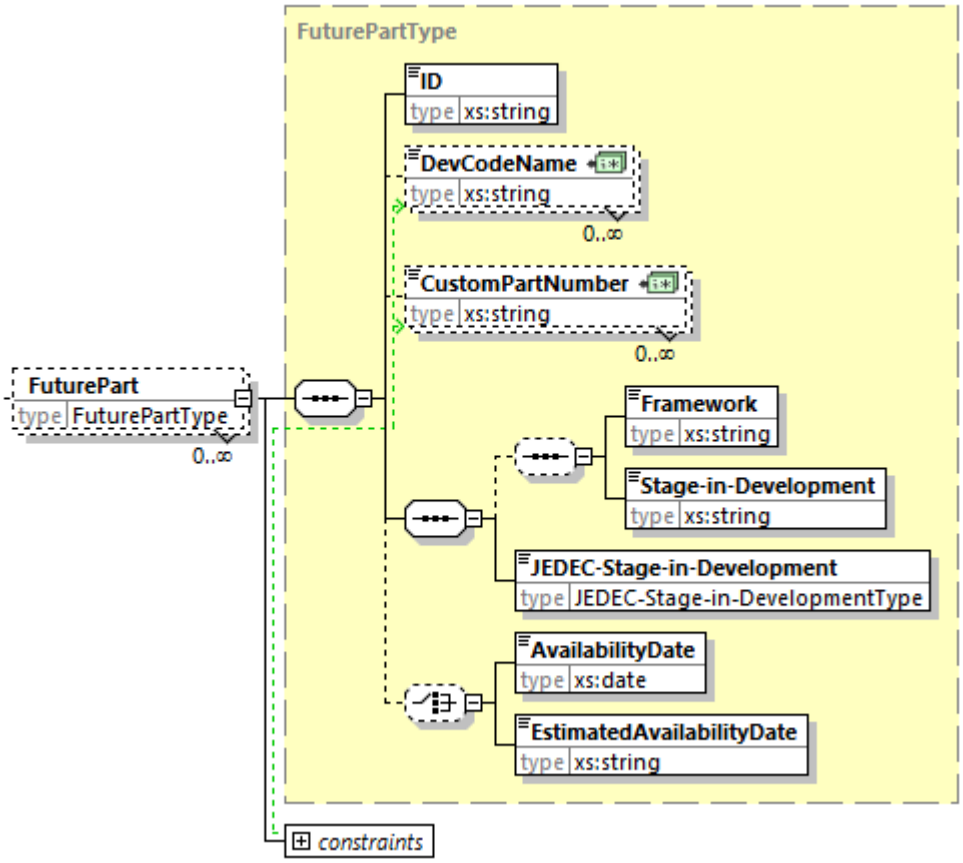
The *FunctionalPartNumber* is an expanded version of the *BasePartNumber* that represents the functionality of the part. This *FunctionalPartNumber* may omit characters from the full orderable part number, for example, characters that define the physical package of the intended device, the material composition of the device or the packing format in which the device is shipped. However, the *FunctionalPartNumber* must contain the characters to sufficiently define the part from a functionality perspective. The example here shows the characters "…10" which denotes the output nominal voltage of the device, whereas the characters "…DVBR" are omitted because they represent the package designator and the packing quantity in tape or reel.

The *PartDescription* is a description that defines the PartNumberSeries. It is not intended to define the description of an individual orderable part number within this group of *OrderablePartNumbers/PartNumbers*.

An example of the xml is:

```
<OrderablePartNumber>
	<ID>OrderablePart-ID1</ID>
	<Name>TLV740P 300-mA, Low-Dropout Regulator With Foldback Current Limit</Name>
	<PartNumber>TLV74010PDBVR</PartNumber>
	<Version>Initial release</ Version >
	<BasePartNumber>TLV740P</BasePartNumber>
	<FunctionalPartNumber>TLV74010P</FunctionalPartNumber>
	<PartDescription>300-mA, Low-Dropout Regulator With Foldback Current Limit
</PartDescription>
	<PartNumber>TLV74010PDBVR</PartNumber>
	<PartNumber>TLV74018PDBVR</PartNumber>
	<PartNumber>TLV74033PDBVR</PartNumber>
	<ManufacturerID>Texas Instruments</ManufacturerID>
</OrderablePartNumber>
```

4.5.3 Future Part

path	**PartModel/ManufacturerPartNumber-Array/ManufacturerPartNumber/FuturePart**
diagram	
type	**FuturePartType, JEDEC-Stage-in-DevelopmentType.**

Component Manufacturers will sometimes interchange data with customer/OEMs and want to leverage the PartModel structure during the Part development. Since these parts are not yet released to production, a *DevCodeName* and/or *CustomPartNumber* may be used to identify the Part, along with a status in development.

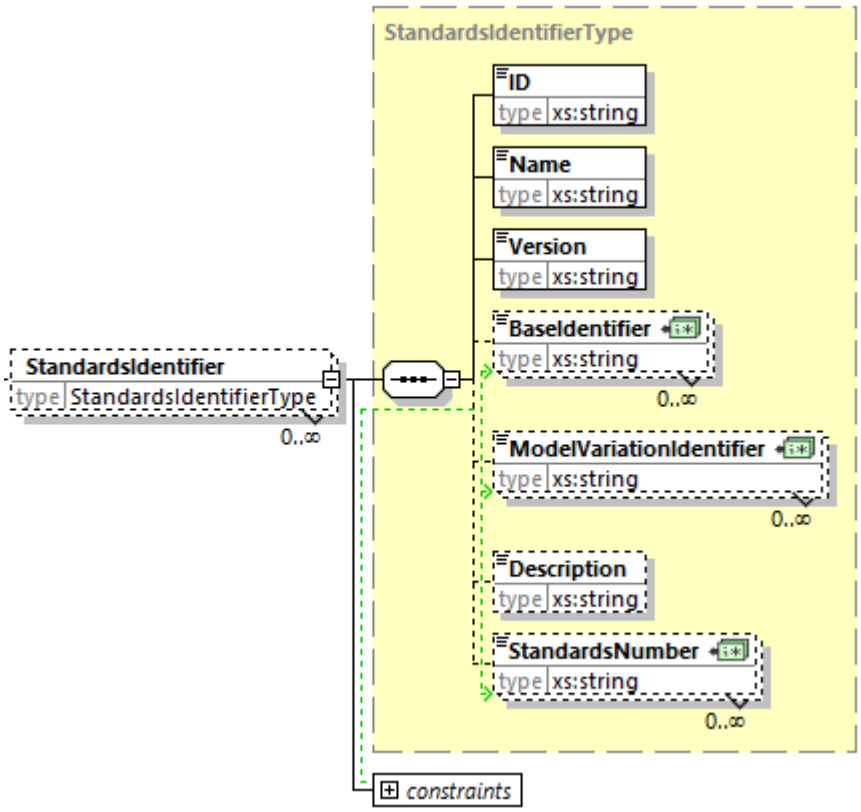
Component Manufacturers and their customers may decide to use their own framework of a life cycle status such as APQC, Agile or other PLM life cycle status process or even their own internal life cycle status program. In this case, the *Framework* identifies the life cycle that is being used and may contain a link to a website that explains these stages. This is accompanied by the *Stage-in-Development* according to the *Framework* criteria.

However, since the PartModel is intended to carry information about a Part that will be consumed by various software tools, it is necessary to cross reference a *Stage-in-Development* according to the *Framework* criteria over to a standardized *JEDEC-Stage-in-Development*. The following are seven enumerated values with an explanation for the *JEDEC-Stage-in-Development*.

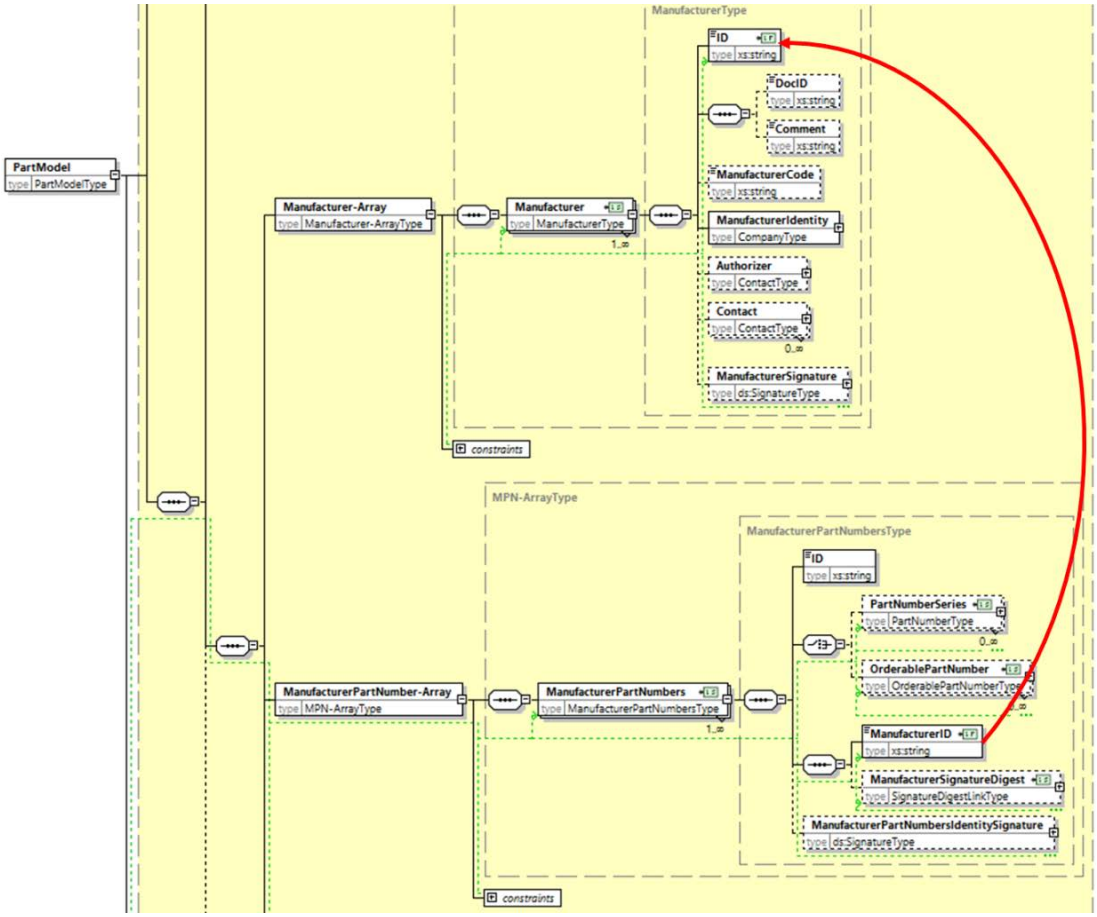
4.5.3 Future Part (cont'd)

1. The Concept and Feasibility Stage:
    i. This initial stage involves developing the concept for a new semiconductor product or technology. Engineers assess the feasibility of the idea based on technical requirements, market needs, and potential benefits.
2. Design Stage:
    i. During this phase, detailed design work takes place. This includes circuit design, layout design, and simulation to ensure that the semiconductor device meets performance specifications and can be manufactured reliably.
3. Prototyping and Testing Stage:
    i. Prototyping involves fabricating a small number of test devices based on the design. These prototypes are used for initial testing and validation of the design concept. Testing includes functional testing, performance testing, and reliability testing.
4. Process Development Stage:
    i. In parallel with design and prototyping, process development occurs. This stage focuses on optimizing the manufacturing process required to produce the semiconductor device. Various process parameters are refined to ensure consistent device performance and yield.
5. Qualification Stage:
    i. Once the design and manufacturing processes are finalized, the semiconductor device undergoes qualification testing. This involves subjecting the devices to a series of tests under different environmental conditions (e.g., temperature, humidity, electrical stress) to ensure reliability and durability.
6. Pilot Production Stage:
    i. Pilot production involves manufacturing a small quantity of devices using the finalized manufacturing process. The goal is to identify any issues that may arise during mass production and to optimize the production flow.
7. Design for Manufacturability (DFM) and Design for Test (DFT) Optimization:
    i. Throughout these stages, engineers focus on optimizing the design for manufacturability (DFM) and testability (DFT). This includes implementing features in the design that facilitate efficient manufacturing and testing processes.

4.5.4 Standards Identifier

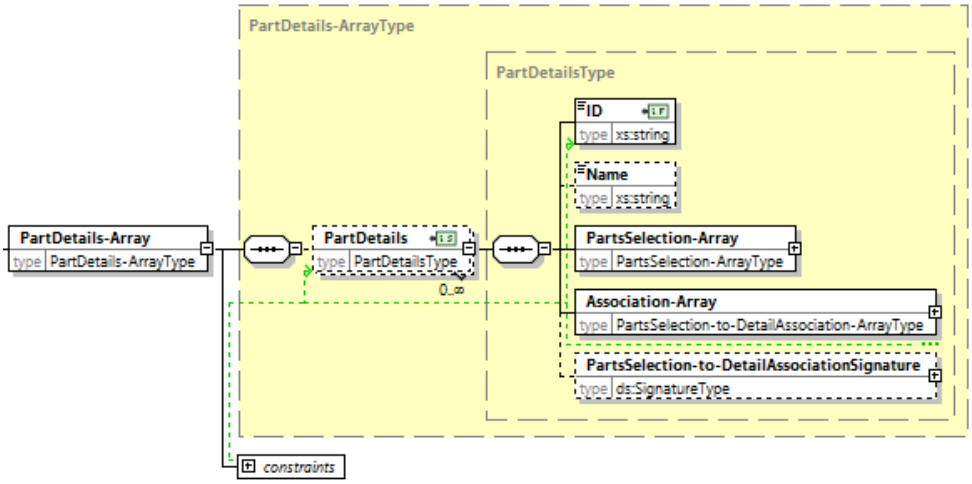
path	**PartModel/ManufacturerPartNumber-Array/ManufacturerPartNumber/StandardsIdentifier**
diagram	
type	**StandardsIdentifierType.**

4.5.5 Linking the Manufacturer to the Manufacturer Part Number

path	**PartModel/Manufacturer-Array, PartModel/ManufacturerPartNumber-Array**
diagram	
type	**Manufacturer-ArrayType->ManufacturerType, MPN-ArrayType->ManufacturerPartNumbersType.**

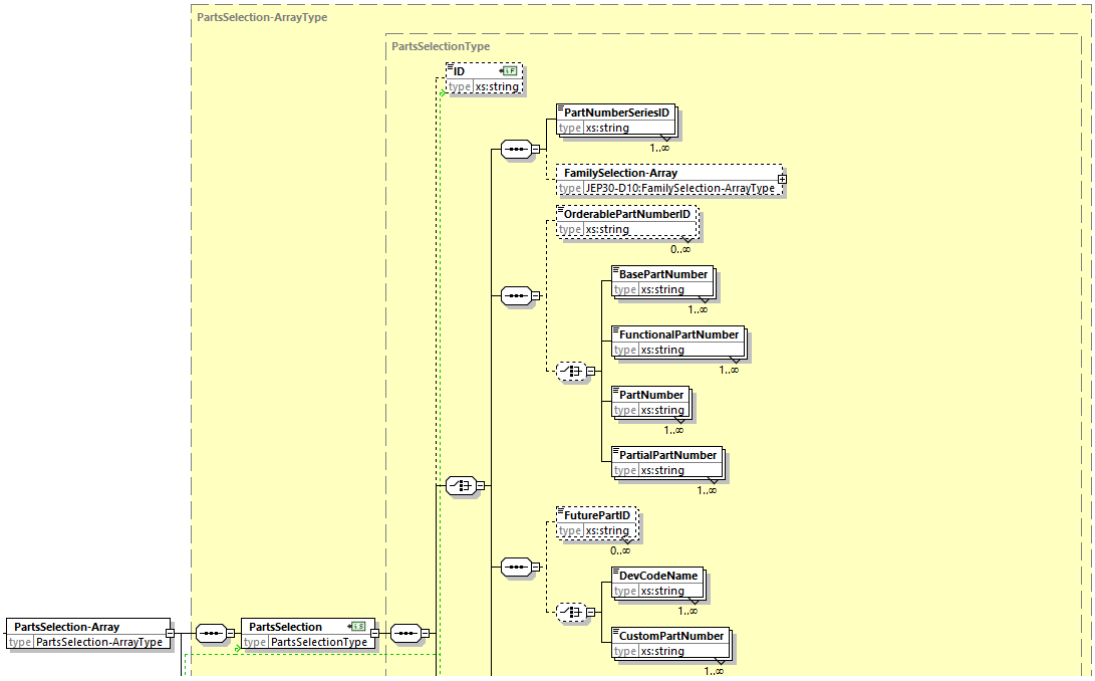
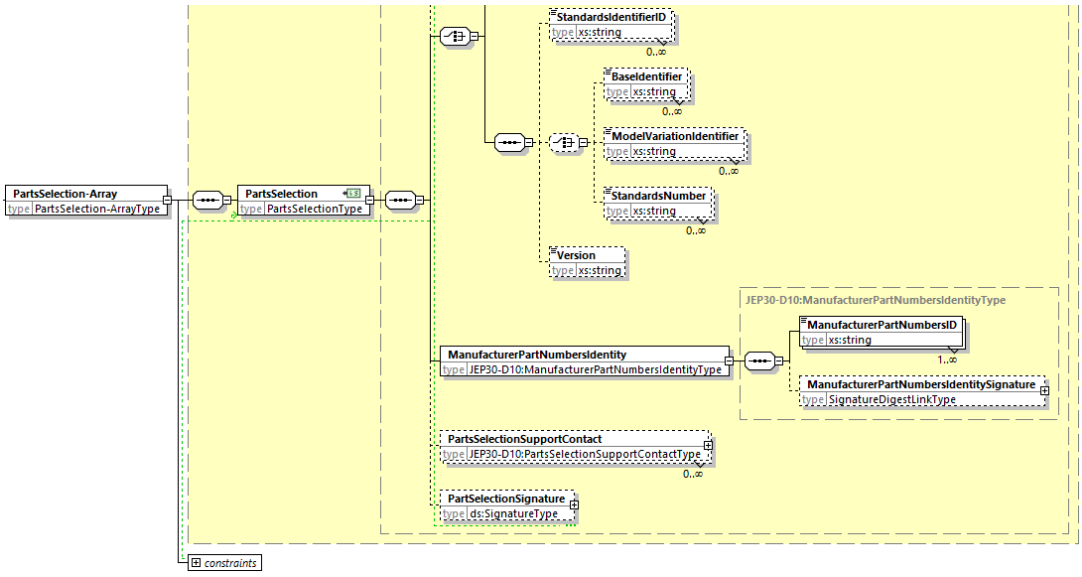
The *ManufacturerPartNumber-Array* can support multiple parts in a single Part Model file. Each of these parts must be referenced to the manufacturer of the part. The method used to connect the Part to the Manufacturer is to link the "*ManufacturerID*" under the *ManufacturerPartNumber-Array/ManufacturerPartNumber* as shown above.

4.6 Part Details - Array

path	**PartModel/PartDetails-Array.**
diagram	
type	**PartDetails-ArrayType, PartDetailsType, PartsSelection-ArrayType**

The linking of the Parts to its technical data is done via the *PartDetails-Array* section.

4.6.1 Parts Selection - Array

path	**PartModel/PartDetails-Array/PartDetails/PartsSelection-Array.**
diagram part 1 of 2	
diagram part 2 of 2	
type	**PartsSelection-ArrayType, PartsSelectionType, JEP30-D10:FamilySelection-ArrayType, JEP30-D10:ManufacturerPartNumbersIdentityType, JEP30-D10:PartsSelectionSupportContactType, SignatureDigestLinkType, ds:SignatureType**

4.6.1 Part Selection - Array (cont'd)

The above diagram shows that by referencing the *PartNumberSeries/ID* as defined in section 4.5.1 Part Number Series above via the keyref element *PartNumberSeriesID*, or referencing the *OrderablePartNumber/ID* as defined in section 4.5.2 Orderable Part Number – Array above via the keyref element *OrderablePartNumberID*, that a sub-set of the list of parts and represented under the element *PartsSelection/ID*, which is used in the follow on Association section to connect to the technical content.

The concepts of a *PartialPartNumber* along with the *OrderablePartNumberID*, is that by specifying a portion of the full part number, the same group of orderable part numbers could be captured by specifying a particular portion of that part number string that is consistent across all the part numbers that is to be selected for mapping to a particular piece of technical content. For example, instead of specifying all the orderable part numbers, as in:

```
<PartsSelection>
	<OrderablePartNumberID>OrderablePart-ID1</OrderablePartNumberID>
	<PartNumber>TLV74010PDBVR</PartNumber>
	<PartNumber>TLV74018PDBVR</PartNumber>
	<PartNumber>TLV74033PDBVR</PartNumber>
</PartsSelection>
```

A *PartialPartNumber* could be used to reduce the number of Part number entries.

```
<PartsSelection>
	<OrderablePartNumberID>OrderablePart-ID1</OrderablePartNumberID>
	<PartialPartNumber>TLV740</PartialPartNumber>
</PartsSelection>
```

Alternatively, if the *PartialPartNumber* is specified as

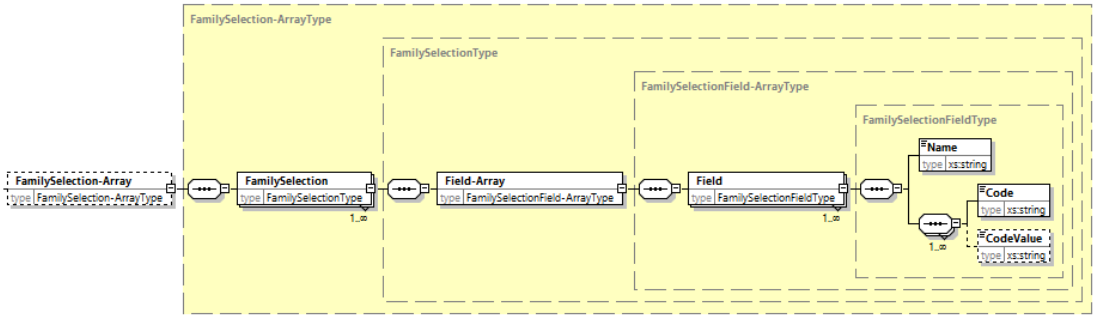
```
	<PartialPartNumber>TLV7401</PartialPartNumber>
```

Then only two of the three part numbers are selected in this example.

```
<PartsSelection>
	<OrderablePartNumberID>OrderablePart-ID1</OrderablePartNumberID>
	<PartNumber>TLV74010PDBVR</PartNumber>
	<PartNumber>TLV74018PDBVR</PartNumber>
</PartsSelection>
```

If using the *PartNumber* along with the *OrderablePartNumberID*, then the part number strings entered must be an exact match of the part numbers entered into section 4.5.2 Orderable Part Number above. This is also enforced by the schemas via the Key and Key Ref concepts.

4.6.1.1 Family Selection - Array

path	**PartModel/PartDetails-Array/PartDetails/PartsSelection-Array/PartsSelection/FamilySelection-Array.**
diagram	
type	**PartsSelection-ArrayType, PartsSelectionType, FamilySelection-ArrayType, JEP30-D10:SignatureDigestLinkType, ds:SignatureType**

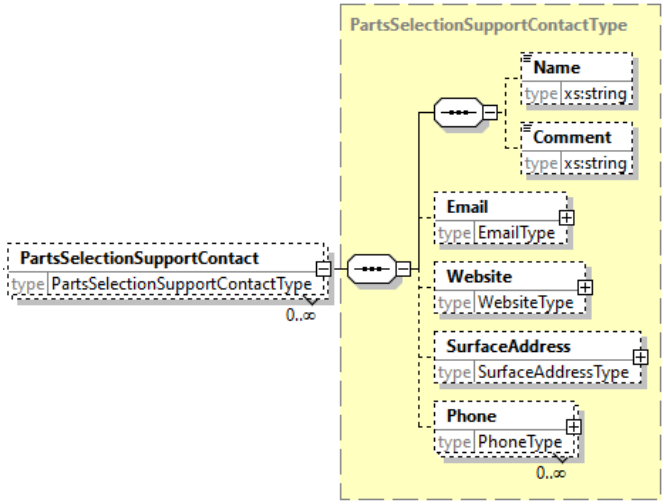
If the selection of parts is from the Part Number Series, then the concepts of a *FamilySelection* that is tied to a specific *PartNumberSeries ID*, is used to select the parts for mapping to the technical content. Because this is connected to the specific part number series, (e.g., all the parts represented on a single datasheet), only the specific *Fields* and their *Code* combinations that makes up a *FamilySelection* needs to be specified in order to be mapped to the technical content. It is not necessary, nor advisable to specify all fields and their corresponding values to compute the full part number string as this is then redundant with section 4.5.1.1 Field – Array above, e.g.,

```
<FamilySelection-Array>
      <FamilySelection>
            <Field-Array>
                  <Field>
                        <Name>Case Code</Name>
                        <Code>0402</Code>
                  </Field>
            </Field-Array>
      </FamilySelection>
</ FamilySelection-Array>
```

is all that is required to connect to the package details for an 0402 package.

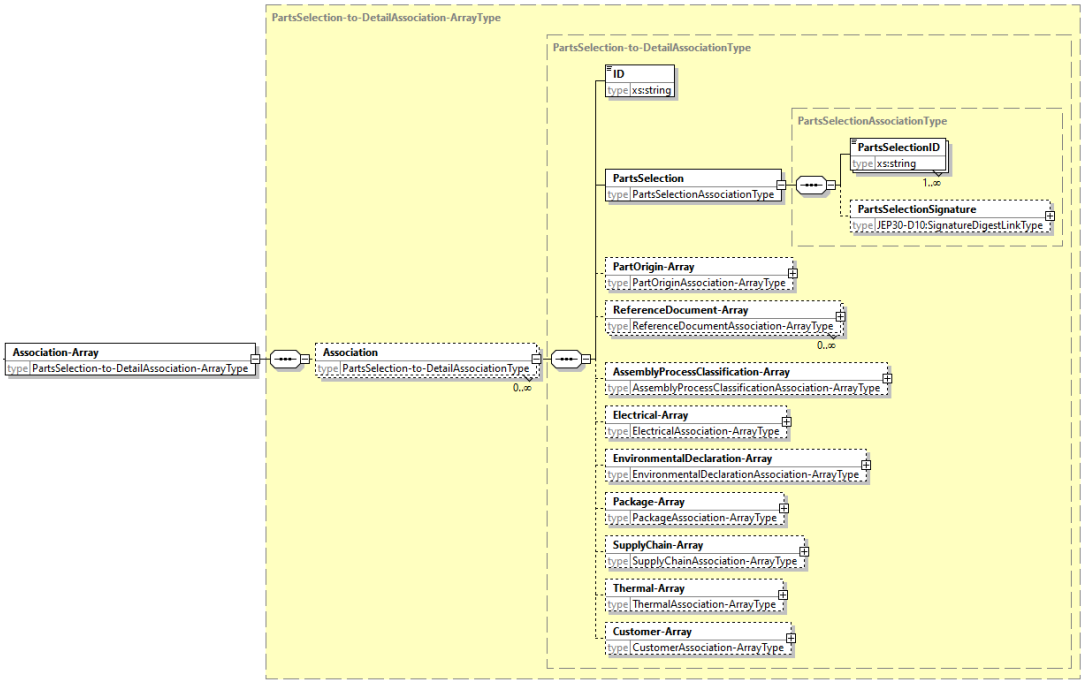
The purpose of the *FamilySelection array* with the embedded nested *Field-Array* provides the ability to select multiple combinations of *Field-Array* groupings, and then connecting that group of families to the same technical content.

4.6.1.2 Parts Selection Support Contact

path	**PartModel/PartDetails-Array/PartDetails/PartsSelection-Array/PartsSelection/PartsSelectionSupportContact**
diagram	
type	**PartsSelectionSupportContactType, EmailType, WebsiteType, SurfaceAddressType, PhoneType.**

This section enables the Component Manufacturer to provide Product support for the parts contained within this PartModel file. The *PartSelection* structure identifies the selection of Parts to which this Support Contact Details applies.

4.6.2 Association - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array.**
diagram	
type	**PartsSelection-to-DetailAssociation-ArrayType, PartsSelection-to-DetailAssociationType, PartsSelectionAssociationType, JEP30-D10:SignatureDigestLinkType, PartOriginAssociation-ArrayType ReferenceDocumentAssociation-ArrayType, AssemblyProcessClassificationAssociation-ArrayType, ElectricalAssociation-ArrayType, EnvironmentalDeclarationAssociation-ArrayType, PackageAssociation-ArrayType, SupplyChainAssociation-ArrayTypeType, ThermalFamilyAssociation-ArrayType, CustomerAssociation-ArrayType.**

The *Association* array is linked to the *PartsSelection-Array* via the *PartsSelectionID*. All links to the technical content shown here are optional, solely for the purpose of enabling the supplier to provide a portion of data (such as Electrical, or Package, or any combination of the technical sections), in one submission to their customer, and still be able to comply with the schema. This is because some customers will have use cases in which only a sub-set of the data is needed, of the component manufacturers may want to hold back certain information because of IP concerns. However, a Part Model is not considered “Complete” for a given use case until all the necessary sections are populated, for the intended use of the PartModel. Note the presence of the “Symbol Family ID” since some part manufacturers may want to provide reference symbols for their part, especially if the part is a complex electrical part.

The Part Model file can support all the data across all the components of the xml schema, for just one Part, or for multiple Parts. The concepts of separating out the MPN from the Family details is based on the following concepts

1. The list of MPN data can be very big.

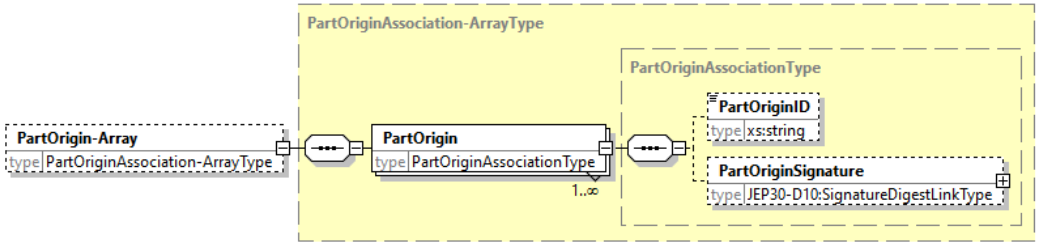
4.6.2 Association - Array (cont'd)

2. There can be many different MPN's that can be inserted into the XML file under the *ManufacturerPartNumber-Array* that can reference the same Thermal Data via the *ThermalFamilyID*.

This concept eliminates data duplicity, minimizes data file size, and makes the transfer of data highly efficient. The linking of different xml components of data is enabled using the respective ID's and verified via Keys and Key Refs.

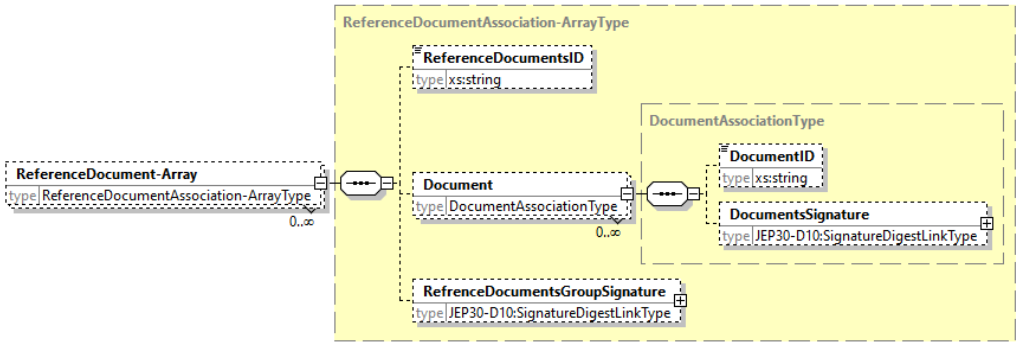
Throughout all the various associations as described below in the sub-sections 4.6.2.x, Signatures with the type *SignatureDigestLinkType* pulls the *DigestMethod* and *DigestValue* when such sections have been digitally signed, into their respective association within this *Association-Array*, thus providing assurance that this association has not been tampered with.

4.6.2.1 Part Origin - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/PartOrigin-Array.**
diagram	
type	**PartOriginAssociation-ArrayType, PartOriginAssociationType, JEP30-D10:SignatureDigestLinkType.**

The *PartOriginID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/PartOrigin-Array/PartOrigin/ID* which maps the part origin data to these parts.

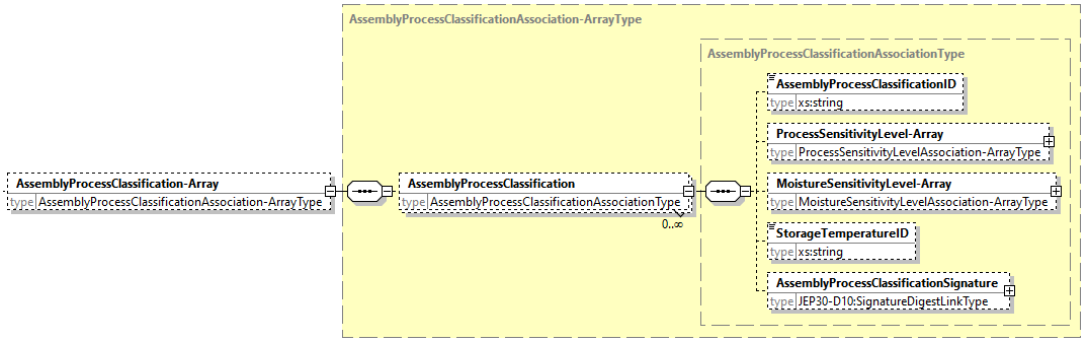
4.6.2.2 Reference Document - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/ReferenceDocument-Array.**
diagram	
type	**ReferenceDocumentAssociation-ArrayType, DocumentAssociationType, JEP30-D10:SignatureDigestLinkType.**

4.6.2.2 Reference Document - Array (cont'd)

The *ReferenceDocument–Array* is structured so that you can reference a single document or an array of documents as described in detail in section 4.7 Reference Manufacturer Part Number – Array below. The *ReferenceDocumentID* via its Key Ref has a reference to the *ReferenceDocuments-Array/ReferenceDocuments/ID* unique key. The *DocumentID* via its Keyref has a reference to the *ReferenceDocuments-Array/ReferenceDocuments/Document/ID* key.

4.6.2.3 Assembly Process Classification - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/AssemblyProcessClassification-Array.**
diagram	
type	**AssemblyProcessClassificationAssociation-ArrayType, AssemblyProcessClassificationAssociationType, ProcessSensitivityLevelAssociation-ArrayType, JEP30-D10:SignatureDigestLinkType, MoistureSensitivityLevelsClassificationAssociation-ArrayType.**

The *AssemblyProcessClassification–Array* is structured so that you can reference a digitally signed combined set of *ProcessSensitivityLevels*, *MoistureSensitivityLevelsClassification*, and *StorageTemperature* as described in detail in document JEP30-A100 Part Model Assembly Process Classification Guidelines for Electronic-Device Packages – XML Requirements.

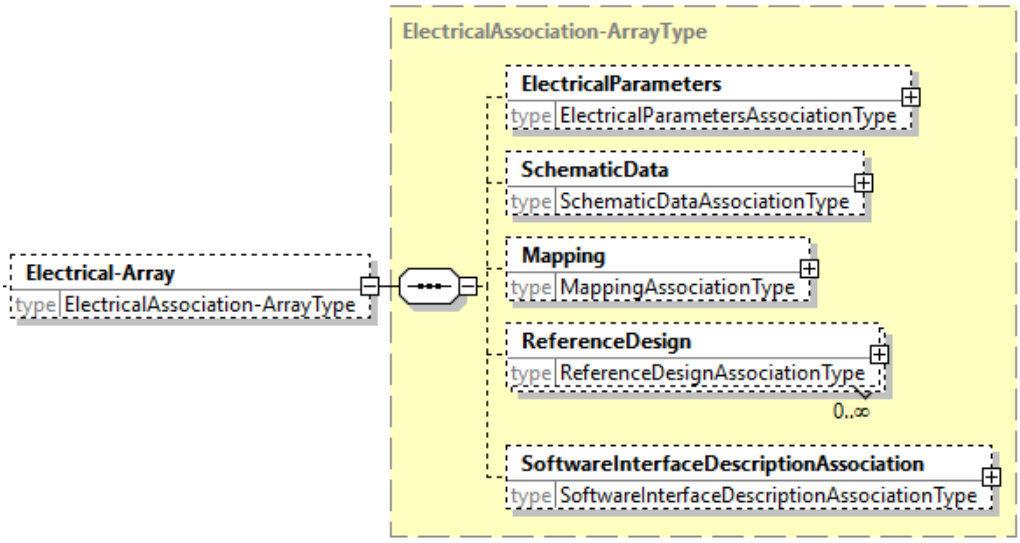
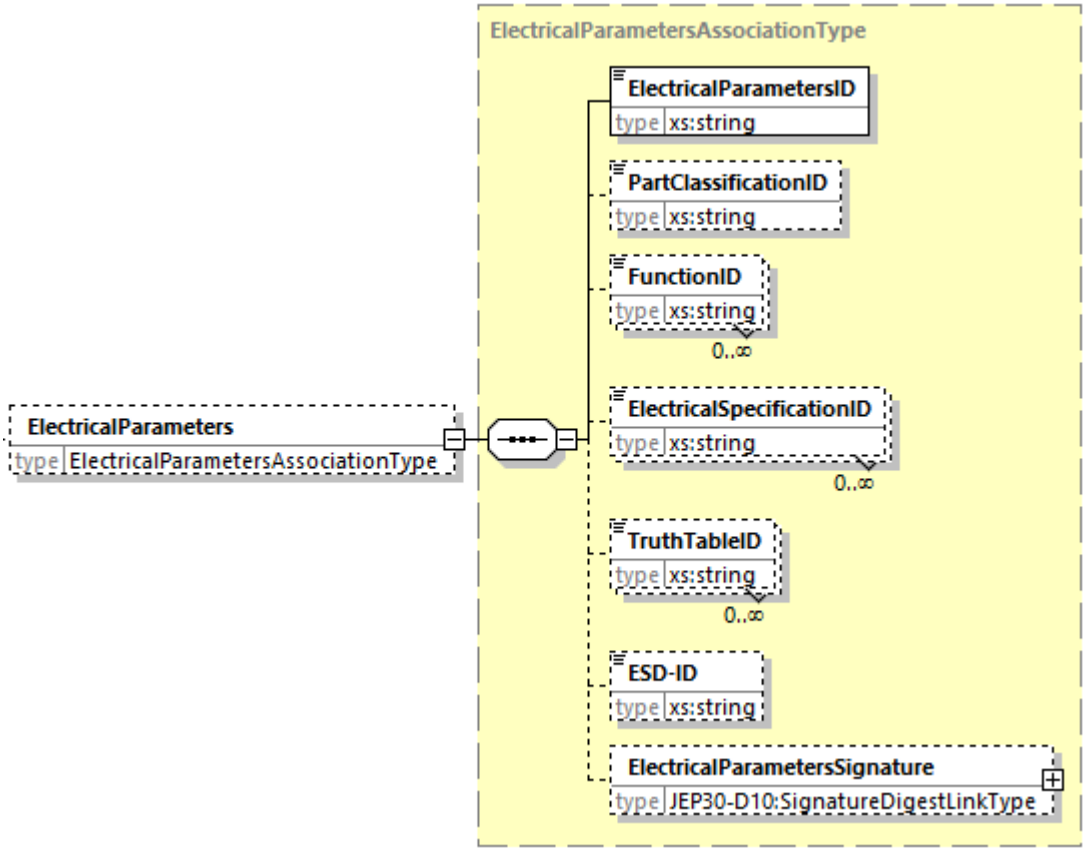
The *AssemblyProcessClassificationID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/AssemblyProcessClassificationSection/AssemblyProcessClassification-Array/AssemblyProcessClassification/ID* which maps assembly process classification data to these parts.

The *ProcessSensitivityLevelsID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/AssemblyProcessClassificationSection/AssemblyProcessClassification-Array/AssemblyProcessClassification/ProcessSensitivityLevel-Array/ProcessSensitivityLevels/ID* which maps process sensitivities levels data to these parts.

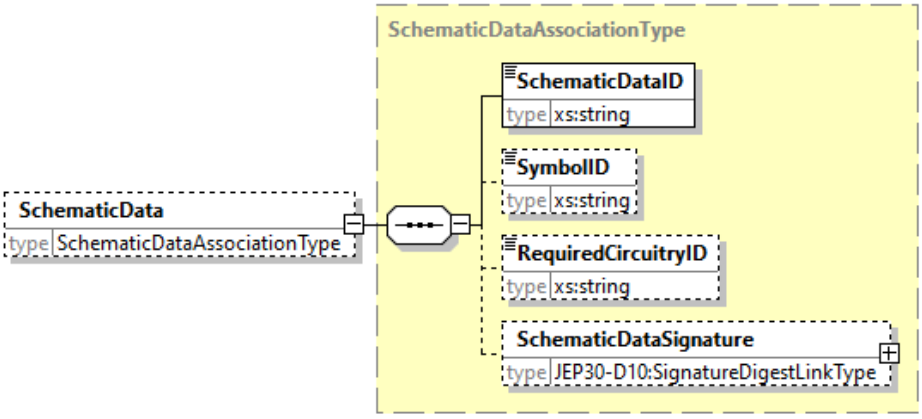
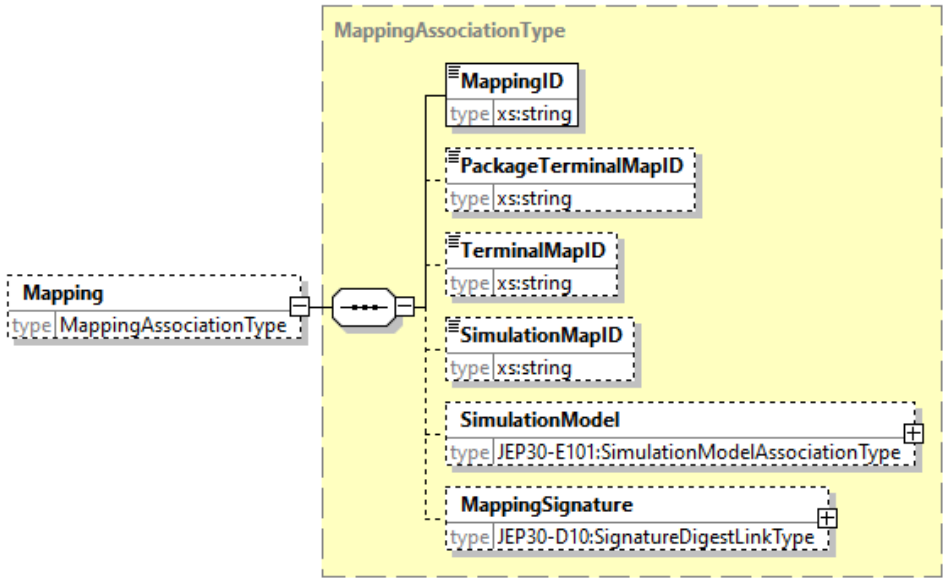
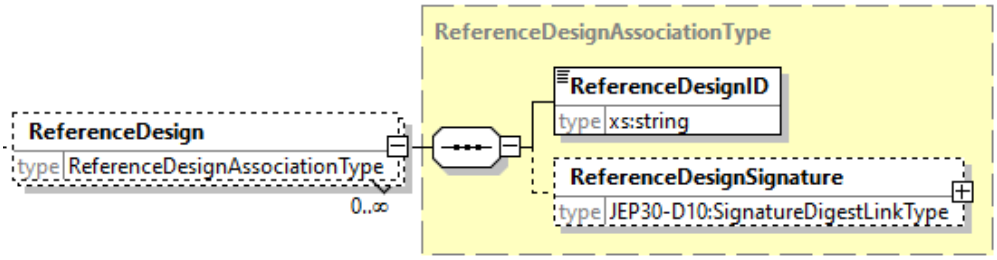
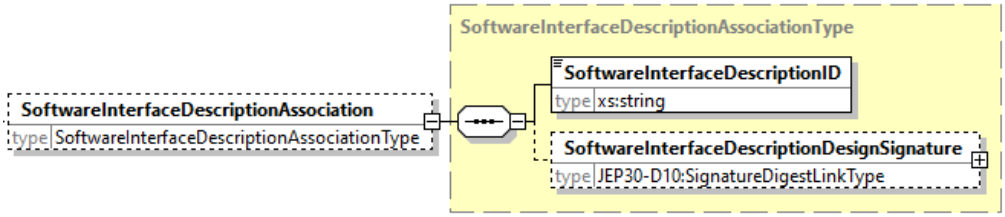
The *MoistureSensitivityLevel ID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/AssemblyProcessClassificationSection/AssemblyProcessClassification-Array/AssemblyProcessClassification/MoistureSensitivityLevel-Array/MoistureSensitivityLevel/ID* which maps moisture sensitivity level data to these parts.

The *StorageTemperatureID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/AssemblyProcessClassificationSection/AssemblyProcessClassification-Array/AssemblyProcessClassification/StorageTemperature-Array/StorageTemperature/ID* which maps storage temperature data to these parts.

4.6.2.4 Electrical - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Electrical-Array.**
diagram part 1 of 6	
diagram part 2 of 6	

4.6.2.3 Electrical - Array (cont'd)

diagram part 3 of 6	
diagram part 4 of 6	
diagram part 5 of 6	
diagram part 6 of 6	
type	**ElectricalAssociation-ArrayType, ElectricalParametersAssociationType, SchematicDataAssociationType, MappingAssociationType, ReferenceDesignAssociationType, SoftwareInterfaceDescriptionAssociationType, JEP30-D10:SignatureDigestLinkType**

4.6.2.3 Electrical - Array (cont'd)

The *Electrical–Array* is structured so that you can reference different types of electrical content, each of which can be digitally signed as described in detail in document JEP30-E100 Part Model Electrical Guidelines for Electronic-Device Packages – XML Requirements.

The *ElectricalParametersID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/ElectricalParameters-Array/ElectricalParameters/ID* which maps the electrical parameters data to these parts.

The *PartClassificationID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/ElectricalParameters-Array/ElectricalParameters/PartClassification-Array/PartClassification/ID* which maps the part classification data to these parts.

The *ElectricalID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/ElectricalParameters-Array/ElectricalParameters/Electrical-Array/Electrical/ID* which maps the electrical data to these parts.

The *ElectricalSpecificationID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/ElectricalParameters-Array/ElectricalParameters/ElectricalSpecification-Array/ElectricalSpecification/ID* which maps the electrical specification data to these parts.

The *TruthTableID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/ElectricalParameters-Array/ElectricalParameters/ElectricalSpecification-Array/TruthTable/ID* which maps the truth table data to these parts.

The *SchematicDataID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/SchematicData-Array/SchematicData/ID* which maps schematic symbol representation data to these parts.

The *SymbolID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/SchematicData-Array/SchematicData/Symbol-Array/Symbol/ID* which maps schematic symbol representation data to these parts.

The *RequiredCircuitryID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/SchematicData-Array/SchematicData/RequiredCircuitry-Array/RequiredCircuitry/ID* which maps the required circuitry data to these parts.

The *MappingID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/Mapping-Array/Mapping/ID* which maps the mapping data to these parts.

The *TerminalMapID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/Mapping-Array/Mapping/TerminalMap/ID* which maps the electrical details to the physical data for these parts.

The *SimulationMapID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/Mapping-Array/Mapping/SimulationMap/ID* which maps the simulation models to the respective areas of these parts.

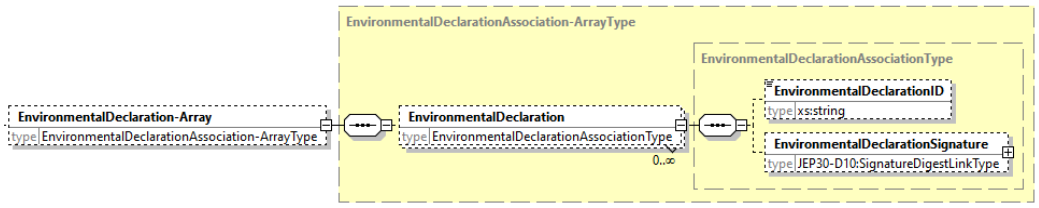
4.6.2.3 Electrical - Array (cont'd)

The *SimulationModelID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/SimulationModel-Array/SimulationModel/ID* which maps the simulation models to these parts.

The *ReferenceDesignID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/ReferenceDesign-Array/ReferenceDesign/ID* which maps example reference designs that these parts could be used in.

The *SoftwareInterfaceDescriptionID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/SoftwareInterfaceDescription-Array/SoftwareInterfaceDescription/ID* which maps internal software description to these parts.

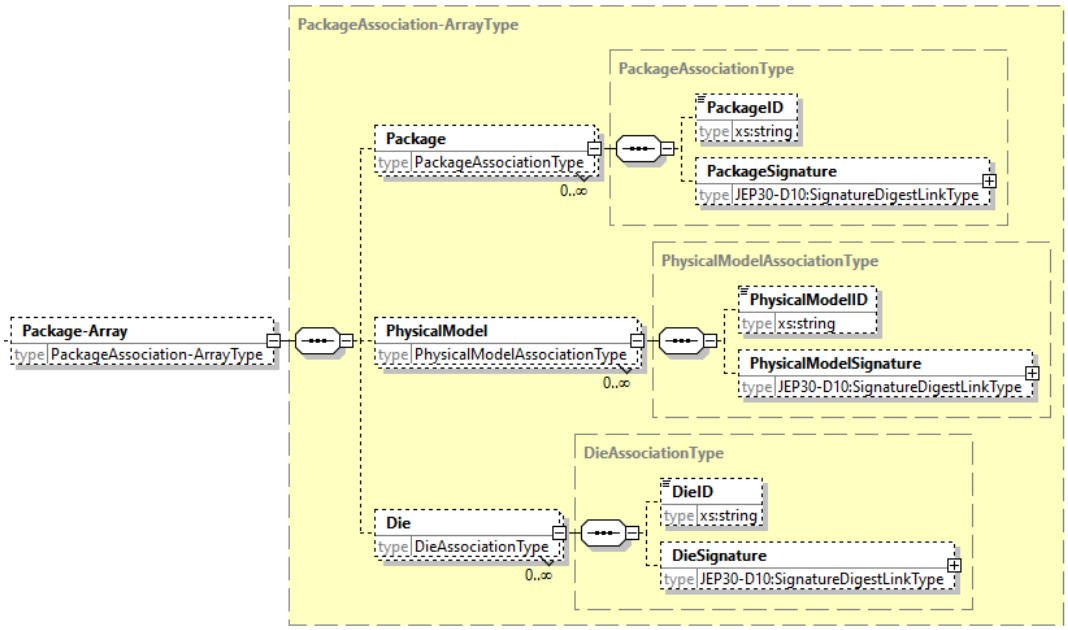
4.6.2.5 Environmental Declaration - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/EnvironmentalDeclaration-Array.**
diagram	
type	**EnvironmentalDeclarationAssociation-ArrayType, EnvironmentalDeclarationAssociationType, JEP30-D10:SignatureDigestLinkType.**

The *Environmental–Array* is structured so that you can reference the environmental declaration content via the Key Ref *EnvironmentalDeclarationID*.

The *EnvironmentalDeclarationID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/EnvironmentalSection/EnvironmentalDeclarationFamily-Array/EnvironmentalDeclaration/ID* which represents the environmental declaration data for these parts.

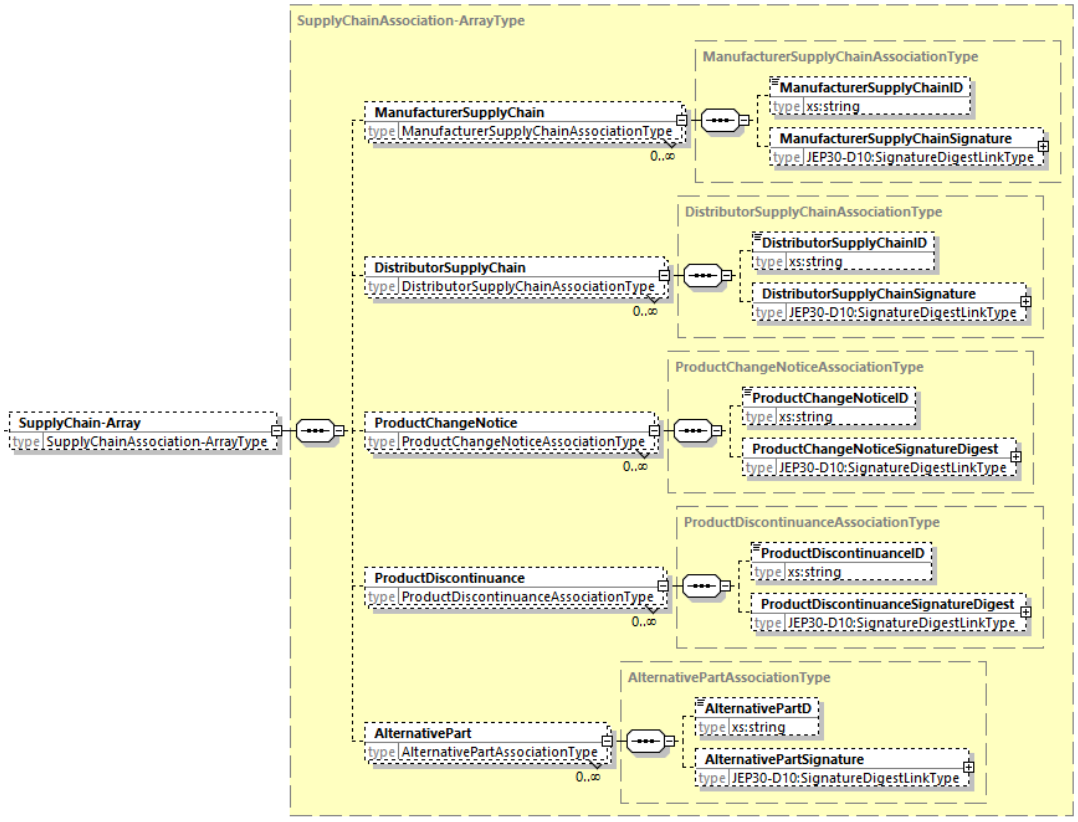
4.6.2.6 Package - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/ Package-Array.**
diagram	
type	**PackageAssociation-ArrayType, PackageAssociationType, PhysicalModelAssociationType, DieAssociationType, JEP30-D10:SignatureDigestLinkType.**

The *Package–Array* is structured so that you can reference different package content, each of which can be digitally signed as described in detail in document JEP30-P100 Part Model Package Guidelines for Electronic-Device Packages – XML Requirements.

The *PackageID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/PackageSection/Package-Array/Package/ID* which represents the device package data.

4.6.2.7 Supply Chain - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/SupplyChain-Array.**
diagram	
type	**SupplyChainAssociation-ArrayType, ManufacturerSupplyChainAssociationType, DistributorSupplyChainAssociationType, ProductChangeNoticeAssociationType, ProductDiscontinuanceAssociationType, AlternativePartAssociationType, JEP30-D10:SignatureDigestLinkType.**

The *SupplyChain–Array* is structured so that you can reference different types of supply chain content, each of which can be digitally signed as described in detail in document JEP30-S100 Part Model Supply Chain Guidelines for Electronic-Device Packages – XML Requirements.

While the various supply chains can reference *ProductChangeNotices* and/or *ProductDiscontinuances* via the *SupplyChainSection*, the association here directly to the Product Change Notices and to the Product Discontinuances can enable higher levels of data size reduction by referencing each Product Change Notices or Product Discontinuances one time via tailored configuration of the *PartsSelectionID*.

The *ManufacturerSupplyChainID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *SupplyChainSection/SupplyChain-Array/ManufacturerSupplyChain/ID* which represents the Manufacturers supply chain data for these parts.

4.6.2.6 Supply Chain - Array (cont'd)

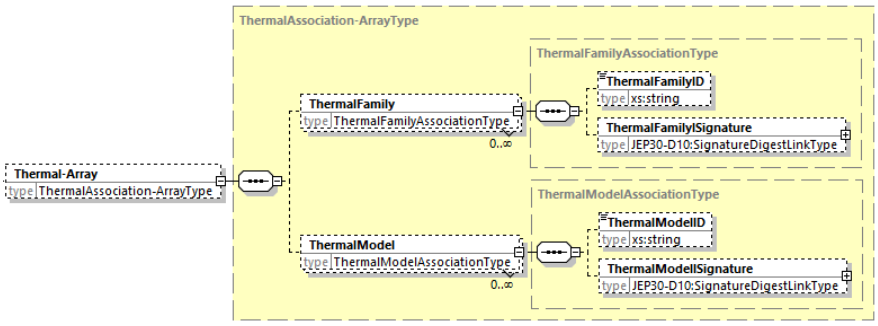
The *DistributorSupplyChainID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *SupplyChainSection/SupplyChain-Array/DistributorSupplyChain/ID* which represents the Distributor supply chain data for these parts.

The *ProductChangeNoticeID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *SupplyChainSection/SupplyChain-Array/ProductChangeNotice/ID* which represents the Product Change Notifications that were issued by the supply chain partners.

The *ProductDiscontinuanceID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *SupplyChainSection/SupplyChain-Array/ProductDiscontinuance/ID* which represents the Product Discontinuances that were issued by the supply chain partners.

The *AlternativePartID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *SupplyChainSection/SupplyChain-Array/AlternativePart-Array/AlternativePart/ID* which maps alternative manufacturer parts to these parts.

4.6.2.8 Thermal Family - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/ThermalFamily-Array.**
diagram	
type	**ThermalFamilyAssociation-ArrayType, ThermalFamilyAssociationType, ThermalModelAssociationType, JEP30-D10:SignatureDigestLinkType.**

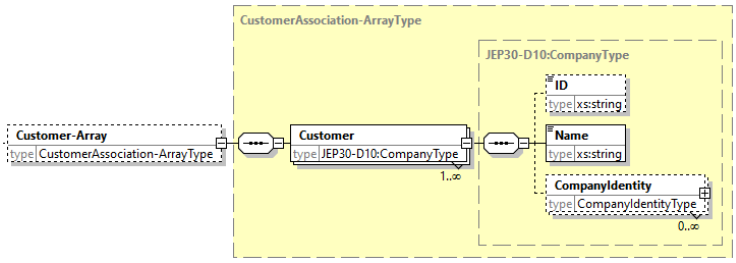
The *Thermal–Array* is structured so that you can reference then Thermal Family details separately from referencing the Thermal Models, each of which can be digitally signed as described in detail in document JEP30-T100 Part Model Thermal Guidelines for Electronic-Device Packages – XML Requirements.

The *ThermalFamilyID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ThermalSection/ThermalFamily-Array/ThermalFamily/ID* which represents the device thermal data.

The *ThermalModelID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ThermalSection/ThermalFamily-Array/ThermalModel/ID* which represents the device thermal model data.

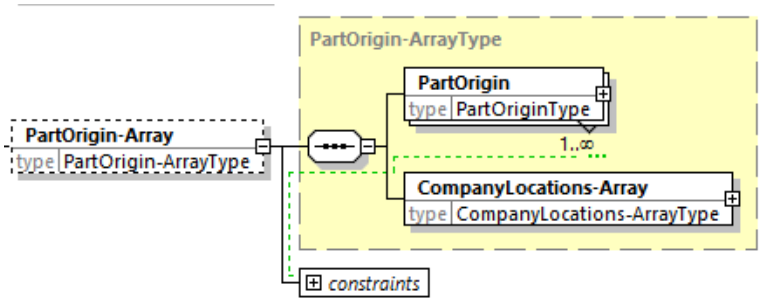
The *ElectricalParametersID* uses a Key Ref to connect the parts as defined by the *PartsSelectionID* to *PartModel/ElectricalSection/ElectricalParameters-Array/ElectricalParameters/ID* which maps the electrical parameters data to these parts.

4.6.2.9 Customer - Array

path	**PartModel/PartDetails-Array/PartDetails/Association-Array/Association/Customer-Array.**
diagram	
type	**CustomerAssociation-ArrayType, JEP30-D10:CompanyType.**

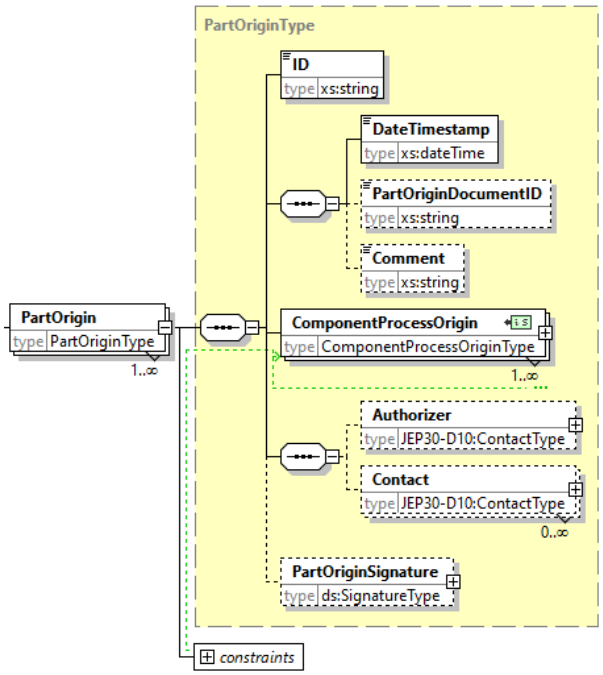
This section enables the component manufacturer to associate a list of Customers to the parts selection as defined by the *PartsSelectionID*.

4.7 Part Origin – Array

path	**PartModel/PartOrigin-Array**
diagram	
type	**PartOrigin-ArrayType, PartOriginType, ComponentSuppliers-ArrayType.**

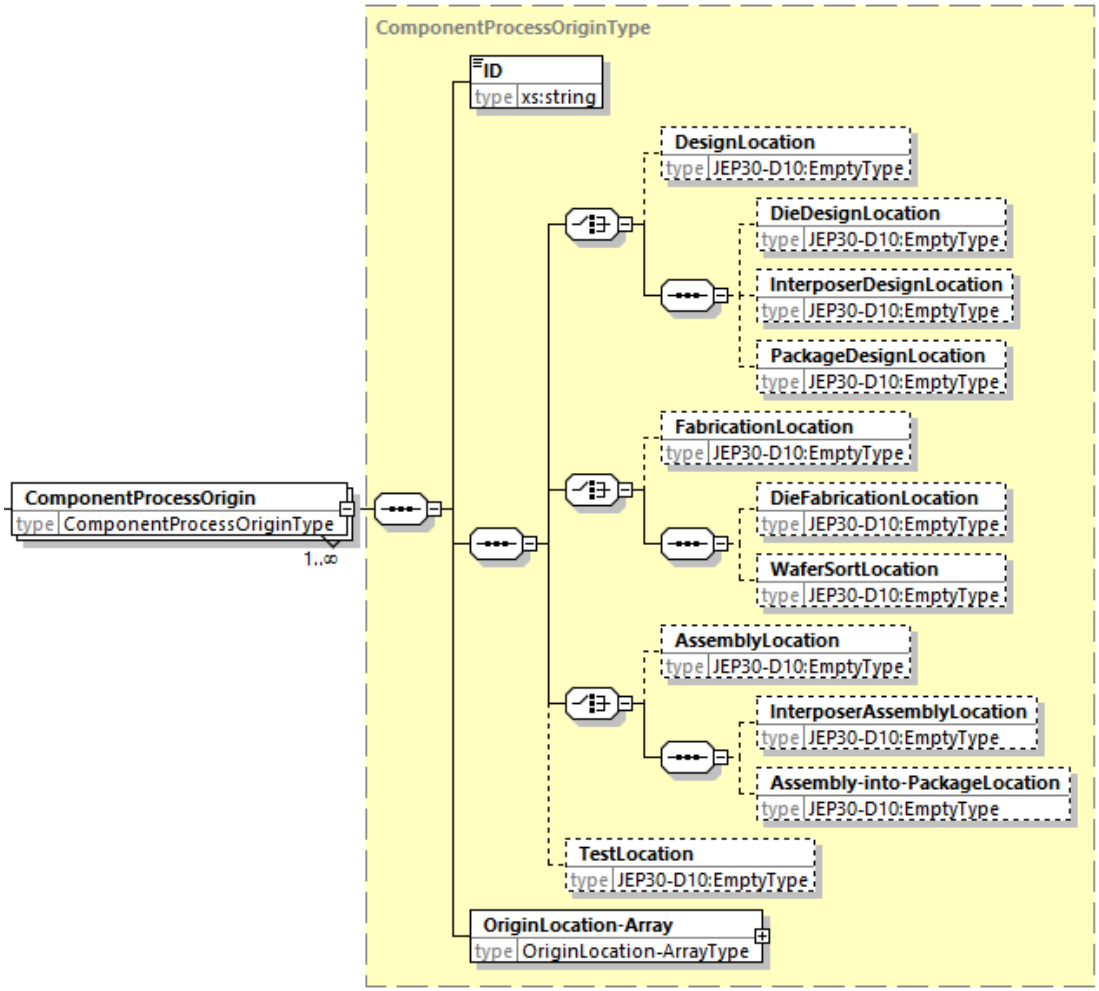
The *PartOrigin–Array* is intended to capture the origin location details for various component processes involved in the design and manufacturing of the part.

4.7.1 Part Origin

path	**PartModel/PartOrigin-Array/PartOrigin**
diagram	
type	**PartOriginType, ComponentProcessOriginType, JEP30-D10:ContactType, JEP30-D10:SignatureDigestLinkType.**

The *PartOrigin* is intended to capture all the reported source location details for various design and manufacturing components of the part.

4.7.1.1 Component Process Origin

path	**PartModel/PartOrigin-Array/PartOrigin/ComponentProcessOrigin**
diagram	
type	**ComponentProcessOriginType, JEP30-D10:EmptyType, OriginLocation-ArrayType.**

The *ComponentProcessOrigin* location can be captured for any of the four major process steps in the creation of a part, namely, Design, Fabrication, Assembly and Test. Within these major process steps, there are sub-process that can be sometimes performed at different locations, and even by different companies. This structure provides the means of defining the company location to each of the process steps.

If the design function is performed at different locations, then the option is available to separately define the location for each sub-design process for the Die, Interposer and Package. This is more typical, since as devices become more complex, a device may contain several different dies, that are mounted onto a silicon interposer, which can then be assembled onto an organic interposer with other components. These dies may come from different manufacturers. The assembly onto a silicon interposer is typically done in a foundry facility, whereas the assembly onto an organic interposer can be done at a non-foundry facility.

4.7.1.1 Component Process Origin (cont'd)

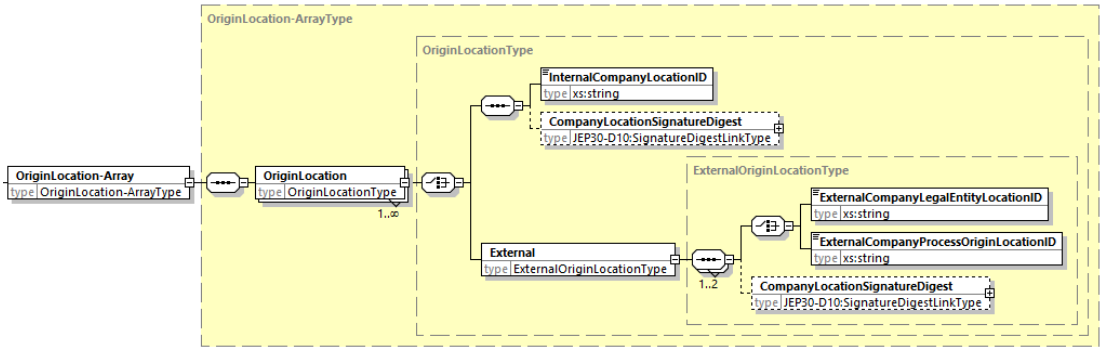
When integrating into a package, many times the OSAT facility will have pre-defined package styles that have been out-sourced to a design facility, or they may perform the design internally or by the Component Manufacturer. Custom packages may be designed in conjunction with the Customer Product so that the terminal locations are strategically placed to match the requirements of the higher-level assembly to which the part will be assembled into.

Die fabrication is performed at a foundry on wafers. If the integration of the die is being performed into the next higher-level assembly at the same location, then typically the wafer sort operation is performed at that same location. However, there are times when the wafer is shipped to other locations or companies where wafer dicing is performed.

Similar to the previous process steps, assembly at various levels can be performed at the one location or at different locations depending upon the type of assembly operation.

After the assembly is complete, the component is tested before release from production. This can be also located at a testing facility that is different to the locations of the previous process.

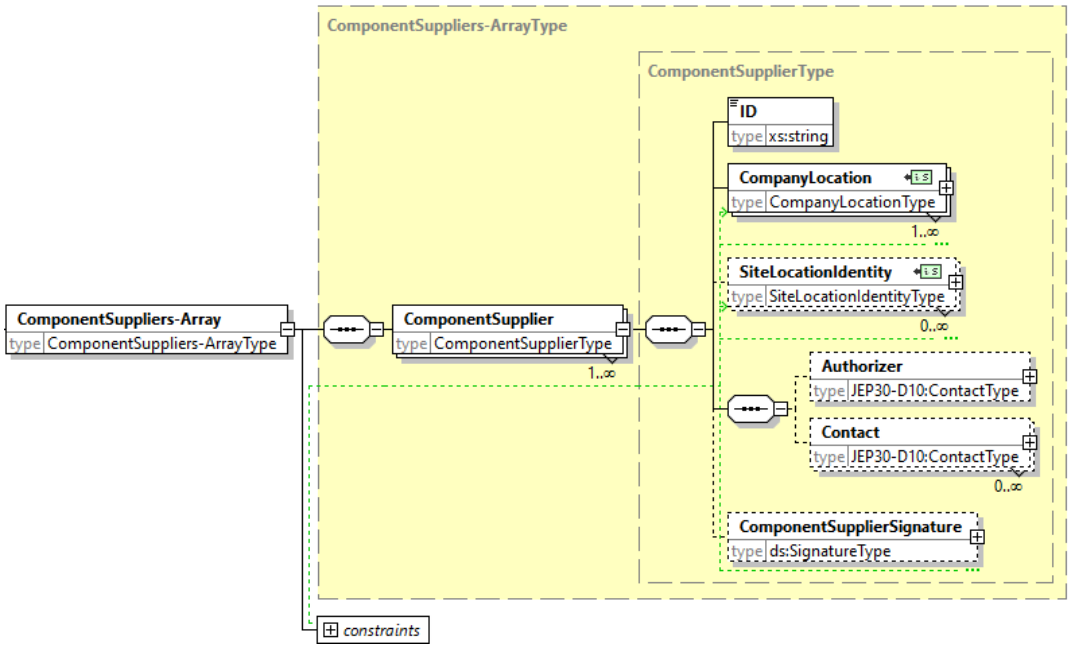
4.7.1.1.1 Origin Location - Array

path	**PartModel/PartOrigin-Array/PartOrigin/ComponentProcessOrigin/OriginLocation-Array**
diagram	
type	**OriginLocation-ArrayType, OriginLocationType , ExternalOriginLocationType ,** **JEP30-D10:SignatureDigestLinkType.**

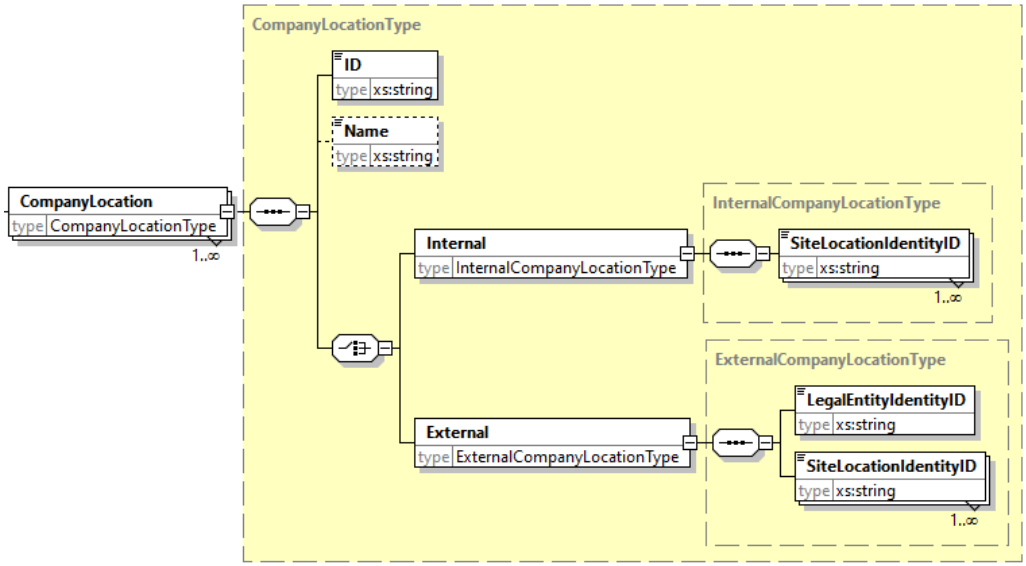
The *InternalCompanyLocationID*, *ExternalCompanyLegalEntityLocationID*, and the *ExternalCompanyProcessOriginLocationID* connects to the company location details under the *CompanyLocations-Array/CompanyLocation/ID*.

It is recommended that the *PartOrigin* details contain the *Contact* and *Authorizer* details and that this structure is digitally signed off via the *PartOriginSignature*. This will help to build trust in the supply chain for the part.

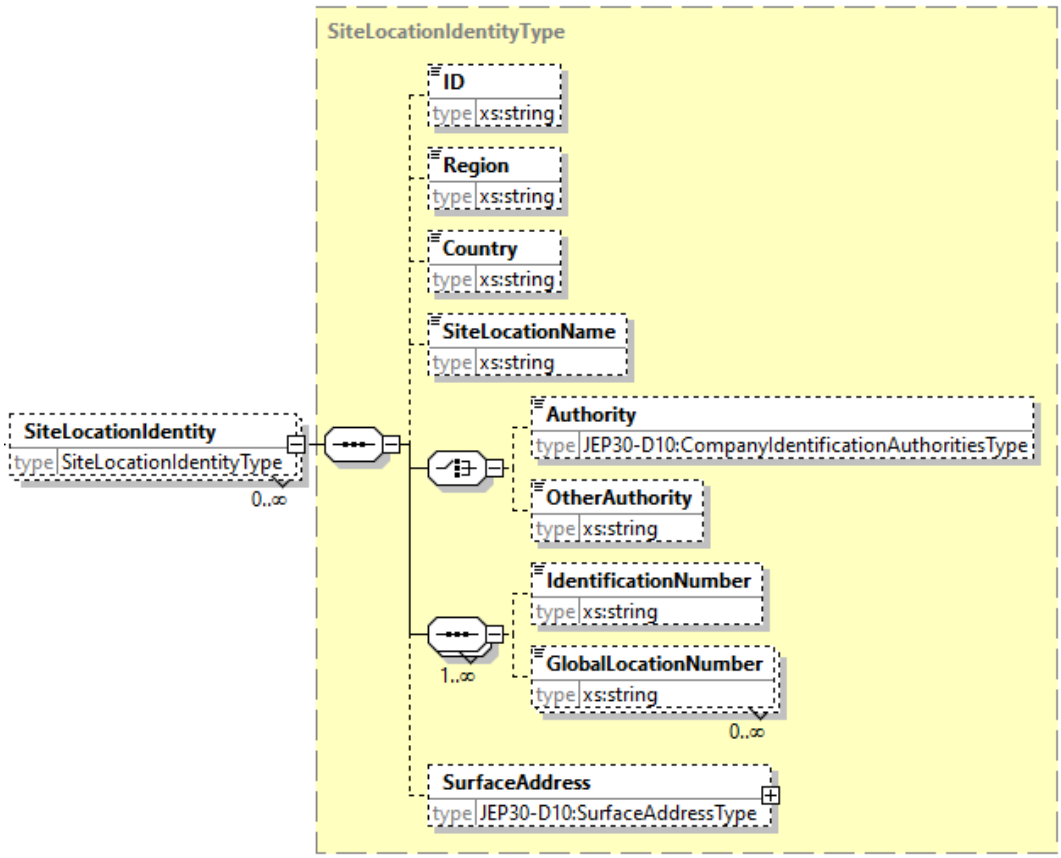
4.7.2 Component Suppliers - Array

path	**PartModel/PartOrigin-Array/ComponentSuppliers-Array**
diagram	
type	**ComponentSuppliers-ArrayType, ComponentSupplierType, CompanyLocationType, SiteLocationIdentityType, JEP30-D10:ContactType, ds:SignatureType.**

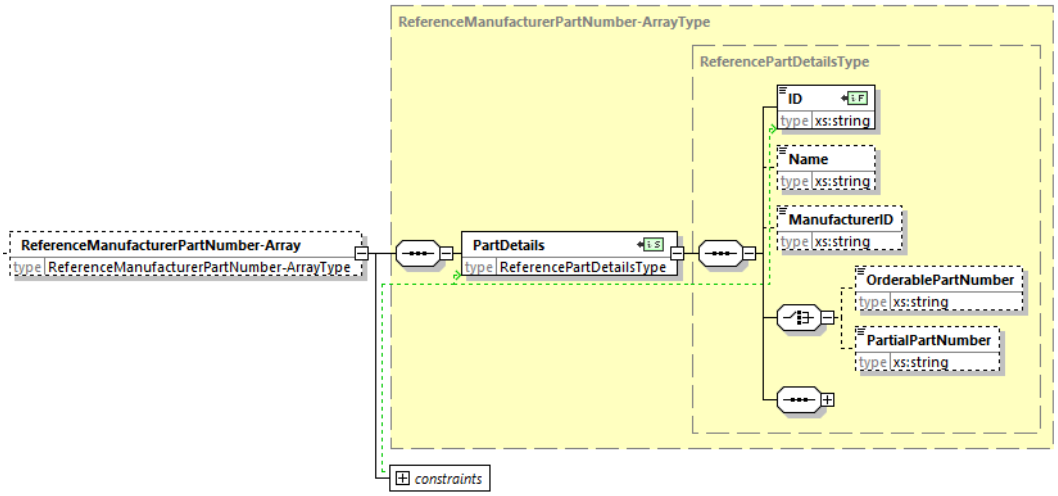
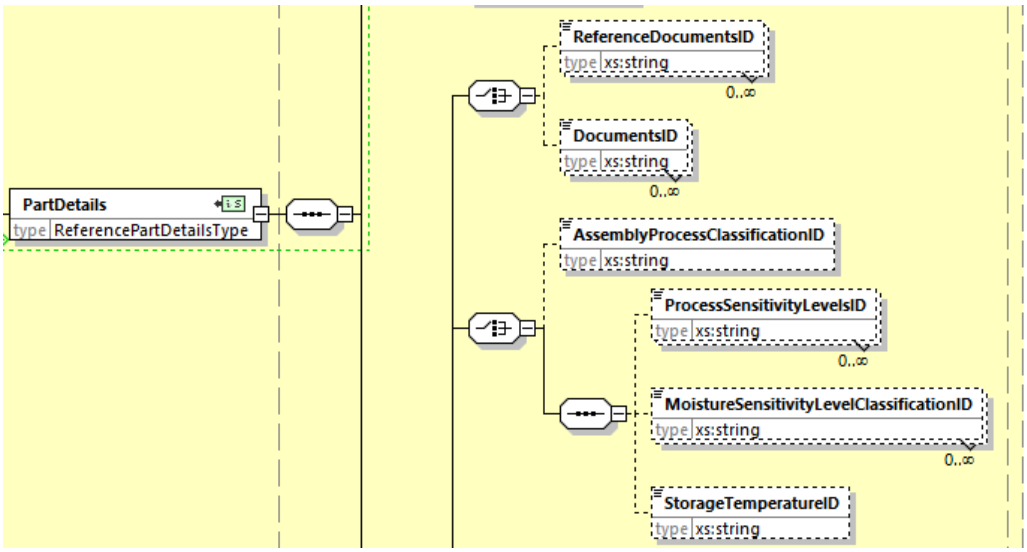
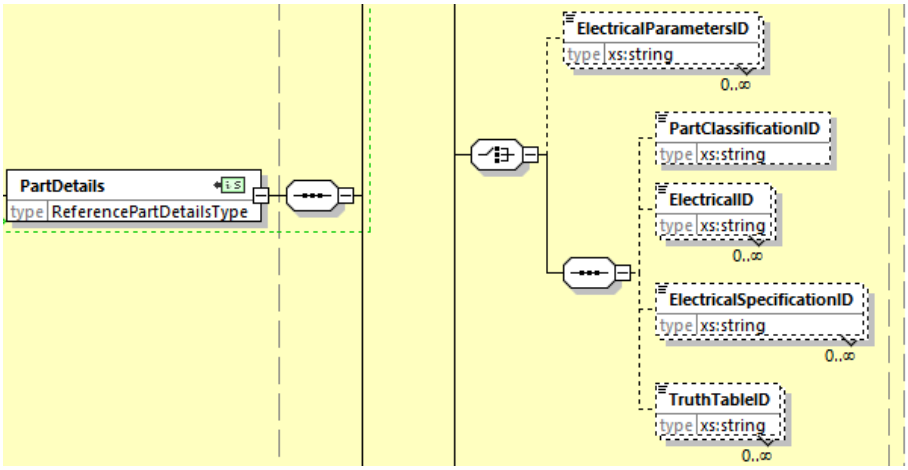
4.7.2.1 Company Location

path	**PartModel/PartOrigin-Array/ComponentSuppliers-Array/ComponentSupplier/CompanyLocation**
diagram	
type	**CompanyLocationType, InternalCompanyLocationType, ExternalCompanyLocationType.**

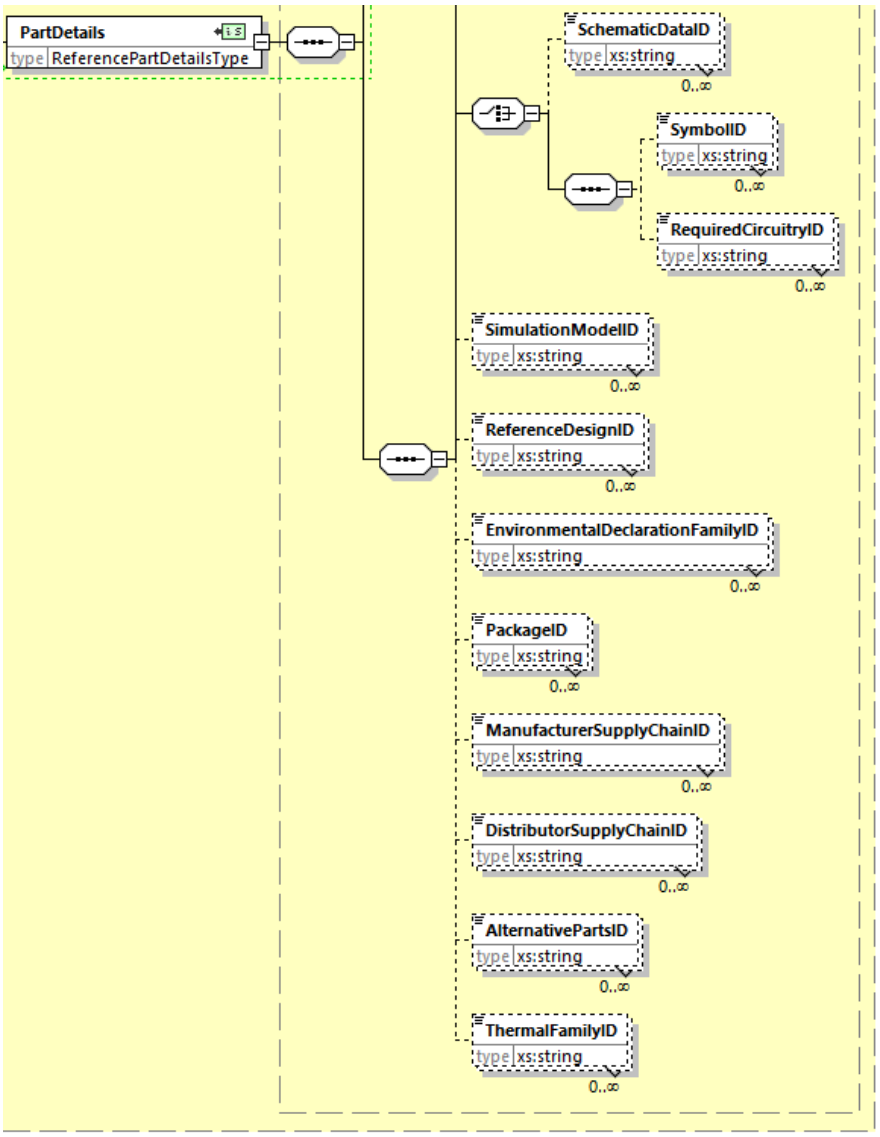
4.7.2.2 Site Location Identity

path	**PartModel/PartOrigin-Array/ComponentSuppliers-Array/ComponentSupplier/SiteLocationIdentity**
diagram	
type	**CompanyLocationType, InternalCompanyLocationType, ExternalCompanyLocationType.**

4.8 Reference Manufacturer Part Number – Array

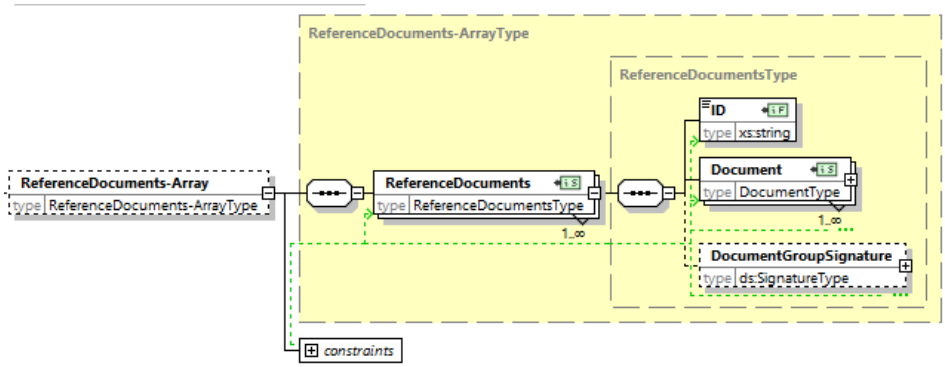
path	**PartModel/ReferenceManufacturerPartNumber-Array**
diagram part 1	
diagram part 2	
diagram part 3	

4.8 Reference Manufacturer Part Number - Array (cont'd)

diagram part 3	
type	**ReferenceManufacturerPartNumber–ArrayType, ReferencePartDetailsType, EquivalencyPartType.**

As shown in the Electrical section, a part manufacturer may provide a Reference Design for a given Part. This Reference Design will include one or more parts, which may be included into this XML file. These parts are referred to as "Reference Part Numbers". Parts listed in the XML file under this section, may only have limited technical data populated, since the intent of the Part Manufacturer who is providing the reference design, may be to only specify the critical technical detail, (such as a 10K Resistor, but the part number, manufacturer, and other technical details of that part is irrelevant, and left up to the Product designer to choose). Therefore, all elements here are optional, including the actual identification of the reference part.

4.9 Reference Documents - Array

path	**PartModel/ReferenceDocuments-Array**
diagram	
type	**ReferenceDocuments–ArrayType, ReferenceDocumentsType, DocumentType, ds:SignatureType.**

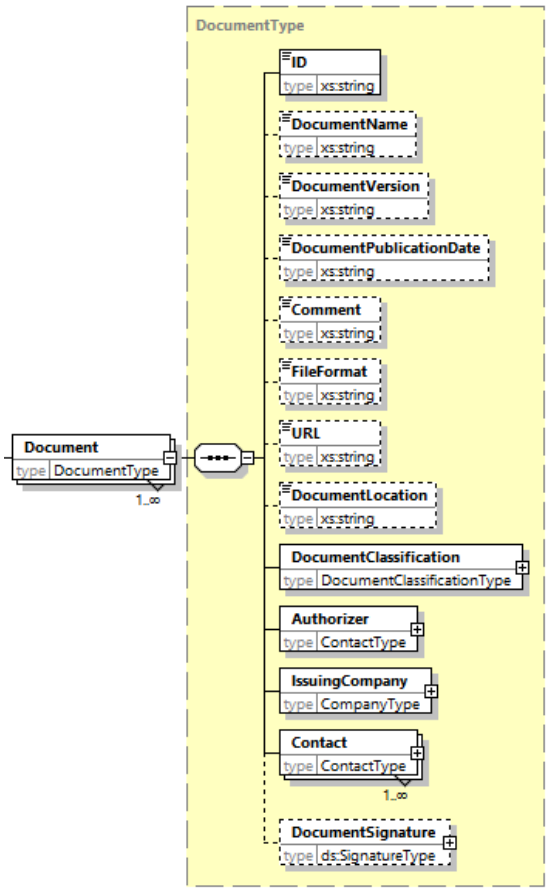
The *ReferenceDocuments–Array* is structured so that you can reference a single document or an array of documents that belongs to the parts identified by the *PartsSelectionID*. Typically, in this association, all the parts that are identified in the parts identity section would point to a single datasheet file in which case the *PartsSelectionID* is configured to represent all the parts as represented by either the *PartNumberSeriesID* or the *OrderablePartNumberID*.

Sometimes component manufacturers will have a library of reference documents that could be applicable to multiple parts or multiple part families. An example of such is where component manufacturers standardize on a set of package definitions for use across their entire set of parts that they offer to the market. In these situations, very often the datasheet will then refer to other documents to represent the package physical properties. In a datasheet that represents parts that map to different packages, then *PartsSelectionID* is configured to represent only those parts that point to a specific package document (i.e. QFP), and then a second Association is created to represent the next set of parts via a different *PartsSelectionID* that is now configured to represent the next group of parts that point to the next package document (i.e. QFN).

If multiple part families (as defined by multiple *PartNumberSeriesID* or the *OrderablePartNumberID*) are contained within the same PartModel file (i.e., from multiple datasheets), then it is best to store all documents that are shared by these part families into a single reference document array instance. This will reduce the duplication of documents that represent multiple part families. The specific documents that point to a single part family (as defined by a single *PartNumberSeriesID* or the *OrderablePartNumberID*), and that are not shared with other part families that are defined in the PartModel file, should be represented in a single Reference Document array that is represented by the *PartsSelectionID* that is configured to this single *PartNumberSeriesID* or to this single *OrderablePartNumberID.*

The collection of documents under a *ReferenceDocuments–Array* can be optionally digitally signed to provide the consumer a higher degree of confidence that the documents contained within the array are un-tampered.

4.9.1 Document

path	**PartModel/ReferenceDocuments-Array/ReferenceDocuments/Document**
diagram	
type	**DocumentType, DocumentClassificationType, ds:SignatureType.**

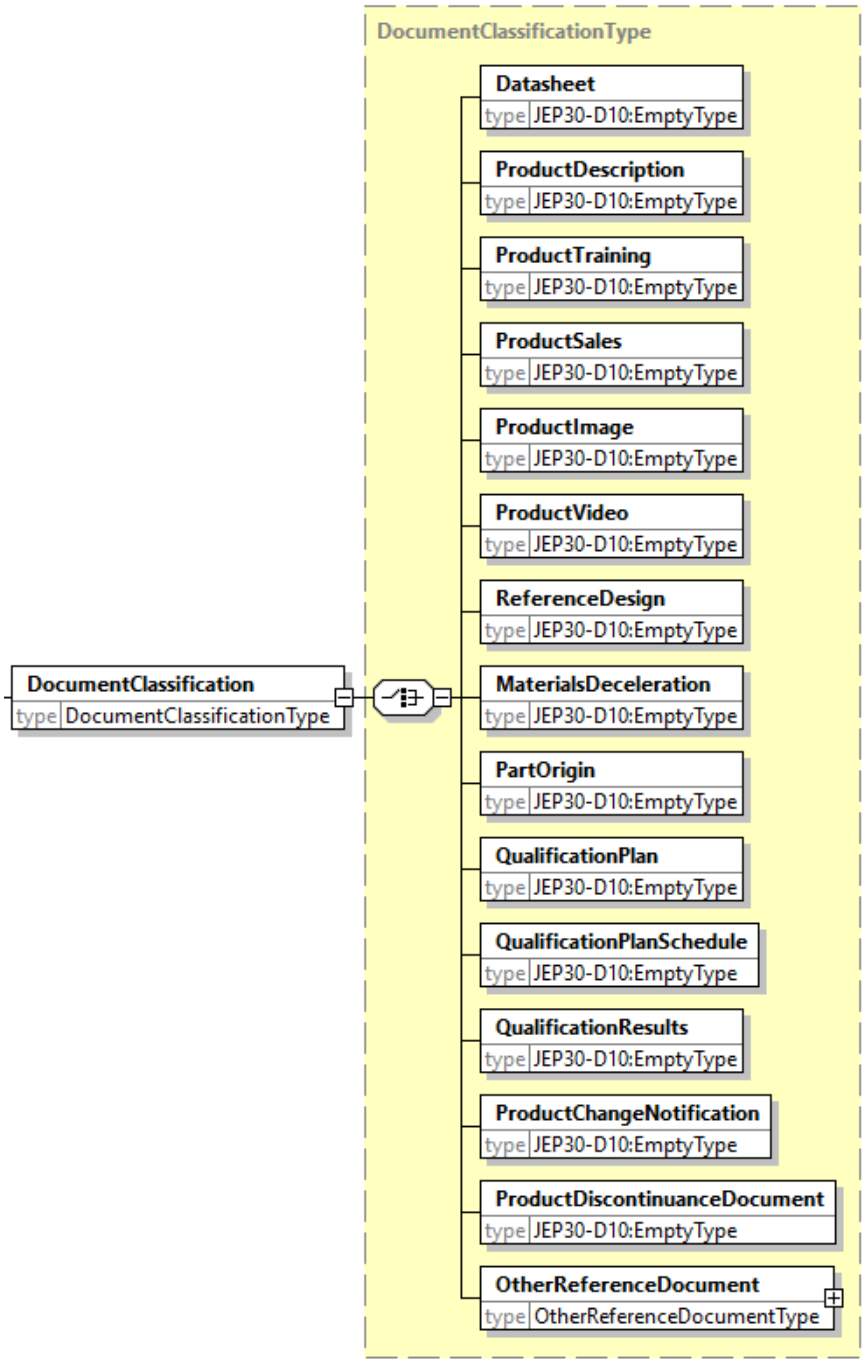
Each individual document defined under the *ReferenceDocuments–Array* is required to be accompanied with a set of meta data that identifies the classification of the document as defined in section 4.8.1.1 Document below. Additional metadata is required to define the Authorizer of the document, the details of the Issuing Company, and a point of contact for questions specific to that document. This metadata provides a method of traceability for their document, and provides a level of confidence to the consumer, especially when the document is also accompanied with a digital signature to prevent future tampering.

There is also a set of optional metadata that can be captured, as shown in the diagram above. It is preferred although not mandatory, to provide the *DocumentName*, *Version*, and *DocumentPublicationDate*, so that customers can cross reference the version of the part identity (via the *PartNumberSeries/Version* or the *OrderablePartNumber/Version*) to the version of the document. This helps to reduce the incidences in which customers design products to one version of the part and later find that a different version of the part procured through the supply chain does not match their product design, driving high wastage costs for the customer.

4.8.1 Document (cont'd)

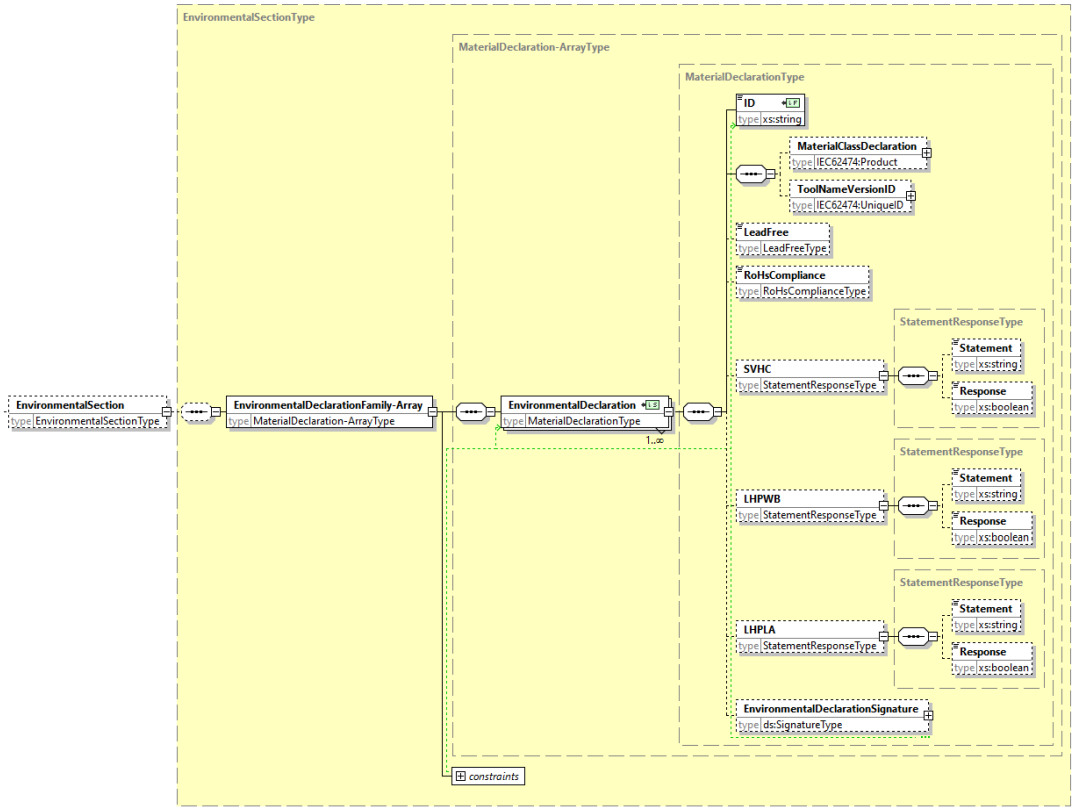
FileFormat simply provides information as to the format of the file such as pdf, xlsx, xml, etc. When component manufacturers store documents in their cloud, then the URL link to that specific version of the document should be recorded in the *URL* field. However, some component manufacturers maintain the same URL link to all version of the document and simply replace a previous version of the document with the newer version of that document at the same URL link. Such situations create a disconnect between the submitted Part Model that represents Version (A) to a document in the URL link that refers to Version (B). It is therefore advisable that Part Model files submitted are accompanied with actual copies of the document when the tracking and alignment of the Part Model version Document Version needs to be maintained. In those situations, the *DocumentLocation* field can be used where the consumer of the part model can obtain the specific version of the documents referred to within this Part Model file.

4.9.1.1 Document Classification

path	**PartModel/ReferenceDocuments-Array/ReferenceDocuments/Document/DocumentClassification**
diagram	
type	**DocumentClassificationType.**

All documents referred to within the *ReferenceDocuments-Array* are required to be classified in accordance with the above diagram. If a document is provided that does not fit into one of the pre-defined categories, then the type and name of the document can be stored under the "*OtherReferenceDocument*" element.

4.9.1.2 Environmental Section

path	**PartModel/EnvironmentalSection**
diagram	
type	**EnvironmentalSectionType, MaterialDeclaration-ArrayType, MaterialDeclarationType, IEC62474:Product, IEC62474:UniqueID, LeadFreeType, RoHsComplianceType, SVHCType, LHPWBType, LHPLAType, ds:SignatureType**

The electrical and electronics industry and its supply chain use material declarations to track and declare specific information about the material composition of its products. To harmonize requirements across the supply chain and to improve economic efficiencies, IEC 62474 provides an International Standard for the exchange of material composition data and provide requirements for material declarations.

The JEP30 Part Model integrates the IEC 62474 standard by importing the IEC62474 standard schema into the Part Model schema. It then refers to the Material Declaration via the *MaterialClassDeclaration* element using the type "*iec:Product*" which references the type called *Product* in the IEC 62474 schema. The structure below this *MaterialClassDeclaration* element follows the IEC 62474 schema and is documented in the IEC 62474 standard. Updates to the IEC 62474 standard schema will automatically reflect here within this Part Model schema, so customers always have access to the latest version of the material declaration standard as published by IEC.

Component Manufacturers are now able to include their material declaration along with other part related technical content in one submission to their customers. In addition, if multiple part numbers as defined via the *PartNumberSeriesID* or the *OrderablePartNumberID* have the same material declaration, this can now be facilitated via the *PartsSelectionID*.

Annex A Workaround for tools that fail to consume Key Refs

The Part Model schema declares many Keys and Key Ref constraints that enforce the uniqueness of identifiers and the integrity of references to those identifiers. These constraints are essential for validating the correctness of Part Model XML documents.

Unfortunately, some well-used schema tools (most notably the xerces library and the programs that are built on top of it) fail at validating some correct documents because of these constraints. As a workaround, the Key Ref constraints can be removed from the schema files to bypass the limitations in those tools.

The Key Ref constraints can be removed by executing the following XQuery script against all the schema files:

```
declare function local:rm-keyrefs($nodes as node()*) as node()* {
  for $node in $nodes
  let $is-elem := $node instance of element()
  return
    if ($is-elem and name($node) = 'xs:keyref') then ()
    else if ($is-elem) then element { node-name($node) } { $node/@*, local:rm-keyrefs($node/node()) }
    else if ($node instance of document-node()) then local:rm-keyrefs($node/node())
    else $node
};

for $doc in root()
return local:rm-keyrefs($doc)
```

XQuery is a standard language for describing queries and transformations of XML documents. Most XML processing tools are bundled with an XQuery engine, which can be used to run the script above for all the schema files.

In order to be compliant with the standard, XML documents that are generated using tools that consume a schema without Key Refs still need to be validated against the original schema. One freely available program that has been verified to validate Part Models appropriately is xmllint, which is part of the libxml project.

Annex B (informative) Differences between JEP30 and its predecessors

This table briefly describes most of the changes made to entries that appear in this standard, JEP30, compared to its predecessor; Punctuation changes may or may not be included.

Change Record History		
Initial Issue:	Date: February 2018	Item Number: 11.2-938

Issue: A	Date: March 2023	Item Number: 11.2-938S
Description of changes		
Section 1 Scope: Updated scope to include Material Deceleration and Supply Chain		
Section 2.1 Applicable Documents: Added in reference to JEP30-S100 and JEP30-E101 documents.		
Section 3.2.1, Page 8: Updated explanation on *EmptyType*.		
Section 3.2.2 Sample XML Schema Types: Updated image to show *EmptyType*		
Section 3.2.3, Page 10: Add new section to explain the Key Refs in the schema		
Section 4.1 Part Model: Updated *PartModelType* to show the addition of Supply Chain section.		
Section 4.2, Page 16: Added “EmptyType” to elements “Distribute” and “Request-Reply”		
Section 4.2.1, Page 17: Updated Contact type to include optional Website information		
Section 4.5.1 Table 3, Page 22: Add in row with count of enumerated values in the list for each field.		
Section 4.5.1.1, Page 24: Added in “CodeValue” since some codes can be converted to values.		
Section 4.5.1.2, Page 26-28: Added in examples of how the combination constraints works.		
Section 4.5.1.3, Page 30: Updated Orderable Part Number branch to remove Part Details ID and to add in Manufacturer ID. The linking of the Orderable Part Number to the Technical content is moved to the Part Details branch to be consistent with the Part Number Construction concept.		
Section 4.5.1.3, Page 31: Updated Orderable Part Number XML example		
Page 31: Moved the “Linking the Manufacturer to the Manufacturer Part Number” from section 4.6 to section 4.5.1.4, since the Manufacturer ID is moved to the Orderable Part Number section.		
Section 4.6, Page 32: Updated the Part Details section for selecting the Part Numbers that maps to the technical content. Changed “Family” to “FamilySelection” to enable the verification of data via unique keys and Key Refs. Moved Equivalency Parts to the new Supply Chain section in JEP30-S101.		
Section 4.12, Page 42: Added in “Other Reference Documents” in under Reference Documents to enable different types of non-standard documents to be provided along with the PartModel		
Added Annex A: Provide a workaround for tools that fail to consume unique keys and Key Refs		

Issue: B	Date: August 2023	Item Number: 11.2-1032
Description of changes		
Introduced a Foreword and Introduction at beginning of document		
Section 4.5.1 and section 4.5.2: Add Base Part Number, Functional Part Number and part Description into Part Number Series and orderable Part Number		

Annex B (cont'd)

Issue: C	Date: November 2023	Item Number: 11.2-1040
Description of changes		
Updated Table 2 with the revisions of the next release		
Section 4.6.2.3, Updated the Electrical-Array association with Software Interface Description Association		

Issue: D	Date: February 2024	Item Number: 11.2-1053
Description of changes		
Section 3.2.2, Add in definition of "Group".		
Section 4.3 Manufacturer – Array: Updated Manufacturer to include "Standards Organization Identity"		
Section 4.5 Manufacturer Part Number – Array: Updated Manufacturer Part Numbers to include "Standards Identifier"		
Section 4.5.3: Added new section for Standards Identifier		
Section 4.6.2.5, Updated the Package-Array association with Physical Model and Die Association.		

Issue: E	Date: August 2024	Item Number: 11.2-1059
Description of changes		
Section 4.1 PartModel: Add PartModel Status attribute and new element for PartOrigin-Array		
Section 4.1.1 Schema Release and Versioning: Update Table 2		
Section 4.2 Business Info: Added Declaration section		
Section 4.2.2.1 Contact Type: Added details about contact details.		
Section 4.2.2.2 Company Type: Updated image for Company Type		
Section 4.3.1 Manufacturer Identity: Updated diagram.		
Section 4.5 Manufacturer Part Number – Array: Inserted Future Part and updated description paragraph.		
Section 4.5.2 Orderable Part Number: Updated the image to make the Part Number optional		
Section 4.5.3 Future Part: Added new section for Future Part with explanation		
Section 4.5.4 Orderable Part Number: Updated the image to make the Standard Number optional		
Section 4.6.1 Parts Selection – Array: Add Parts Selection Support Contact details		
Section 4.6.1.2 Added new section for "Parts Selection Support Contact"		
Section 4.6.2 Association – Array: Add in Part Origin - Array and Customer - Array		
Section 4.6.2.1 Part Origin – Array: Add in section for Part Origin association reference		
Section 4.6.2.9 Customer – Array: Add in section for Customer - Array		
Section 4.7 Part Origin – Array: Add in new section for Part Origin details		
Section 4.9.1.1 Document Classification: Added new PartOrigin document classification		
Section 4.9.1.2 EnvironmentalSection: Added ToolNameVersionID from the IEC62474 schema, SVHC, LHPWB and LHPLA		

Standard Improvement Form **JEDEC Standard No. JEP30E**

The purpose of this form is to provide the Technical Committees of JEDEC with input from the industry regarding usage of the subject standard. Individuals or companies are invited to submit comments to JEDEC. All comments will be collected and dispersed to the appropriate committee(s).

If you can provide input, please complete this form and return to:

JEDEC
Attn: Publications Department
3103 North 10th Street
Suite 240 South
Arlington, VA 22201-2107

Email: angies@jedec.org

1. I recommend changes to the following:

☐ Requirement, clause number _____

☐ Test method number _____ Clause number _____

The referenced clause number has proven to be:

☐ Unclear ☐ Too Rigid ☐ In Error

☐ Other _____

2. Recommendations for correction:

3. Other suggestions for document improvement:

Submitted by

Name: _____ Phone: _____

Company: _____ E-mail: _____

Address: _____

City/State/Zip: _____ Date _____


JEDEC®