Recommended Practices
for
STEP AP242 Edition 4
Domain Model XML
Configuration Management

Release 1.3

20 October 2025

Contacts:

Organizational	
Jochen Boy PROSTEP AG mailto:jochen.boy@prostep.com	Frédéric Darré Cimpa S.A.S. frederic.darre.external@airbus.com
Technical	
Guillaume Hirel T-Systems guillaume.hirel@t-systems.com	Kevin Le Tutour AFNeT Services kevin.letutour@afnet-services.fr

Table of Contents

List of Figures

List of Tables

Document History

Revision	Date	Change
1.0	2021-11-26	Initial public release
1.1	2023-04-06	Updates for AP242 Edition 3 and extended PDM-IF Scope
1.2	2023-12-07	Added support multiple versions of same Effectivity/Configuration/ Product Class/Specification Added VersionBranchEffectivity (AP242 Ed.4) Further updates based on current PDM-IF Scope
1.3	2025-10-20	Added Validation Properties Added References to Electrical Wire Harness and Project Schedule Management Systems recommendations Further updates based on current PDM-IF Scope

Working History – Changes since 0.9 (July 2019)

Revision	Date	Author	Change
0.99	2021-11-21	G. Hirel	Initial adaptation for AP242 Edition 3 Domain Model See section 1.1.3 for Implementation changes to AP242 Ed.2 See section 1.1.4 for Implementation changes to AP242 Ed.3 Throughout: Updated references from AP242 Ed.1 TC BO Model to AP242 Ed.3 Domain Model Added ProductClass.VersionId and Effectivity.Version.Id Section 1.1.2: URL of Domain Model corrected; Scope: remark added, that some features have not yet been fully tested by the involved Implementor Forums Section 3.4: typo corrected New Section 4.1.2 Simplified representation for the exchange of one Explicit Configuration; Simplified representation for the exchange of one Explicit Configuration: reworked and example added Section 4.1.4.1: additional recommendation regarding the listed specifications and effectivities Section 4.1.3.3: added ProductConfigurationRelationship Section 4.1.3.4: added ConfiguredAssemblyEffectivity Section 4.2.2.2: added ActualStartDate, PlannedStartDate, Offset in Event Section 4.2.7.2: add precisions regarding the consistence of the effectivities on the levels of the product structure + new Bugzilla issue #9072; ConditionalConfiguration: Postprocessor recommendations added. Section 4.2.8: added EffectivityRelationship Section 4.2.8.4 Effectivity Assignment: Pre- and Postprocessor recommendation updated. Section 4.3.4.2: SpecificationConditionAssignment.ConditionType: ‘validity’replaced by ‘part usage Section 5.1: issue added regarding the relevance of the PDM date/status/version filter settings
1.0	**2021-11-26**	**J. Boy**	**Public Release**
1.0.1	2022-05-09	G. Hirel	Section 4.2.7.2: adding behavior precision on ‘inherited’ effectivities + Todo (null effectivity) Section 4.2.1: mapping added for null effectivity Section 6: update of nesting recommendation; example diagrams for the different nesting use cases

			Annex B: new issue numbers after switch from PDES Inc. Bugzilla to ISO Jira
1.0.2	2023-03-01	G. Hirel	Section 1.1.2: update Ed3 Domain Model URL to official value from the ISO Section 4.1.4.1: add recommendation for idContextRef according to template “Identifier” Section 4.1.7: updated handling of EndItems Section 4.2.8.3: behavior details added when DefiningSpecifications and EffectivityAssignments are applied to a ProductConfiguration + if multiple EffectivityAssigments apply. Section 4.2.8.4: wording improved in the Postprocessor Recommendations for null effectivities Section 4.3.4.1: Figure 41: Instance Model SpecificationConditionAssignment updated Section 4.3.4.3: Definition of attribute Condition.Parameters updatedSection 5.1: new JIRA issue TCSC410303-660 created to support PDM configuration filter parameters Section 6: all figures updated regarding the use of EffectivityContext (optional); update of the figures 43 to 49 and dummy ProductClass if the conditions apply to Effectivities without Context nor making use of Specifications; handling of ExchangeContext.IdentificationContext in all nested files containing the product structure filtered for one ProductConfiguration Annex B: clone issue numbers created in JIRA (out of the migrated Bugzilla issues)
1.1	**2023-04-06**	**J. Boy**	**Public Release**
1.1.01	2023-06-26	G. Hirel	Section 4.2: update of the handling of EffectivityContext Section 4.3.4.1: complex example fixed (removal of the EqualsConditions in case they are not at the top level of the Condition) Section 4.3.5: Use of ConditionParameter for a compact mapping of complex ConditionalEffectivities Section 6: added handling of Specification/Category associated to more than one ProductClass
1.1.02	2023-11-22	G. Hirel	Section 4.1.3: added handling of multiple versions of the same ProductClass Section 4.1.3.1: attribute ‘Scope’ added (Edition 4) Section 4.1.4.1: added handling of multiple versions of the same ProductConfiguration Section 4.2.1: added handling of multiple versions of the same Effectivity Sections 4.2.1 to 4.2.7: update of the handling of EffectivityContext

			NEW Section 4.2.6: VersionBranchEffectivities (Edition 4) Section 4.3.1.1: added handling of multiple versions of the same Specification Section 4.3.5: handling of VersionBranchEffectivities added (Edition 4) Section 6: note added that in case nested files are exported out of a product configuration (only one variant of the product structure), all nested files shall reference the ProductConfiguration as ExchangeContext.IdentificationContext
1.2	**2023-12-08**	**J. Boy**	**Public Release**
1.2.01	2023-12-14 - 2025-07-30	G.Hirel	New section 1.1.5: changes from Ed.3 to Ed.4 Section 4.1.3.1: ProductClass.VersionId mandatory in case of VersionBranchEffectivities Sections 4.1.3.1 (ProductClass) and 4.1.4.1 (ProductConfiguration): added information about the relevant kind of Classification. Section 4.1.4.1: 'deprecated' consolidated regarding the use of DefiningSpecifications and EffectivityAssignment Sections 4.1.4.1 and 4.3.1.1: updated handling of idContextRef for embedded objects New section 4.1.4.2 + section 6: EffectivityControlledProductConfiguration and deprecated use of ProductConfiguration.EffectivityAssignment and .DefiningSpecifications Section 4.2.1: added handling of EffectivityContext for null Effectivity; recommendation added for PDM system that do not support to combine simple effectivities and specifications, or effectivies having a different EffectivityContext within one effectivity Sections 4.2.1 to 4.2.7: InheritanceType/ConfigurationType added as optional attribute to all subtypes of Effectivity, according to Ed4 FDIS Domain Modelhow to generate the UUID v5 values out of the key attribute values Sections 4.2.2, 4.2.4, 4.2.6: recommendation added if the end of the effectivity range is meant 'included' or 'excluded' Section 4.2.6: new optional Boolean attribute 'EndIncluded' in the VersionBranchEffectivity; VersionBranchEffectivity: updated figure; StartVersion/EndVersion replaced by Root/Leaf according to Ed4 FDIS Domain Model Section 4.2.7.2: definition of InheritanceType completed Section 4.3.1.1: updated handling of Identifier and IdentifierString for Specification

			Section 4.3.4: update of the semantic of the XML embedment of SpecificationConditionAssignment in ProductClass Section 6: added ProductConfiguration to the list of possible referenced objects in nested files; added nesting files using EER mechanism, and recommendation Section 7 moved to Annex C New Section 7 for validation properties; replace NumericalValue with IntegerValue + overwork of the validation properties (still draft status) Section 7.1.3: taking transformation of effectivities into account when verifying the validation properties; PropertyValueAssignment added to ProductClass, ProductConfiguration (already recommended by [242-EWH]) and ProductDesignAssociation (missing in Ed4) Add references to PSMS Recommendation [242-PSMS] Add references to Electrical Wire Harness Recommendation [242-EWH]
1.3	**2025-08-29**	**J. Boy**	**Public Release**

1 Introduction

1.1 Document Overview

1.1.1 Goal and Objectives

The goal of this document is to describe the recommended structure and attribute population for particular instance models created from the entities and attributes defined by the STEP AP242 “Managed Model-based 3D Engineering” Domain model and populated according to its XML Schema.

The selected instance models illustrate how to encode configuration management information that needs to be exchanged in support of key industry requirements common across the mechanical design domain. The objectives of the usage guide are to:

- Support the short-term requirements of the Aerospace & Defense and the automotive industries in the realm of configuration management.
- Prevent the emergence of “flavors”, i.e., diverging/conflicting implementations of the AP242 Domain Model XML.
- Ensure consistency with existing Recommended Practices, in particular, the joint CAx-IF / JT-IF Recommended Practices for AP242 Domain Model XML Product and Assembly Structure.

1.1.2 Scope

This document describes the Recommended Practices for the exchange of Configuration Management data. It is based on the third edition of STEP AP242, which was published in 2022. It is based on the fourth edition of STEP AP242, which was published in 2025. It contains the Domain Model (ISO 10303-4442:2025) and the corresponding XML schemas, which can be found at

https://standards.iso.org/iso/ts/10303/-4442/ed-5/tech/xml-schema/domain_model/

Therefore, the XML header of AP242 Domain Model XML files which shall comply to AP242 Ed.4 shall look like the following:

```
xmlns:n0="https://standards.iso.org/iso/ts/10303/-4442/ed-5/tech/xml-schema/domain_model/" xsi:schemaLocation="https://standards.iso.org/iso/ts/10303/-4442/ed-5/tech/xml-schema/domain_model/ https://standards.iso.org/iso/ts/10303/-4442/ed-5/tech/xml-schema/domain_model/DomainModel.xsd"
```

Note: AP242 Edition 4 (ISO 10303-242) contains **Edition 5** of the Domain Model (ISO 10303-4442), hence it is important the string included “ed-5”. Edition 4 of the Domain Model was used in the DIS version of AP242 Ed. 4 and does not support all features described in this document.

This document has not the intention to define all aspects of PLM and configuration management. Focus is to support the implementation of AP242 in software systems for some predefined use cases that are introduced in chapter 1.1. Only those configuration management aspects will be highlighted in chapter 3.3 that are relevant to match the domain and use case specific view to the corresponding technical view for the software developer. Because this document is only a recommendation, some document specific definitions of configuration management terms are used, which reflect industry practice. Also, the PLM system landscape and the lifecycle processes applied by industry may have a more detailed view to the handling of product structures for configuration management. This document, however, raises the claim to give a summary of the most common aspects.

The following AP242 configuration management aspects are in scope of this document, but have not yet been fully tested by the involved Implementor Forums:

- Implicit configuration management
- ConfiguredAssemblyEffectivity
- Bill of Material (BOM)
- Software components are also supported as parts of kind ‘software’
- Configuration calculation results and NC programs, however, only as referenced documents.

The following are out of scope for this document because they are or will be covered in other documents:

- CAx-specific Capabilities such as Kinematics [242-KIN], Composites, Wire Harness (see [242-EWH], assembly level PMIs (see [242-PMI]), etc.
- Basic PDM Capabilities for Product Identification and Assembly Structure [242-PAS]
- Change Management [242-CM]
- External Element References (into Part 21 files or between Domain Model XML files) [242-PAS]
- Implementations based on any edition of the AP242 Domain Model (ISO 10303-3001) other than Edition 1, TC, 3 (2022) and 4 (2025).
- Revision rules for Effectivity, ProductClass, ProductConfiguration and Specification, as these are already covered by [242-PAS]

The following aspects are out of scope for this document:

- QuantifiedOccurrence
- Software code management
- Numerical control (NC) code management
- Configuration management system
- Configuration baseline
- Configuration control, that is, control of changes to configuration items and related documentation
- Configuration status accounting, that is, recording and reporting of information needed to manage configuration items effectively, including the status of proposed and approved changes
- Configuration audit and verification, that is, auditing of configuration items to verify conformance to documented requirements
- Configuration management planning, including the use of standards, guidelines, best practices
- Any other aspects of configuration management that are outside of a PDM/PLM system
- The concept of product specification, that is, the capability to describe products with a large number of variants
- Specified occurrences for configurations, due to lack of use case

The following aspects may – depending on user demand – be included in future releases of this document:

- Feasibility of configurations, that is, distinction that some combinations of configuration options result in valid product variants, others in invalid ones
- Automatic completion of configurations, that is, indications of the dependance of configuration options on the presence of other options, which then may be added automatically
- Product breakdown, that is, distinction of types of breakdowns and breakdown elements into, for example, physical, functional, zonal and system.
- Other types of BoMs than engineering BOM (eBOM) and manufacturing BOM (mBOM).

1.1.3 Implementation Changes between AP242 Ed.1 TC and AP242 Ed.2

For a number of constructs, the implementation changed between the Ed.1 and Ed.2, i.e. ISO 10303-3001 Ed.2 and ISO 10303-4442. These changes became necessary due to deficiencies found in the original definitions of these elements, or to lay the foundation for future extensions of the data model.

The changes include structural changes such as embedding and referencing of XML elements as well as technical changes such as new element types or changed definition of attributes. Except very few of them, all these changes are upward compatible.

The list below provides an overview for the necessary implementation changes to support AP242 Ed.2 in scope of this document:

- SerialEffectivity attributes startNumber/endNumber renamed to StartId/EndId (#6648) (section 4.2.4)
- DataValidityEffectivity replaced by DatedEffectivity; StartDefinition/EndDefinition are moved from Effectivity to DatedEffectivity (section 4.2.2)
- New attribute LotId in LotEffectivity (section 4.2.3)
- Type of PartView/ProductConfiguration.Occurrence changed from DefinitionBasedOccurrence (Single and QuantifiedOccurrence) to Occurrence (embraces also SpecifiedOccurrence) (see section 4.1.4.1) (rollbacked in Ed.3)
- Embedment of Condition along Condition.Parameters

  Allows much more compact mapping of logical expressions (section 4.3.4.3)
- ProductDesignAssociation now embedded into ProductConfiguration

  Attribute AssociatedConfiguration is removed (section 4.1.4.2)
- ProductConfiguration now embedded into ProductConcept (ProductClass)

  Attribute MemberOf is removed (section 4.1.4.1)
- Specification now embedded into SpecificationCategory

  Attribute Category is removed (section 4.3.1)
- ConfiguredAssemblyEffectivity now embedded into ProductDesignAssociation

  Attribute ResolvedConfiguration is removed (section 4.1.4.5)

1.1.4 Implementation Changes between AP242 Ed.2 and AP242 Ed.3

- Rollback of the Ed2 change “Type of PartView/ProductConfiguration.Occurrence changed from DefinitionBasedOccurrence (Single and QuantifiedOccurrence) to Occurrence (embraces also SpecifiedOccurrence)” (see section 4.1.4.1)

1.1.5 Implementation Changes between AP242 Ed.3 and AP242 Ed.4

- New object VersionBranchEffectivity (see section 4.2.6)
- New object EffectivityControlledProductConfiguration (see section 4.1.4.2). The previous mapping workaround using EffectivityAssignment is deprecated
- Attribute ‘Scope’ added to ProductClass (see section 4.1.3.1)
- Attributes ConfigurationType and InheritanceType now optional for all subtypes of Effectivity rather than only in ConditionalConfiguration (see sections 4.2.1 to 4.2.7.2)

1.1.6 Intended Audience

This document is intended to be an implementation guide for developers of PDM and file translation application systems that support and exchange configuration management information with other systems and applications, in support of the design engineering and related downstream business processes.

1.1.7 Intended Use

This document is intended to be a manual and companion to the developer of STEP data exchange and translator software used by applications that support configuration management definitions. It provides guidelines for the consistent preprocessor instance model creation and requirement value encoding to enable meaningful information exchange between different systems and applications using the STEP AP242 Domain model, and guidelines for the consistent interpretation by a postprocessor of the STEP AP242 Domain model exchange file.

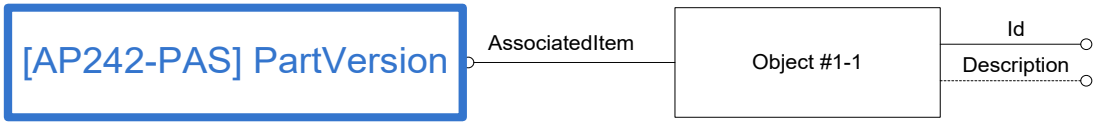
1.1.8 Document Style

The overall document proceeds in an incremental, step-by-step fashion to describe, and in parallel to illustrate the recommended instantiations of the XML elements in the STEP AP242 Domain model.

The “template” concept is used in this document. Structures and substructures are defined in one section and then reused in other sections of the documents. These templates are represented by the blue boxes in the diagrams.

Templates for common core constructs, such as product, are defined in the “Recommended Practices for AP242 Domain Model XML Product and Assembly Structure” and reused here. References to such templates are prefixed with [242-PAS].

The instantiation diagrams are presented using a graphical notation intended to illustrate the instance model. Detailed notation is described in chapter 1.1.10.



Following each instance diagram, a table lists all the attributes of each displayed entity according to the XML schema specification of ISO 10303-4442. The table includes not only the attributes of the EXPRESS schema of the AP242 Domain Model, but also inverse attributes of all possible relations to the element in question. Attributes that are considered important for the scope of these Recommended Practices are in these tables written in black. Attributes that are written in

grey are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

ENTITY <Object>	**Attribute Type**
Id	Id
Description	OPTIONAL DescriptionSelect
AssociatedItem	PartVersion

Below the table, all recommended attributes (written in black) are listed and recommendations are made for them.

Attribute recommendations:

- **Id:** The id attribute specifies the identifier of the <Object> entity.
- **associatedItem**: the PartVersion that identifies a part which is a valid implementation for the <Object>

Finally, a STEP AP242 Domain model XML exchange structure example is included. The example exchange file corresponds directly to the instance model diagram and illustrates the very same thing using a different notation, i.e., STEP AP242 Domain model XML syntax versus the graphical instance model notation.

```
<Object uid="obj---01">
      <Id id="Manufactural Product"/>
      <AssociatedItem uidRef="pv---02"/>
</Object>
```

1.1.9 Document Structure

The overall scope of requirements is partitioned into a set of major sections corresponding to the identified units of functionality. Within a major section, there may be sub-sections. These sub-sections further divide the scope into smaller components of coherent functionality (called “feature”) that interact with each other to realize the functionality of the entire unit.

There is generally a description of requirements and a corresponding instance diagram associated with each section and sub-section of this document. Each instance diagram is followed by a detailed explanation and specific recommendations for the entities used in the instantiation diagram example. The entity listing and explanation is in turn followed by the corresponding XML exchange structure example.

Within a section, diagrams corresponding to sub-sections incrementally build upon one another to finally achieve a complete instance model example that illustrates the entire scope of the unit of functionality.

1.1.10 Instantiation Diagrams

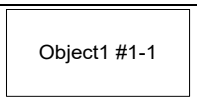
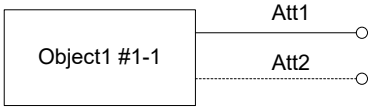
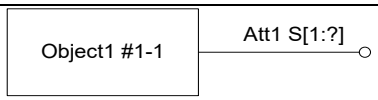
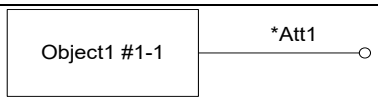
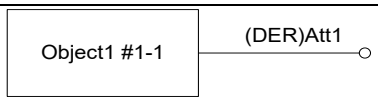
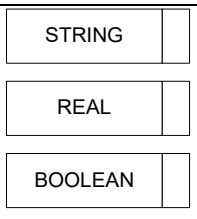
The diagrams are presented using a graphical notation intended to illustrate the instance model.

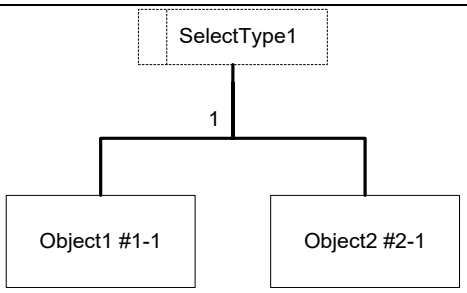
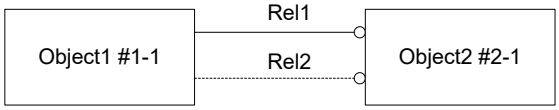
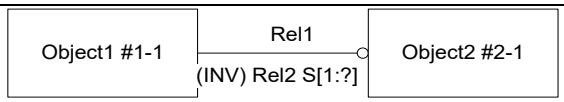
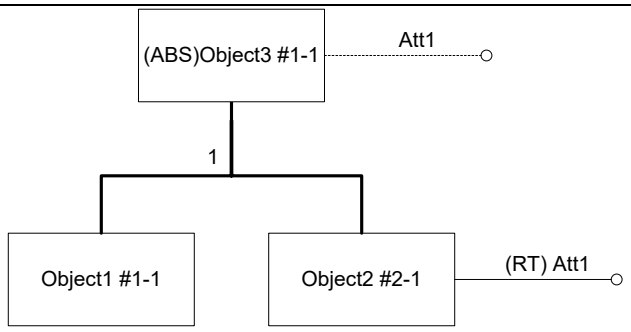
This notation is not EXPRESS-G and does not illustrate the XML schema; rather it is a graphical illustration of a specific population of a particular instance model of the schema. This notation supports:

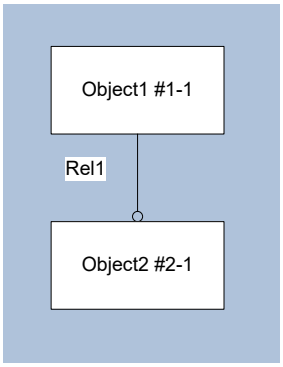
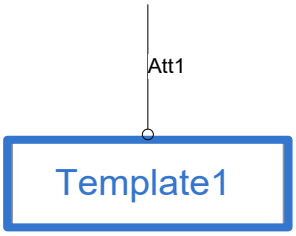
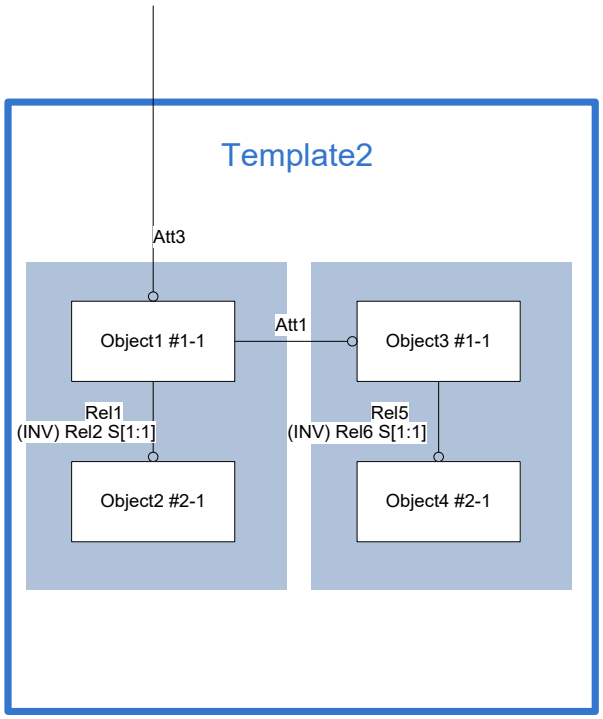
- Illustration of entity instances and attribute values (both mapped as XML elements)
- Illustration and identification of referenced entity instances that are either fully illustrated in the current figure, or that refer to another template (if not fully illustrated in the current figure)

- Indication of optional attributes and optional reference entity instances (dashed lines),
- Illustration and identification of groups of functionally related instances (shaded bounding box), showing how XML elements are embedded into each other (the XML elements representing the entity instances placed below are embedded into the XML element representing the entity instance placed above), and
- Identification of specific attribute values (typically string values, may also be enumerated type values or numerical values).

A legend for the diagram notation is shown below:

	Object (instance of an EXPRESS ENTITY) After the #, an instance number is given
	Att1: mandatory attribute Att2: optional attribute
	Aggregate type for the definition of the cardinality constraint: **B**: Bag (non-ordered and may contain duplicates) **S**: Set (non-ordered and may not contain duplicates) **L**: List (ordered) [x : y]: lower size : upper size ?: unconstrained **A**: Array (indexed) [x : y]: lower index : upper index
	Additional constraint on the object: the attribute(s) depicted with '*' have to contain unique values. *Currently not used in this document*
	Derived Information from another object or attribute *Currently not used in this document*
	Simple data types

	User-defined data type *Currently not used in this document*
	Enumeration Type (consists of a limited list of possible values defined for this type)
	Select Type (is used if a relationship from an object may apply either to Object1 or Object2) For a better readability, the members of a select type are displayed using the inheritance link (see below) and the mutually exclusive constraint. This has the same semantic as a select type. *Currently not used in this document*
	Attribute as relationship between two Objects (mandatory or optional), The circle at the end of the line gives the direction. Rel1: mandatory relationship Rel2: optional relationship
	Rel2: inverse attribute (i.e. in the converse direction to Rel1) with cardinality constraint *Currently not used in this document*
	Inheritance from a Supertype to its Subtypes An Abstract Supertype (ABS) cannot be instanced without one of its non-abstract Subtypes 1: Only one subtype can be instantiated at a time (the subtypes are mutually exclusive). Per default, all the subtypes may be combined (not supported by XML) RT: an inherited attribute is retyped, for example through restriction of its type, length, value range, cardinality, mandatory instead of optional or DERIVED

	Currently not used in this document
	Objects shown under each other within a blue colored square are embedded into each other in XML: here Object2 #2-1 is embedded into Object1 #1-1 as its XML element Rel1
	The templates defined in this document are reused in other sections. This is the simple way to refer to a template (if the object referenced within the template is implicit, for example the object 'Classification' for the template 'Classification'
	If the template is more complex and the object referenced within the template is shown explicitly, portions of the reused template are displayed within a blue frame.

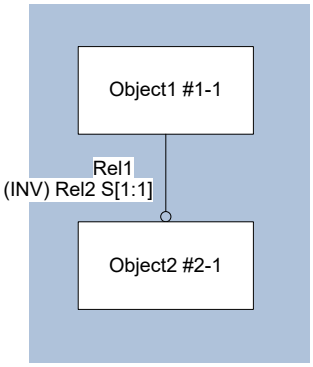
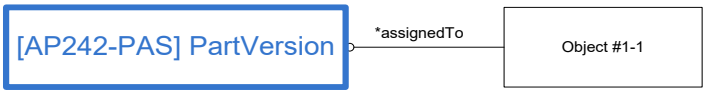
	Alternatively, a template may be reused through adding new XML containments to it
	If the remplate is defined in an other document, it will be marked with a prefix: • [242-PAS] for "Recommended Practices for AP242 Domain Model XML Product and Assembly Structure",

Table 1: Instance Diagram Notation

1.2 Organizational Framework

These "Recommended Practices for AP242 Domain Model XML Configuration Management" is developed and supported by a number of "communities", specifically the vendor and user communities devoted to the development and implementation of AP242 and its associated Domain Model. This section describes those communities' responsibilities.

1.2.1 Vendor Communities

The **PDM-IF Implementor Group (IG)** will be responsible for the overall organization and development of this document. The PDM-IF IG will:

- Coordinate the creation of the document
- Verify the approach of the recommended practices in PDM-IF Test Rounds, and publish result summaries of testing "AP242 Domain Model Configuration Management"
- Ensure the consistency with other "AP242 Domain Model XML Recommended Practices"

1.2.2 User Communities

The **PDM-IF User Group (UG)** is a forum of PDM experts from the Aerospace and Defense as well as the automotive industries. The PDM-IF UG is responsible for development of the document and will:

- Support the development of the document
- Provide subject matter experts
- Provide industry requirements and ensure they are fulfilled
- Ensure the consistency with PDM standards spanning the complete product life cycle

1.3 Maintenance of this Document

This document describes the recommended practices to implement configuration management data using the AP242 Domain Model. It is based on the joint Cax-IF / JT-IF "Recommended Practices for AP242 Domain Model XML Product and Assembly Structure". Since configuration management is an extension to the core scope that is specific to the PDM domain, it is documented in a separate document with references to the core document where needed.

It is the responsibility of the PDM-IF not only to maintain this document, but also to ensure its consistency with the core document.

AFNeT, PDES, Inc., prostep ivip Association and VDA as the hosting organizations of the related implementor forums will maintain and extend the document as long as it provides utility to the vendor community.

2 Reference to Recommended Practices

2.1 Reference to Core Document [242-PAS]

This document is an extension of the joint CAx-IF / JT-IF Recommended Practices for AP242 Domain Model Product and Assembly Structure. Definitions of templates and XML elements contained in the core document are not repeated here.

Where necessary, references to sections of the core document are given in this format:

[242-PAS] 4.6.7

This means a reference to the Recommended Practices for AP242 Domain Model Product and Assembly Structure, section 4.6.7.

This Recommended Practices for AP242 Domain Model XML Configuration Management are built on the following version of the core document:

Version 4.0; dated 2025-10-20

This document is available on the MBx-IF web site.

2.2 Listing of Recommended Practices Version in Exchange Files

For validation purposes, STEP processors shall state which Recommended Practice document and version thereof have been used in the creation of the STEP file. This will not only indicate what information a consumer can expect to find in the file, but even more importantly where to find it in the file.

This shall be done by adding a pre-defined string to the second string element of the `Documentation` attribute of the `Header` element in the XML file (for details see section [242-PAS] 4.1.5), while the first value shall be the value defined in section 3 of [242-PAS]. The value follows a specific pattern well established in Part 21 files:

`Document Type---Document Name---Document Version---Publication Date`

The string corresponding to this version of this document is:

```
<Documentation>PDM-IF Rec.Pracs.---AP242 Domain Model XML
Configuration Management---1.3---2025-10-20</Documentation>
```

General Postprocessor Recommendation:

If a postprocessor encounters attribute values, or object instantiations different from the ones recommended in this version of the document, but is needed for the use case, an additional exchange agreement is supposed to be in place among the parties involved in the data exchange.

3 Fundamental Concepts and Assumptions

This chapter discusses fundamental concepts and assumptions for Configuration Management. Focus is on items that are in scope of this document (see chapter 1.1.2), but also some related aspects are elaborated upon to clarify the context of in-scope items.

3.1 The Big Picture

ISO 10303 (STEP) is an International Standard for the computer-interpretable representation of product information and for the exchange of product data. Its objective is to provide a neutral mechanism capable of describing products throughout their life cycle. The document in hand is a guide to some aspects of only ISO 10303-242: Managed model-based 3D engineering, also called AP242.

AP242 supports the definition of product concepts, which define products from market or customer-oriented viewpoints. As they are offered to the customers, these product concepts often define conceptual product models that are available for or delivered to the customer in different configurations or variants.

AP242 is suitable for the following use cases; see chapter 3.2 for details):

- Neutral file exchange
- Archiving and retrieval of product data
- Implementation and sharing of product databases within PLM processes.

AP242 has not the aspiration to support all aspects of Configuration Management and is not designed to be the internal data model of a Configuration Management System.

The scope of AP242 in the context of PLM in general is shown in Figure 1: AP242 Big Picture. The red boundaries and red text indicate the capabilities of AP242. The white circles represent sub-domains within general PLM.

The focus of this document is the subdiscipline of ***configuration identification*** of the ***physical features*** for typical or serialized Configuration Items (CIs) (see chapter 3.3.5).

In scope is product structure with ***explicit or implicit configuration***. Configuration is restricted to product structure and related documents with CAD models, drawings, and additional descriptions. Software components and NC-programs are only in scope in this release of the document if they are managed as parts; management of software or NC code is out of scope.

Product breakdown is out of scope of this release of the document.

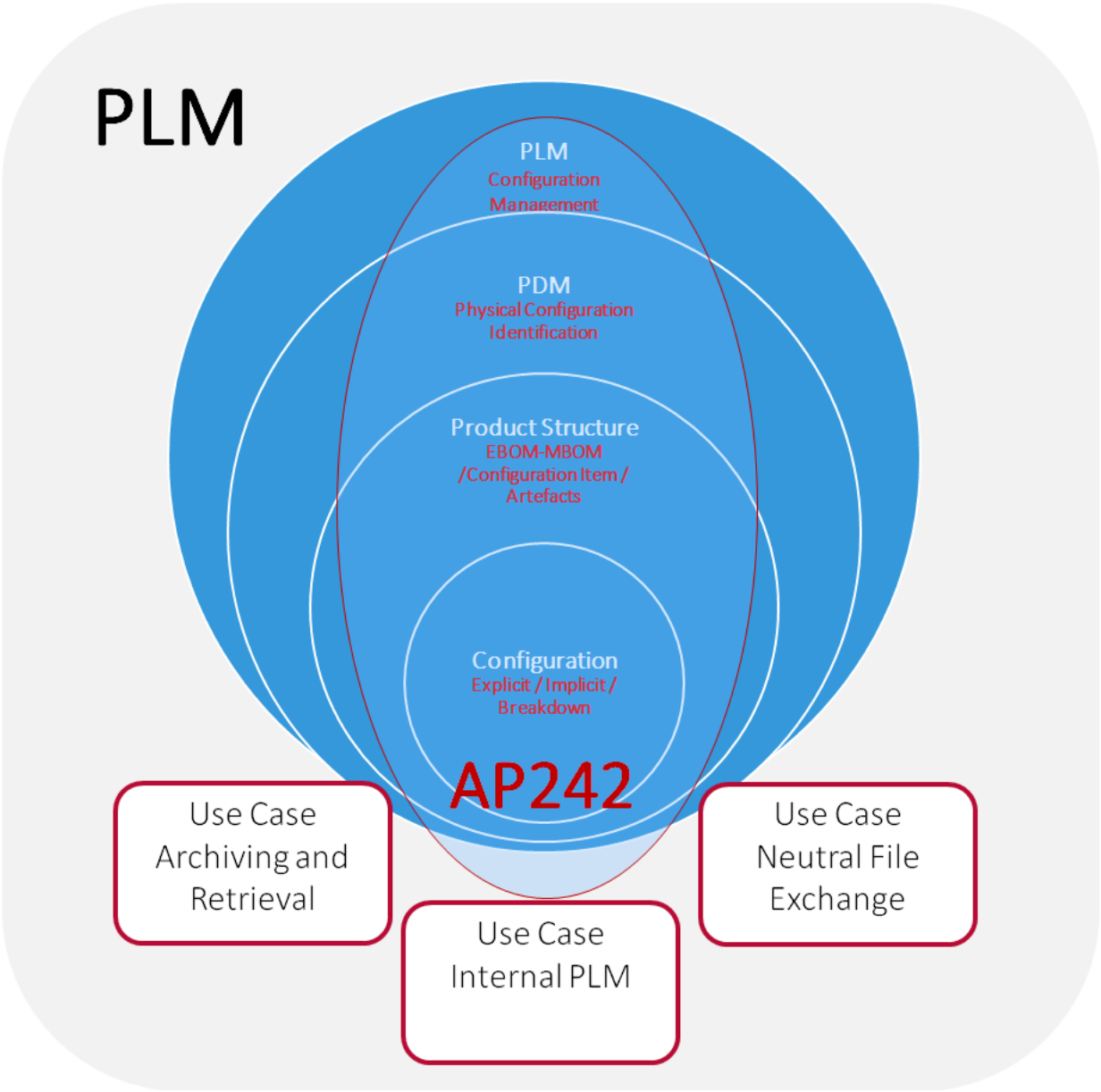


Figure 1: AP242 Big Picture

3.2 Use Cases

With respect to Configuration Management, AP242 supports the following three main use cases, which are in explained in the sub-chapters, below:

- Configuration Management for supplier exchange
- Configuration Management for archiving and retrieval
- Configuration Management for internal PLM.

3.2.1 Configuration management with AP242 for supplier exchange

Main characteristics of supplier data exchange are:

- different organizations
- different systems
- asynchronous data exchange.

In this use case, the focus for AP242 is to support explicit configurations. If partners support the same configuration management methodology, also implicit configuration could be used.

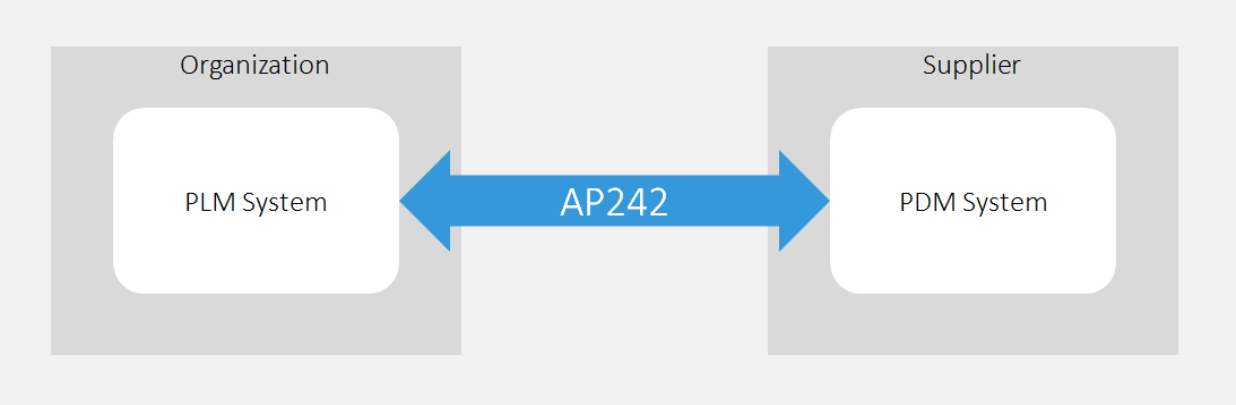


Figure 2: Configuration Management for supplier exchange

3.2.2 Configuration management with AP242 for archiving

Main characteristics for long-term archiving and retrieval are:

- same organization
- different systems (along the timeline)
- different people (all “tacet” knowledge is gone; the archival package needs to be totally self-contained).

In this use case, the focus for AP242 is to support implicit and explicit configurations with additional organization-specific information. Organization-specific information could be, for example:

- details of the status network
- information about users, groups, roles
- details of the file repository.

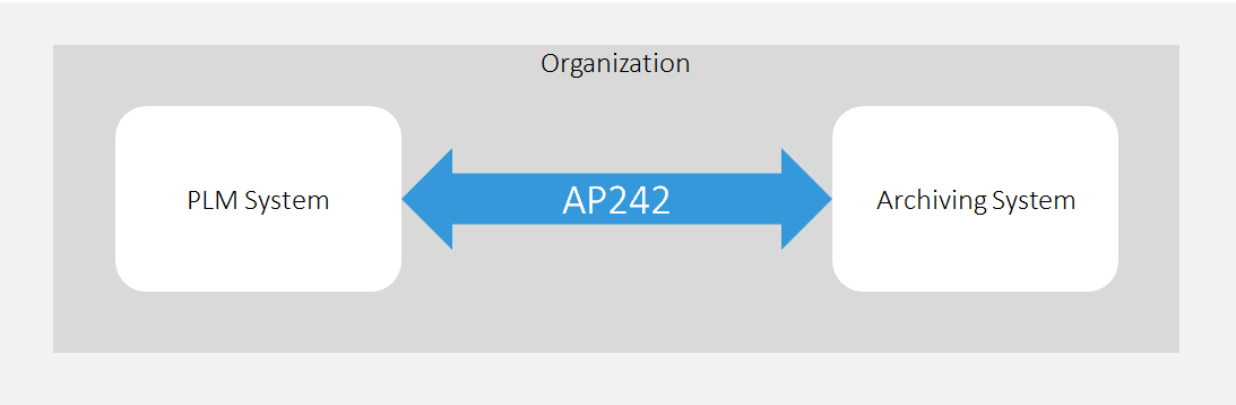


Figure 3: Configuration Management for archiving

3.2.3 Configuration management with AP242 for internal PLM

Main characteristics of internal PLM for synchronous integration of several internal systems are:

- same organization
- different systems
- synchronous integration.

In this use case, the focus for AP242 is to support the daily business with all needed information about specifications, conditions, and breakdowns. Internal exchange is needed, if different systems are in use to support the product lifecycle and to synchronize information between ERP, PLM and Production Planning and Control System (PPS). The need to exchange data is often linked to lifecycle transitions.

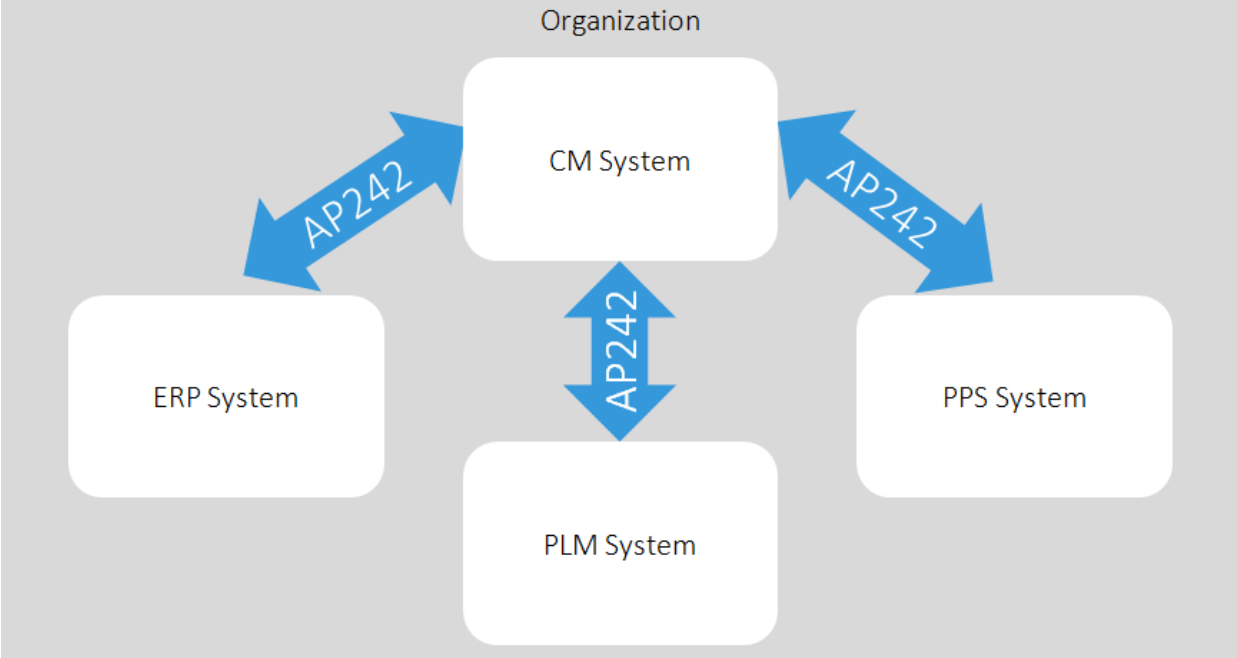


Figure 4: Configuration Management for internal PLM

3.3 Configuration Management

This chapter gives a short introduction to configuration management in general and highlights relevant aspects in AP242.

3.3.1 Definition

Configuration management is a systems engineering process for establishing and maintaining consistency of a product's functional and physical features with its requirements, design, and operational information throughout its life.

Configuration management is a management discipline and is closely linked with:

- Product Structure Management
- Requirements Management
- Change Management
- Release Management
- Supply Chain Management
- Archiving and retrieval.

Configuration Management has five main functions of which only the first one is in scope of this document (see chapter 1.1.2):

- **Configuration Identification**
  Identify and document the functional and physical characteristics of the collection of Configuration Items.
- **Configuration Control**
  Control changes to Configuration Items and related documentation.
- **Configuration Status Accounting**
  Record and report information needed to manage Configuration Items effectively, including the status of proposed and approved changes.
- **Configuration Audit and Verification**
  Audit Configuration Items to verify conformance to documented requirements.
- **Configuration Management Planning**
  Application of standards, guidelines, best practices

The focus of AP242 is physical and functional configuration identification.

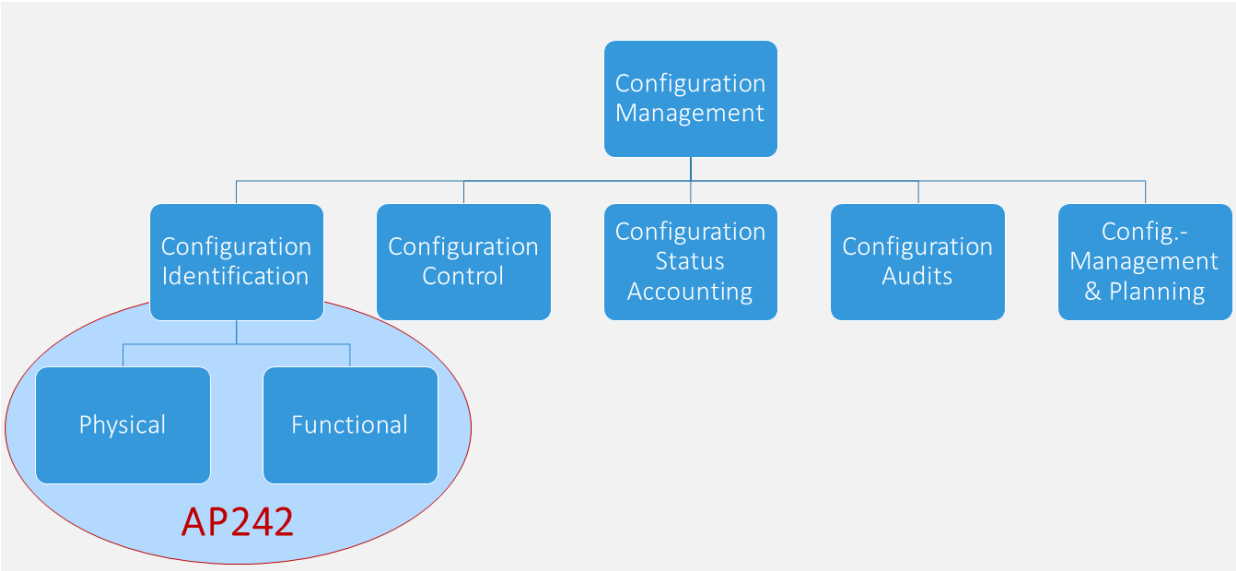


Figure 5: Configuration Management functions

3.3.2 Configuration Identification

Configuration Identification describes the identification and documentation of functional and physical features of the product, such as:

- Identification and placement of Configuration Items in the product structure
- Documentation of physical and functional features
- Methodology of numbering with unique identifiers and classifications
- Definition of baselines.

3.3.3 Configuration

A configuration is a description of a product at a particular time or in a defined delivery status. The description contains all documentation relevant to manufacturing, assembly, quality control and maintenance.

Documentation may include:

- Product Structure
- Documents
    - CAD Models
    - Viewing data
    - Drawings
    - NC programs
    - Textual descriptions
    - Calculation results
- Software Components

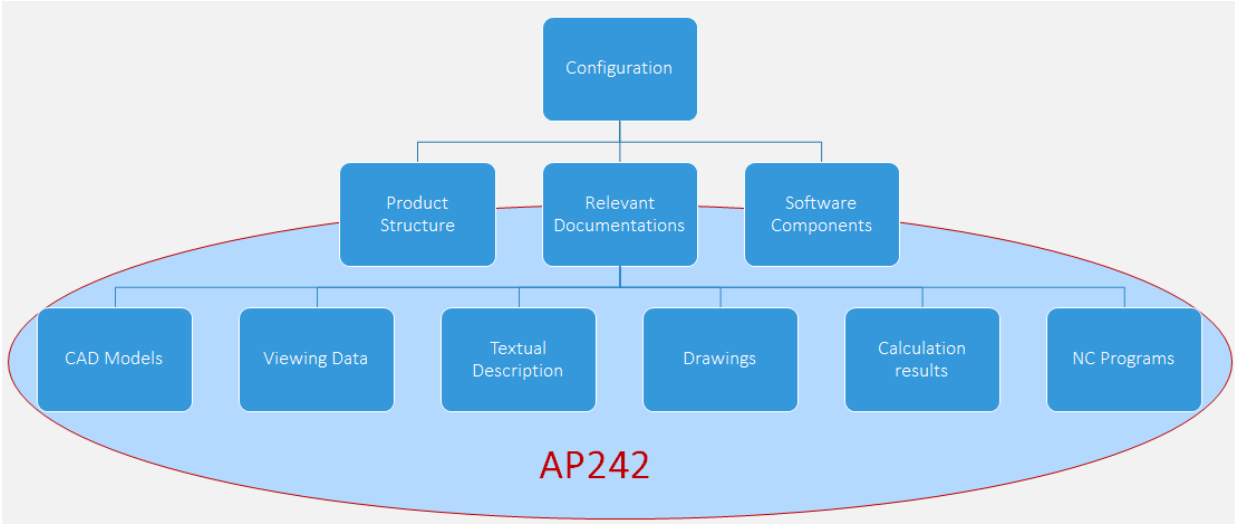


Figure 6: Configuration

Configuration calculation results and NC programs are supported as referenced documents only. Software components are supported as parts of kind ‘software’.

Configurations may be defined explicitly or implicitly, that is, by conditions.

3.3.3.1 Explicit Configuration

Configuration mechanisms are used differently across various industry branches. In Aerospace, explicit representation of configurations of product concept is suitable for the management of design as-planned manufacturing configurations, the traditional BOM inputs to a manufacturing resource planning. These explicit configurations may also be suitable for management of other activities “downstream” from the design phase, such as as-built and as-maintained configurations.

From an Automotive point of view, the manufacturable object is triggered by Date Effectivities, but not for explicit configuration. Instead, the parts, the part occurrences and the conditions of the product specification have additional date effectivities (i.e., they apply from datetime xxx to

datetime yyy). The explicit configuration is computed out of them all (for a given customer order at a given date).

The Explicit Configuration is mostly a 100% product structure (> 100% if for example serial/date ranges are used to define a ProductConfiguration). The content differs, dependent on the use case:

Use Case	Content
Data exchange	No effectivity information
Archiving	No effectivity information
Internal PLM	With effectivity information

Table 2: Configuration use cases

Configuration effectivity in AP242 allows attachment of effectivity information to part versions or to occurrences of component parts in the context of a particular Configuration Item. This enables specification of the valid use of a part version or a part occurrence in the context of a lot, serial number range or time period of a particular product configuration. These kinds of effectivities can also be used in combination with conditional effectivities. This controls the constituent parts that are planned to be used for manufacturing end items of a particular product configuration within a dated time period or for a certain lot, a serial number range of the end items and/or certain conditionalEffectivities.

3.3.3.2 Implicit Configuration

It is recommended to use implicit configuration if the number of possible variations of a product concept is too large. The members of a product concept might therefore be defined implicitly by identifying the product features that characterize it or are available for it as options (so-called 150% structure). Conditions among those features may be specified to manage dependencies and define the valid product variations for a particular concept. This implicit definition of the configurations available for a certain product concept is suitable for customer option and variant specification: once all options are specified a single configuration is determined that may be represented explicitly for downstream applications. The implicit configuration is often driven by the ERP system and applied to the product structure in PLM systems for DMU purpose.

The usage of implicit configuration requires the support of the same specifications for a product at both exchange partners PLM/BoM system, or at least that the specifications are mappable to each other. If needed, the Dictionaries and SpecificationCategories also have to be mappable.

The implicit configuration will be defined in following definitions:

- Effectivities (Serial, Date, Conditional, …)
- Variants (ProductConfigurations)
- Options (Specifications, SpecificationCategories)
- Conditions (AND, OR, NOT, …)
- Dictionaries (Option Pool).

3.3.3.3 Feasibility

The use of feasibility is an example of how implicit configurations are defined by options and conditions.

The feasibility of configurations is out of scope of this document but included here for clarification.

The following table shows the possible variants that could be delivered to the customer and their corresponding options. Respecting given conditions, not all combinations of the options may result in valid product variants. The conditions themselves are not part of this matrix and could be defined as follows:

- If B then not C
- If A then not D

		Variants						
		1	2	3	4	5	6	7
Pr	A	X	X			X	X	
	B	X	X				X	
	C			X	X	X		X
	D			X	X			X
	E					X	X	X
	F		X		X		X	X
	G	X	X	X	X			

Figure 7: Configuration Table Example

Configuration tables are a simple way of handling feasibility rules. More complex tables may be rule based.

The list of variants may be generated out of a configuration table, but sometimes there are so many variants that they are only defined as needed and their feasibility shall be checked.

3.3.3.4 Automatic completion

Automatic completion is another example of how implicit configurations are defined by options and conditions.

The automatic completion of configurations is out of scope of this document but included here for clarification.

Depending on the presence of certain options, some further options may have to be added automatically, for example the option ‘larger battery’ if many electrical devices are ordered by the customer.

3.3.4 Configuration Baseline

A configuration baseline is the description of a frozen product configuration at a defined date and is used for company internal or external purposes.

This document focuses on the release processes and milestone reviews within the engineering phase and the manufacturing phase, as well as the transition from engineering to manufacturing planning. AP242 in general will support the product lifecycle phases from as-planned to as-built. The phases for as-deployed and as-maintained are supported by AP232 and AP239.

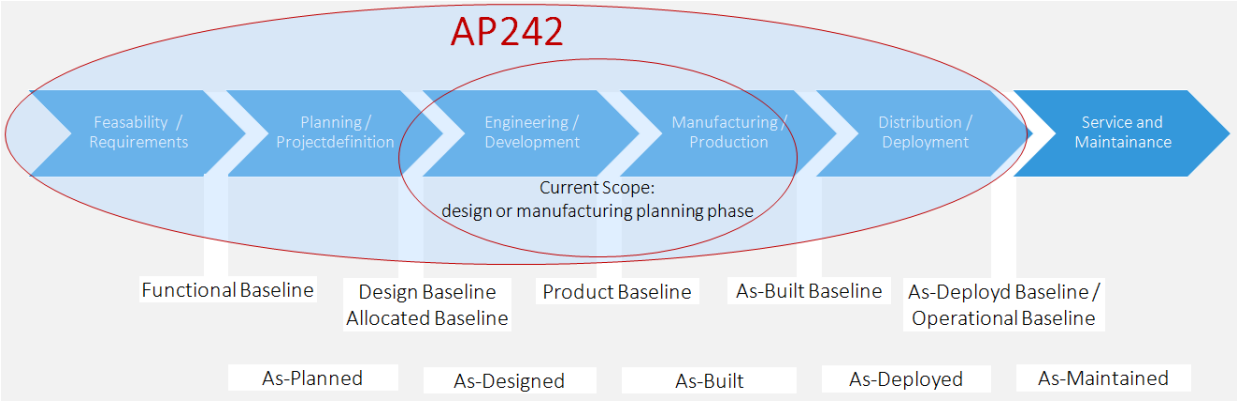


Figure 8: Configuration Baselines

3.3.5 Configuration Item (CI)

A Configuration Item is an item that is identified and controlled over its entire life cycle. It can be a single part, an assembly or a software program or any combinations of those.

The following three different kinds of Cis may be distinguished for engineering and manufacturing:

- Technical Configuration Item (TCI)
- Contractual Configuration Item (CCI)
- Serialized Configuration Item (SCI).

3.3.6 Product Concept

Product concept is the understanding of the applicability of a product in order to showcase its best qualities and maximum features. Marketers spend a lot of time and research in order to target their intended audience. Marketers will look into a product concept before marketing a product towards their customers.

While the “product concept” is based upon the idea of customers who prefer products with the most quality, performance, and features, some customers prefer products that are simple and easy to use. Figure 12 shows the relationship between product concept and configuration.

3.3.7 Bill of Material (BOM)

A Bill of Material (BOM) is an annotated list of constituents of a product at a specific point in time of its life cycle.

The following table gives a rough classification of the different ways of managing BOMs and their association to possible configuration approaches. This classification is only relevant for the use case “internal PLM”. The association to other use cases is industry dependent and, therefore, not included.

Bill of material	Coverage	Configuration management / Engineering
I designed (concrete) BOM	100% BOM	• Order Engineering • Explicit Configuration
Order neutral, configured BOM	150% BOM	• Driven by Change and Release Management • Few Order Engineering
Rule based or feature based variant BOM	150% BOM	• Implicit Configuration • Driven by CPQ (Configure Price Quote) out of ERP • No Order Engineering

Table 3: BOM Classification for Configuration Management for internal PLM

There may be many types of BOMs. eBOM and mBOM are explicitly in scope of this document.

3.3.7.1 eBOM

The engineering bill of material or eBOM is a configuration of the product to show how it is designed. It is used in the product lifecycle phase of design.

3.3.7.2 mBOM

The manufacturing bill of material or mBOM is a configuration of the product to show how it will be assembled. It contains all the parts and assemblies required to build a complete and shippable product. It is used in the product lifecycle phase of manufacturing. The mBOM is normally derived from the eBOM but refactored for the requirements of manufacturing. The eBOM is usually changed as follows:

- The mBOM includes additional information, such as, machines and tools that are necessary to build the product, assembly process information and packaging material.
- The mBOM could also prune the eBOM from all component parts that are assembled by suppliers.
- The mBOM may have a completely different product structure than the eBOM, since it shall reflect the assembly process and not the top-down physical structure of the eBOM.

3.3.8 Configuration Management system

A Configuration Management system consists of a set of tools to identify the information belonging to a product configuration during the entire product life cycle. Data concerning Configuration Management systems is out of scope of this document.

3.3.9 Effectivity

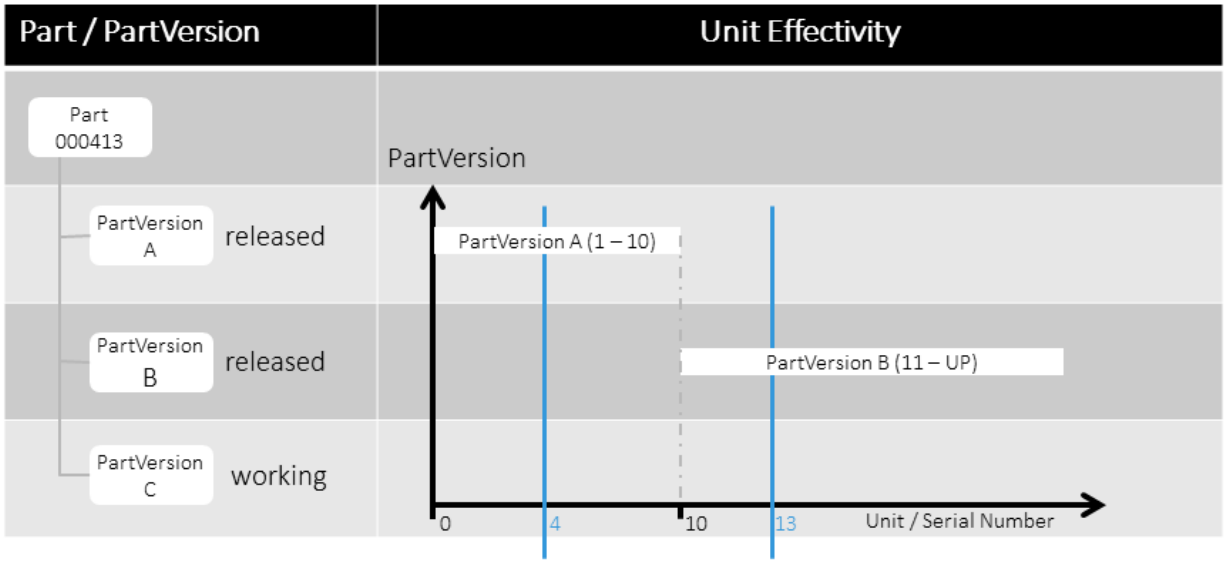
Effectivity describes the validity of data in a certain context. The validity of data can be expressed by conditions that specify, for example, time ranges for which data are applicable. Validity may also be expressed for individual parts, which are identified by serial or unit numbers.

3.3.9.1 Unit Effectivities

A unit (or serial) effectivity could be defined for a single lot or a single serial number as well as for a range or a list of them, or a combination of single value, range and list.

The following scenarios are supported:

- For a given lot or a given serial number
- From 0 to a given serial number
- From a given serial number to a higher serial number
- From a given serial number to undefined.



Unit	Valid PartVersion
4	PartVersion A
13	PartVersion B

Figure 9: Unit Effectivity Example

3.3.9.2 Date Effectivities

A date effectivity is mainly dependent on the initial creation, change, or release management and will mostly be defined by an ERP system. The date effectivity is also used in the PLM System in eBOMs attached to engineering change orders and in mBOMs attached to the manufacturing change orders.

The following three scenarios are supported:

- From undefined to date
- From begin date to end date
- From begin date to undefined.

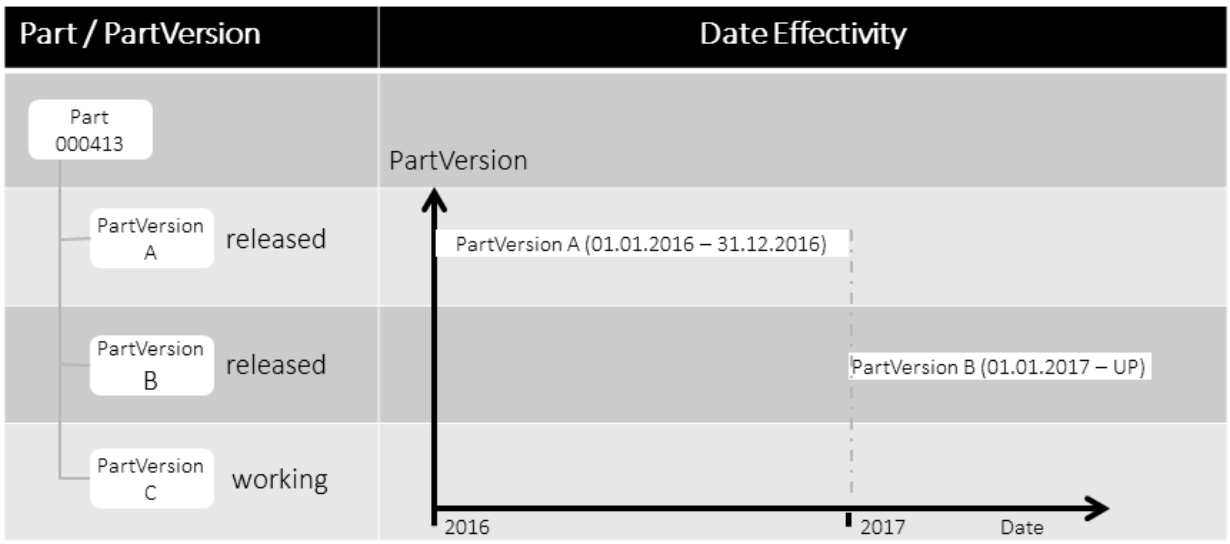


Figure 10: Date Effectivity Example

3.3.9.3 Conditional Effectivity

Conditional effectivity is explained in chapter 3.3.3.2, Implicit Configuration. Like the other kinds of effectivities, conditional effectivity data are exchanged 'for information only'.

3.4 Definitions in ISO 10303-4442

These recommended practices are based on the following document:

- TECHNICAL SPECIFICATION ISO/TS 10303-4442:2025
  Industrial automation systems and integration — Product data representation and exchange
  Part 4442: Domain model: Managed model based 3d engineering

This standard is a part of AP242 and often referred to as the AP242 Domain model.

The figure below provides an overview of the AP242 Domain information model capabilities and their logical groups.

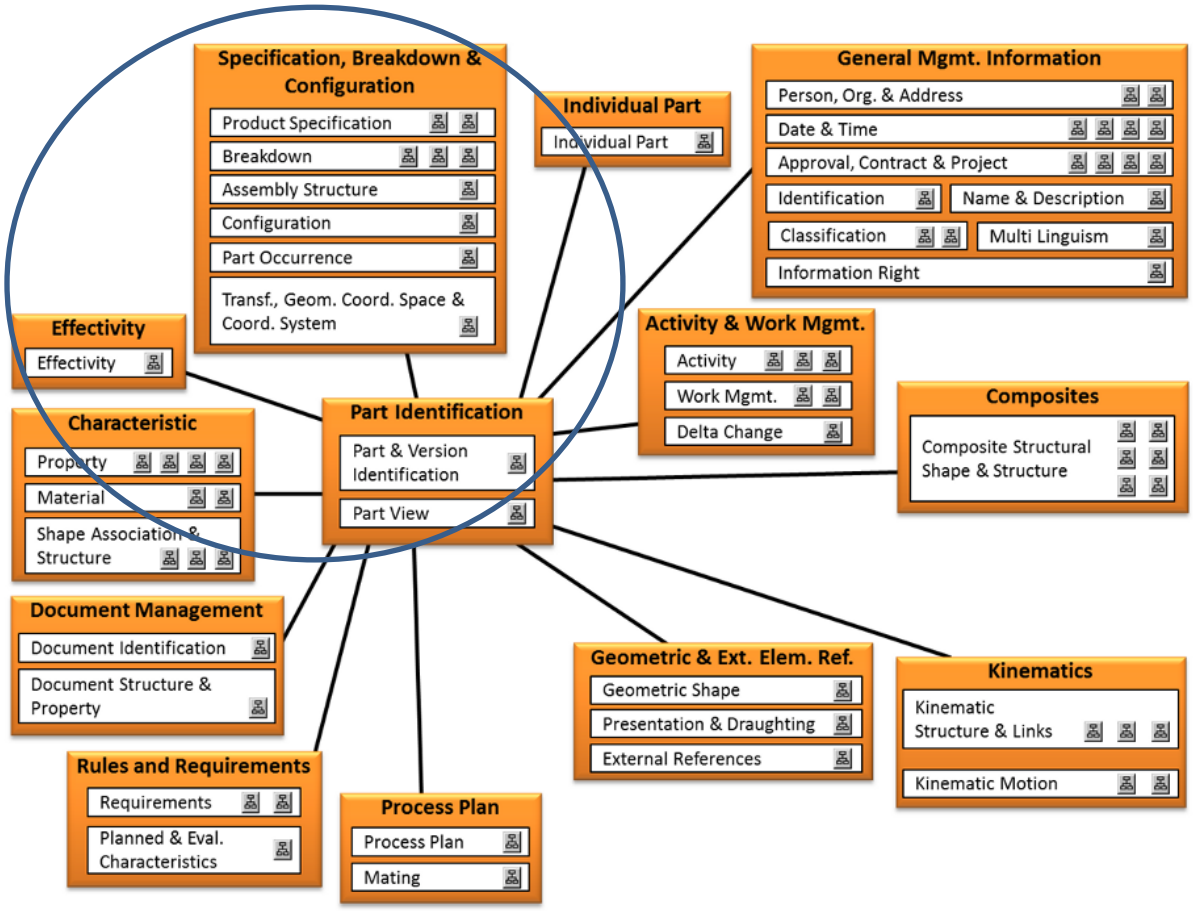


Figure 11: Information requirement domains in ISO 10303-4442

3.4.1 Configuration

AP242 supports configuration management by the four main concepts that are listed on the right-hand side of Figure 12 below. They are renamed and put in relation to one another on the left-hand side.

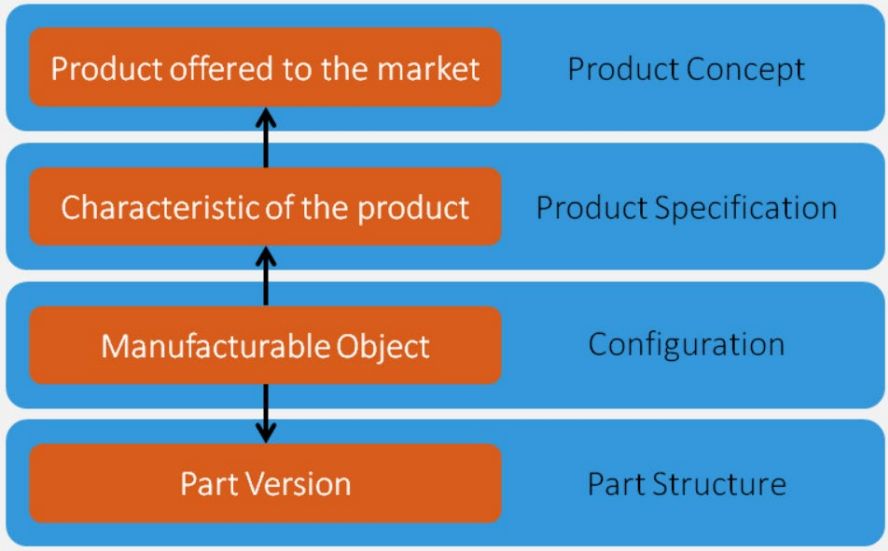


Figure 12: Simplified overview of the AP242 configuration and product specification Domain Model

The figure is a simplified view of the descriptions in chapter 4.2.7. of the STEP AP242 Domain Model.

The configuration capability of the AP242 Domain model provides mechanisms to identify a single manufacturable product out of a large number of variants and versions. The main concepts that are used to handle this identification of a single manufacturable product are the following:

- Product configurations are used to identify a manufacturable product out of a possibly large number of variants of a product class by its associated specifications without having an explicit representation of the product, e.g., a customer order for a car.
- Product design association (ProductPartEquivalence) is used to associate an implicit representation of a manufacturable product (ProductConfiguration) with its explicit representation (PartVersion).
- Manufacturing configuration is used to specify the validity of occurrences of parts (PartInstance) with regard to manufacturing aspects in the context of an implicit as well as an explicit representation of a manufacturable product (ProductConfiguration, PartVersion).
- Configurations are used to configure part occurrences, alternative solutions, breakdown elements, and process plans in the context of the conditions (ClassConditionAssociation, ClassSpecificationAssociation) under which they are valid.

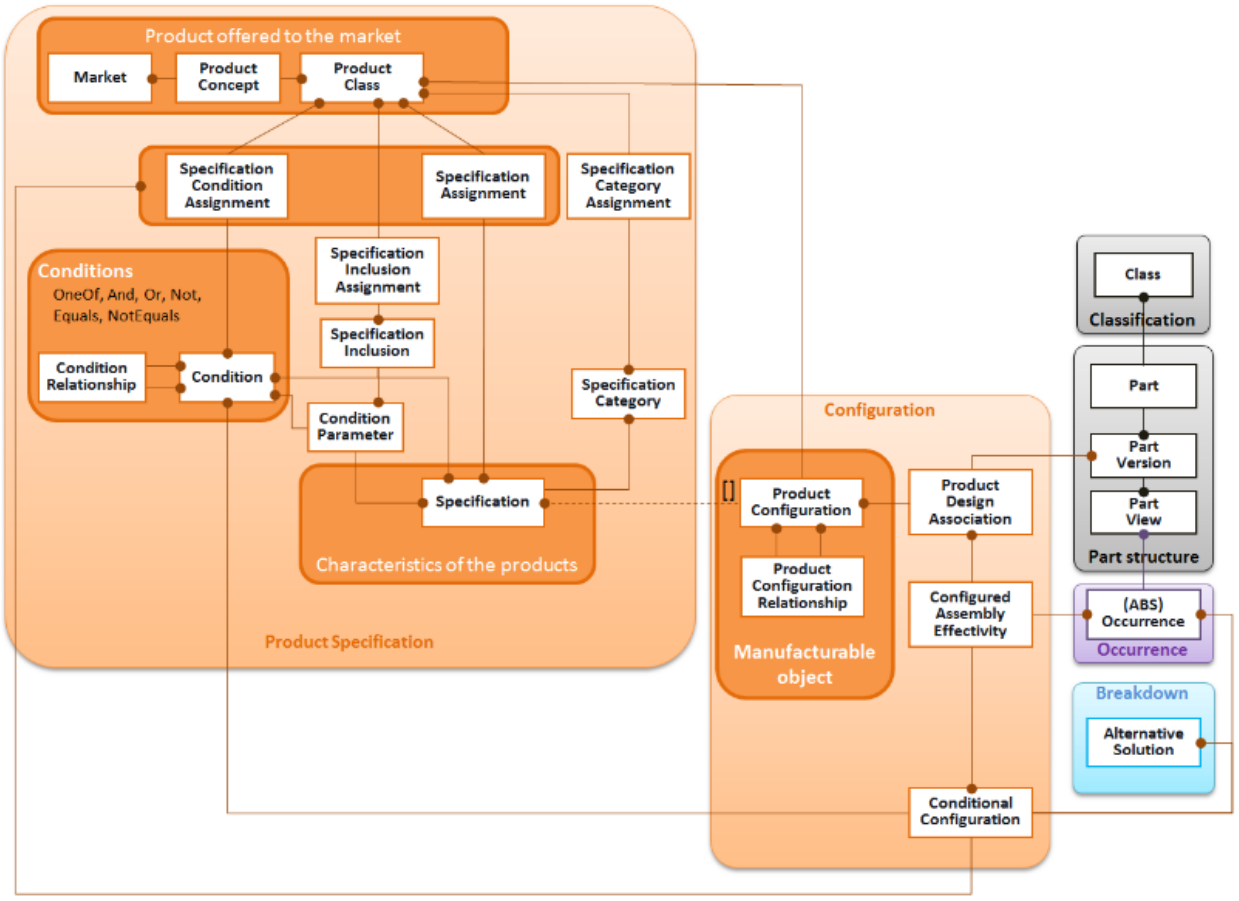


Figure 13: Configuration in ISO 10303-4442

3.4.2 Assembly structure

(See chapter 4.2.3. of STEP AP242 Domain Model)

This capability allows representing relationships between objects to build up various kinds of structures. The structures are defined between part views or between part views and part occurrences.

Among the structures supported by this capability is a hierarchical assembly structure to represent the relationships between an assembly and its constituents. For representing hierarchical assembly structures two concepts are supported by this capability: the part view-based assembly structure concept and the part occurrence-based assembly structure concept.

- The part occurrence-based assembly structure concept allows building up a hierarchical assembly structure by an AssemblyDefinition representing the assembly and one or more Occurrence objects representing its constituents.
- The part view-based assembly structure concept allows building up a hierarchical assembly structure by an AssemblyDefinition object representing the assembly and one or more Part-View objects representing its constituents.

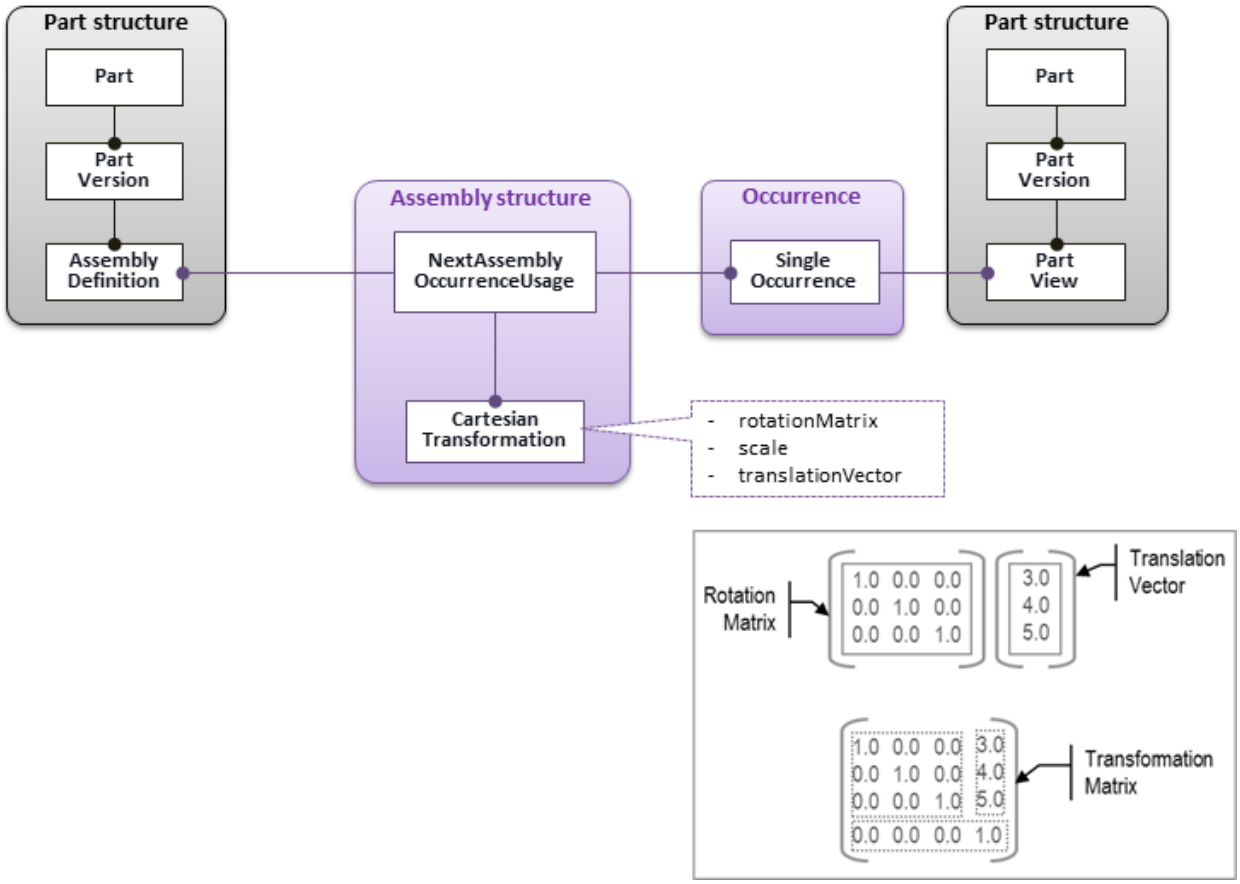


Figure 14: AssemblyStructure with Cartesian Transformation

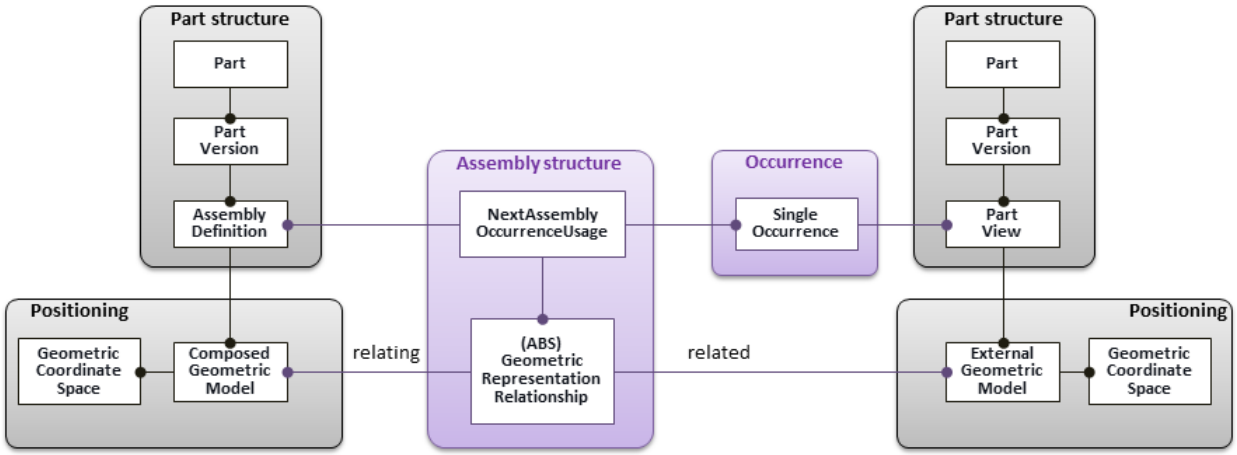


Figure 15: Assembly Structure with Geometric Representation Relationship

The exchange of assembly structures is already defined in the Recommended Practices for STEP AP242 Domain Model XML “Product & Assembly Structure” and is, therefore, out of scope of the document in hand.

3.4.3 Breakdown

(See chapter 4.2.4. of STEP AP242 Domain Model)

This capability enables the representation of parent-child structures, so-called breakdowns. A breakdown consists of breakdown elements. Different types of breakdowns and breakdown elements are distinguished, such as functional and physical. breakdowns may be views of actual product structures with relations to only some of their constituents.

This capability also enables the representation of alternative solutions, such as final, supplier or technical solutions. These are implementations of functional and physical breakdown elements.

Finally, part occurrences, that is, occurrences of parts in certain locations, can be associated with breakdown elements by means of this capability.

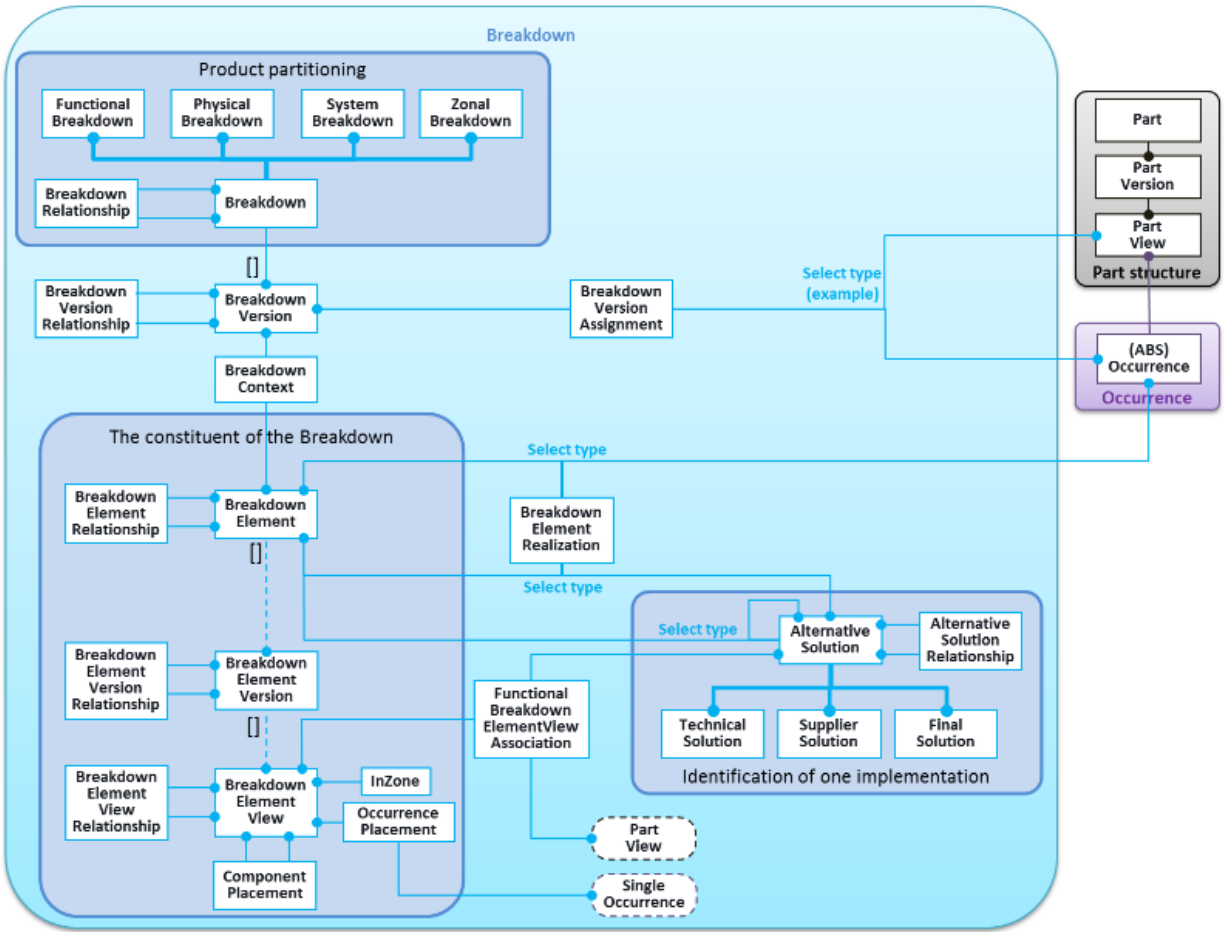


Figure 16: Breakdown Structure

The exchange of breakdowns is not covered in this document.

3.4.4 Part occurrence

(See chapter 4.2.25. of STEP AP242 Domain Model)

Part occurrence provides the capability to represent occurrences of PartView objects. It includes master data about part occurrences as well as relationships between part occurrences. Part occurrences are used to identify an occurrence of a component in an assembly structure, e.g., to identify the elements of an AlternativeSolution.

3.4.5 Part view

(See chapter 4.2.26. of STEP AP242 Domain Model)

This capability provides the ability to represent basic product management information about different views on part versions as well as relationships between different part views.

3.4.6 Effectivity

(See chapter 4.2.12. of STEP AP242 Domain Model)

The Effectivity capability represents information concerning the validity of data. Implicit propagation of data specifying validity is not available. The validity of data can be expressed by effectivities that specify, for example, time ranges within which data may be used. Explicit dates or dates expressed by events can be used in order to represent the relevant points in time. Effectivities may also express rule-based conditions (so-called ConditionalEffectivity) used for implicit configuration. Validity may also be expressed for individual parts, which are identified by serial numbers.

The usage of effectivities in AP242 is described in detail in chapter 4.2.1 .

3.4.7 Product specification

(See chapter 4.2.31. of STEP AP242 Domain Model)

The Product Specification provides the capability to describe products with a large number of variants.

Examples for aircraft products are propeller aircrafts, jet aircrafts, rocket-powered aircrafta, or engines or components of these products. Examples for automotive products are passenger cars, trucks, busses, or engines or components of these products.

Because of the large number of variants there is no unique identification for each of the variants that could be produced. The main concepts that are used to handle this large number of variants are the following:

- product classes are used to identify sets of similar products to be offered to the market
- specifications are used to describe characteristics of the products
- specification categories are used to group similar characteristics of the products
- specification conditions are used to control the usage of a part within a product and to represent conditions for product classes.

The usage of product specification is not covered in this document.

4 Configuration and Effectivity Information

This chapter will describe the templates for the usage of the AP242 Entities in the context of configuration management.

Each template will describe following content:

  - The Instance Model asAP242 Domain Model XML entities and attributes (instantiation picture)
  - AP242 Domain Model XML syntax
- Table with supported attributes
- Attribute recommendations
- Preprocessor Recommendations
- Postprocessor Recommendations

4.1 Configuration Identification

Configuration identification in AP242 is the identification of product concepts and their associated configurations, the composition of which is to be managed. If a configuration of a product concept is implemented by a certain design, i.e., a particular part version, this version can be associated with the configuration and managed using configuration effectivity.

The configuration identification supports two important concepts to describe the products that are sold by an organization to its customers:

- Product Concept Identification
  - Supports the representation of an organization's products as they are conceived or offered to the customers. A product as offered to the market can directly correspond to a manufacturable object, or it can be available to the customer in different configurations where each of these configurations defines a manufacturable object.
- Product Concept Configuration Identification
  - Supports the specification of these product configurations and their associated part designs.

4.1.1 Default Context

The default context (product concept) has to be used, if one of the involved systems does not manage the product concept.

The recommended usage of a context depends on two conditions:

- Support of the involved PLM system
- Complexities of the configuration

PLM systems are different in usage of a context for configurations. Following three features has to be supported for the Use Cases defined in chapter 3.2.

- PLM systems with a mandatory context
- Optional usage of a context
- A context is not supported by the PLM System

There are different complexities in the scenarios:

- In case of an explicit configuration with a 100% structure and only one context, we have a simple scenario with low configuration complexity and no context is needed

- In case of implicit configurations or if more than one context is used, we speak about a complex configuration scenario and a context has to be defined

Most of the PLM systems require a context. To get a most common usage of the context, independent from the involved PLM systems and independent from the complexity of the scenario, it is recommended to support the context in each processor with the feature to handle a default context.

The default context has no market association and the Id should be set to “/NULL”. This is a placeholder string where the target system has to create the context itself.

The mapping of the context is different in the PDM Systems and could be as following:

- Context as a specific item (e.g. Teamcenter, 3DExperience)
- Context as a flag at the root item of the product structure (e.g. Windchill)

The context is defined in the templates:

- ProductConceptIdentification (see chapter 4.1.2)
- ProductConceptConfigurationIdentification (see chapter 4.1.4)

The Instance Model for the default context

```
<ProductConcept uid="PC_1" xsi:type="bom:ProductClass">
  <Id id="/NULL"/>
…
</ProductConcept>
```

Preprocessor Recommendations:

If the source PLM system does not support a context, it has to be created with default values to be compliant to the recommendations.

It is not supported to use more than one context within one AP242 file.

Postprocessor Recommendations:

“/NULL” is a placeholder string already used for similar purposes in other places and means the target system has to create the context itself.

A warning has to be indicated, if more than one context is specified in the configuration.

The postprocessor recommendation depends on the context support of the involved PLM system

Target PLM systems	Recommendation
No support of a context	The context has to be ignored, even if it is not the default context.
Context is optional	If the recommended default context is used, it has to be ignored.
Context is mandatory	If the recommended default context is used, it has to be ignored and the default context of the target PLM should be used.

Table 4: Context handling recommendations

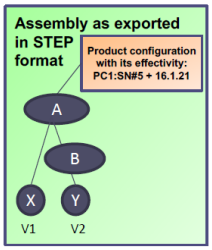
4.1.2 Exchange of one Explicit Configuration

In order to recognize that the content of a AP242 XML population is solely the result of *one* Explicit Configuration (typically 100%), the use of the ConfiguredAssemblyEffectivity (see section 4.1.4.5) is not recommended.

Preprocessor Recommendations:

- The ExchangeContext.IdentificationContext shall be set to the uid of the ProductConfiguration.Id, saying that the whole assembly structure is the result of a ProductConfiguration.
- In case the ProductConfiguration and/or the Effectivities involve Specifications, only those Specifications and their SpecificationCategory(s) shall be exported. The fact that ExchangeContext.IdentificationContext is set to a ProductConfiguration.Id.uid tells, that the ProductClass and the SpecificationCategory(s) contain only the Specifications involved in the ProductConfiguration

The following example shows the result assembly structure (100%) for the ProductConfiguration called 'PC1' that filtered the 150% BoM via two Effectivities: SerialNumber #5 and date 16.1.21. The recommended use of ProductDesignAssociation between ProductConfiguration and the top node of the Assembly is not affected by this simplified representation and shall comply to section 4.1.4.2.



Central reference to ProductConfiguration is enough to say: this file contains the result of the filtering of a 150% product structure

	Object	Attribute	Id	idRoleRef	idContextRef
	ExchangeContext	IdentificationContext			Uid of Identifier of the product conf object
	Organization (uid=org-1)	Identifier	BMW		
Special case: multiple ids	Organization (uid=org-2)	Identifier	Bosch		
	ProductConfiguration	Identifier	PC1	identification information	org-1
Ditto for Part B, X, Y	Part A (uid=p-1)	Identifier	A	exchange identification information	org-1
Special case: multiple ids		Identifier	P1	identification information	org-2
	ProductClass	Identifier	MTB	identification information	Org-1
		LevelType = product family			
		Version id = A			

Figure 17: Example for the exchange of one Explicit Configuration

This simplified representation is not recommended if the AP242 XML population contains the result of more than one Explicit Configuration, or if it contains the full 150% BoM.

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <ExchangeContext uid="EC--0">
…
      <IdentificationContext uidRef="PC__600001--id"/>
    </ExchangeContext>


    <!--************    ProductConfiguration UC1 additional information
************-->
    <ProductConcept uid="PC__600000" xsi:type="n0:ProductClass">
…
      <ProductConfiguration uid="PC__600001">
        <Id>
          <Identifier id="PC1" idContextRef="O__3" idRoleRef="C__7"
uid="PC__600001--id"/>
        </Id>
…
```

```
      <ProductDesignAssociation uid="PDA__600002">
        <AssociatedDesign uidRef="PVe__9"/>
        <Description>
          <CharacterString>filtering information</CharacterString>
        </Description>
      </ProductDesignAssociation>
      <EffectivityAssignment uid="EA__600004">
        <AssignedEffectivity uidRef="SE__600003"/>
        <EffectivityIndication>true</EffectivityIndication>
        <Role>
          <ClassString>actual</ClassString>
        </Role>
      </EffectivityAssignment>
    </ProductConfiguration>
  </ProductConcept>
  <Effectivity uid="SE__600003" xsi:type="n0:SerialEffectivity">
    <EffectivityContext uidRef="PC__600000"/>
    <EndId>
      <IdentifierString>5</IdentifierString>
    </EndId>
    <StartId>
      <IdentifierString>5</IdentifierString>
    </StartId>
  </Effectivity>
```

4.1.3 Template “ProductConceptIdentification”

The products that are conceived or offered by an organization to its customers are often defined as variations or configurations of a common product model, referred to as product concept. This product concept is defined in a market context and can be seen as a logical container for all of its variations. In the AP242 Schema, product concept identification is the representation of these product concepts with their related market context. A product as offered to the market can directly correspond to a manufacturable object, or it can be available to the customer in different configurations where each of these configurations defines a manufacturable object.

The Instance Model: AP242 Domain Model XML entities and attributes

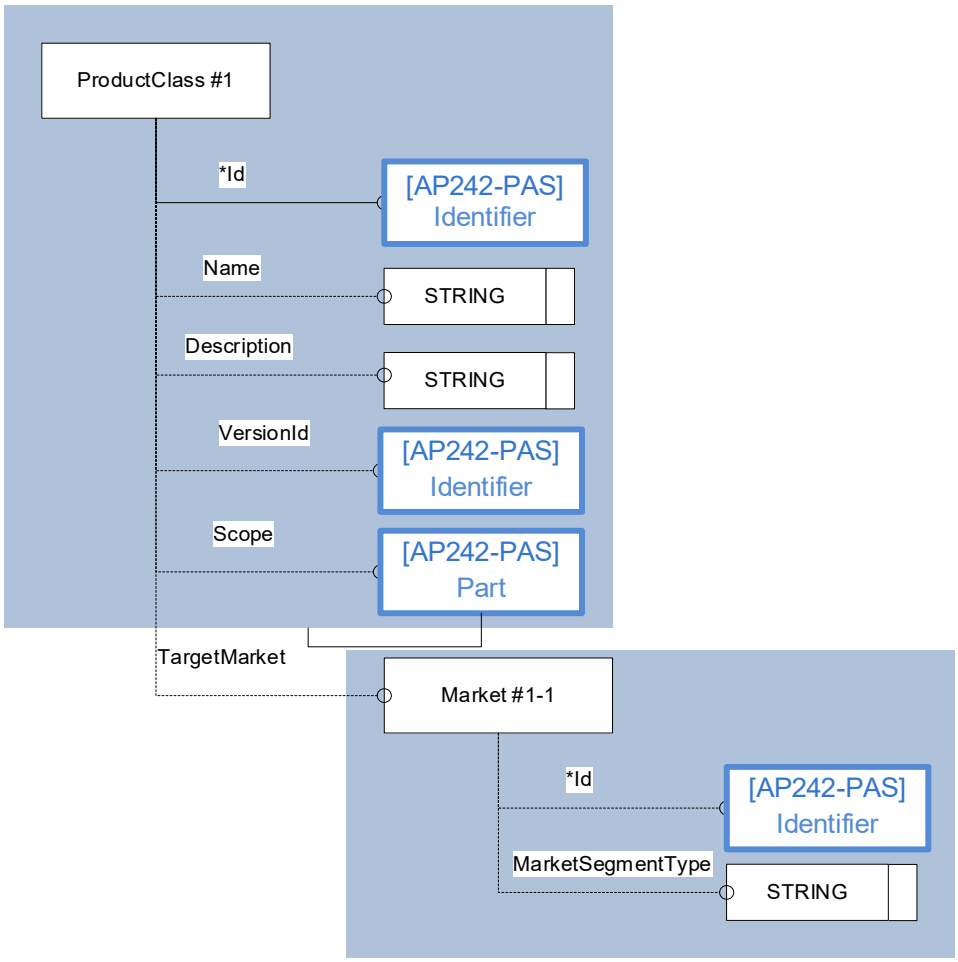


Figure 18: Instance Model ProductConceptIdentification

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <ProductConcept uid="pc--19088" xsi:type="n0:ProductClass">
      <Description>
        <CharacterString>as class description</CharacterString>
      </Description>
      <Id>
        <Identifier uid="pc-asclass--id1" id="as class" idRoleRef="rl--ii"
idContextRef="o--000000178"/>
      </Id>
      <Name>
        <CharacterString>class name</CharacterString>
      </Name>
      <LevelType>
        <ClassString>platform</ClassString>
      </LevelType>
      <VersionId id="A.1"/>
…
    </ProductConcept>
    <Market uid="mk--1">
      <MarketSegmentType>
        <Class uidRef="cl--mk1"/>
      </MarketSegmentType>
    </Market>
    <Class uid="cl--mk1">
```

```
      <DefinedIn uidRef="ecs--mk"/>
      <Id id="automotive"/>
    </Class>
    <ExternalClassSystem uid="ecs--mk">
      <Id>
        <Identifier uid="ecs--id1" id="customized part types" idRoleRef="rl--
ii" idContextRef="o--000000178"/>
      </Id>
    </ExternalClassSystem>
```

4.1.3.1 ENTITY ProductClass

A product concept describes a class of similar products that an organization provides to its customers. It represents the idea of a product as identified by potential or actual customer requirements. Therefore, a product concept may exist before the parts have been defined that implement it. Depending on the kind of industry and products, a product concept might be offered to the customers in one or many different configurations. If exactly one configuration is defined, the product concept corresponds to a particular part design. If the product concept is offered in different configurations, each of these configurations is a member of the class of products defined by this product concept.

Attribute Name	Attribute Type
ClassifiedAs	OPTIONAL SET[1:?] of ClassifiedAsType
Description	OPTIONAL DescriptorSelect
Id	IdentifierSelect
Name	OPTIONAL DescriptorSelect
ProductConfiguration	OPTIONAL SET[1:?] of ProductConfiguration
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
SameAs	OPTIONAL SET[1:?] of SameAsType
TargetMarket	OPTIONAL Market
AnalysisAssignment	OPTIONAL SET[1:?] of AnalysisAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssignment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganizationAssignment
ProjectAssignment	OPTIONAL SET[1:?] of ProjectAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
StateAssignment	OPTIONAL SET[1:?] of StateAssignment
StateDefinitionAssignment	OPTIONAL SET[1:?] of StateDefinitionAssignment
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment
LevelType	OPTIONAL ClassSelect

Scope	OPTIONAL Part
VersionId	OPTIONAL IdentifierSelect
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
BreakdownVersionAssignment	OPTIONAL SET[1:?] of BreakdownVersionAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssignment
ProductClassRelationship	OPTIONAL SET[1:?] of ProductClassRelationship
SecurityClassificationAssignment	OPTIONAL SET[1:?] of SecurityClassificationAssignment
SpecificationAssignment	OPTIONAL SET[1:?] of SpecificationAssignment
SpecificationCategoryAssignment	OPTIONAL SET[1:?] of SpecificationCategoryAssignment
SpecificationConditionAssignment	OPTIONAL SET[1:?] of SpecificationConditionAssignment
SpecificationInclusionAssignment	OPTIONAL SET[1:?] of SpecificationInclusionAssignment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRelationship

Table 5: "ProductClass" Attributes

Attribute recommendations:

- **ClassifiedAs**: the classifications of the ProductClass. The value of this attribute need not be specified except for nested files (see [242-PAS] 9.3) and incremental exchange (see [242-PAS] 9.4). Use “Classification” template (see [242-PAS]).
- **Id:** The id attribute specifies the identifier of the product concept. It is usually assigned by the organization that provides the product belonging to that product concept to its customers. This identifier shall be unique within the given IdentificationContext. Use “Identifier” template (see [242-PAS]).
- **Name:** The name attribute specifies the nomenclature or common name, by which the product concept is referred. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- **Description:** The description attribute specifies additional information about the product concept. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- **TargetMarket:** The market for the products belonging to a product concept is further described in the market_context attribute. The value of this attribute need not be specified.
- ***VersionId*:** the identification or set of identifications of a specific version of the ProductClass. The value of this attribute need not be specified, except if VersionBranchEffectivities are defined using this ProductClass (see section 4.2.6). Use IdentifierString type or "Identifier" template (see [242-PAS]).

- **LevelType:** the level of the ProductClass in a hierarchical structure of ProductClass objects. Although the value of this attribute need not be specified, it is recommended not to leave it unset. Use ClassString type if one of the values below is used, otherwise use “Class” template (see [242-PAS] 4.6.4). According to prostep ivip CC8 Recommended Practices, Version 1.2b, where applicable, the following values shall be used:

RelationType	
‘enterprise‘	The highest level of the ProductClass instance in a file shall always represent the company. This is necessary as some systems have this class as a general starting point. The id of this instance will be a unique identifier for the relevant company. This class carries no variant control information but only characteristics (e.g. Specification, SpecificationCategory etc.) of that class.
‘platform‘	This instance of the ProductClass reflects the level of a product class in a BoM system which represents a main technical product base (e.g. project, platform, engineering series etc.). In some cases, this level carries a complete BoM ("Maximum BoM") for a project, platform, engineering series etc. This level is in some cases called technical documentation. This level carries only characteristics of this class level, no variant control mechanisms are attached.
‘product family’	This instance will be the one to which all variant control mechanisms are attached. For trucks configuration information (Specifications, Conditions etc.) may be associated on level three and four. For cars this may only be attached on level three.
‘furthest pre-configured abstract product class’	Furthest pre-configured abstract ProductClass: this instance represents the furthest configured class of a product, which is not yet a real product. E.g. this could be a complete vehicle, engine, gearbox etc. which has not been evaluated against customer special choices or an abstract vehicle, engine, gearbox etc. which could become a real one after the associated BoM is evaluated. The purpose of this level of a product class instance is in any case to reflect that level of ProductClass of a BoM system which leads to the individual BoM for a single product. This ProductClass level will be particularly exchanged in the context of scenarios to exchange a single BoM.

- ***Scope:*** the top level Part that the ProductClass is associated to. The value of this attribute need not be specified. Use "Part" template (see [242-PAS]), especially in case VersionBranchEffectivities are involved (see section 4.2.6).
- ***ProductConfiguration:*** to assign (optionally) one or multiple ProductConfigurations to define manufacturable variants of the ProductClass. Use the “ProductConfiguration” template (see 4.1.4.1).
- ***ProductClassRelationship*:** to relate to another ProductClass. Use the “ProductClassRelationship” template, see 4.1.4.4.
- **PropertyValueAssignment**: to assign a PropertyValue to the ProductClass, especially used for validation properties (see section 7.1.1). Use the “PropertyAssignment” template; see [242-PAS] 6.2 for details.
- ***SpecificationAssignment:*** to assign (optionally) one or multiple Specifications to the ProductClass. Use the “SpecificationAssignment” template (see 4.3.3).

- ***SpecificationCategoryAssignment:*** to assign (optionally) one or multiple SpecificationCategories to the ProductClass. Use the “SpecificationCategoryAssignment” template (see 4.3.3).
- ***SpecificationConditionAssignment:*** to assign (optionally) one or multiple Conditions to the ProductClass. Use the “SpecificationConditionAssignment” template (see 4.3.4).
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

There might be several levels of product concepts defined in a company. If product concept is not supported, the default context has to be used as product concept. (see chapter 4.1.1)

Especially if VersionBranchEffectivities are involved, multiple versions of a ProductClass are needed. In such a case, since ProductClass.Id and ProductClass.VersionId are in the same object, all versionless attributes of ProductClass (Id, Classification, Level, …) shall be redundantly mapped to all versions, including the multiple identifiers.

Postprocessor Recommendations:

If not supported by the target system, TargetMarket shall be ignored, or a warning shall be returned.

Remark: a product concept represents a conceptual idea of a class of similar products that are offered to a market. No design or manufacturing related product data can be attached to the product concept directly.

4.1.3.2 ENTITY Market

The Market entity is a subtype of BaseRootObject. It defines the market context in which a product concept is defined and may include information characterizing the potential customers of the products of the associated product concept. The application domain is identified by the associated application_context entity.

Attribute Name	Attribute Type
ClassifiedAs	OPTIONAL SET[1:?] of ClassificationSelect
Id	OPTIONAL IdentifierSelect
MarketSegmentType	OPTIONAL ClassSelect
Name	OPTIONAL DescriptorSelect
SameAs	OPTIONAL SET[1:?] of ProxySelect

Table 6: "Market" Attributes

Attribute recommendations:

- **MarketSegmentType:** The market segment type is a label that identifies the kind of consumer preferences associated with a product concept. Use “Class” template (see [242-PAS].
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

There are no specific preprocessor recommendations.

Postprocessor Recommendations:

There are no specific postprocessor recommendations.

4.1.4 Template "ConfigurationItem"

This is the explicit identification of the configurations that exist for a given product concept and the representation of their composition. A Configuration Item is specified as a ProductConfiguration entity. Product configurations can be related to appropriate part versions to specify the designs that implement the product configurations through the configuration design.

The Instance Model: AP242 Domain Model XML entities and attributes

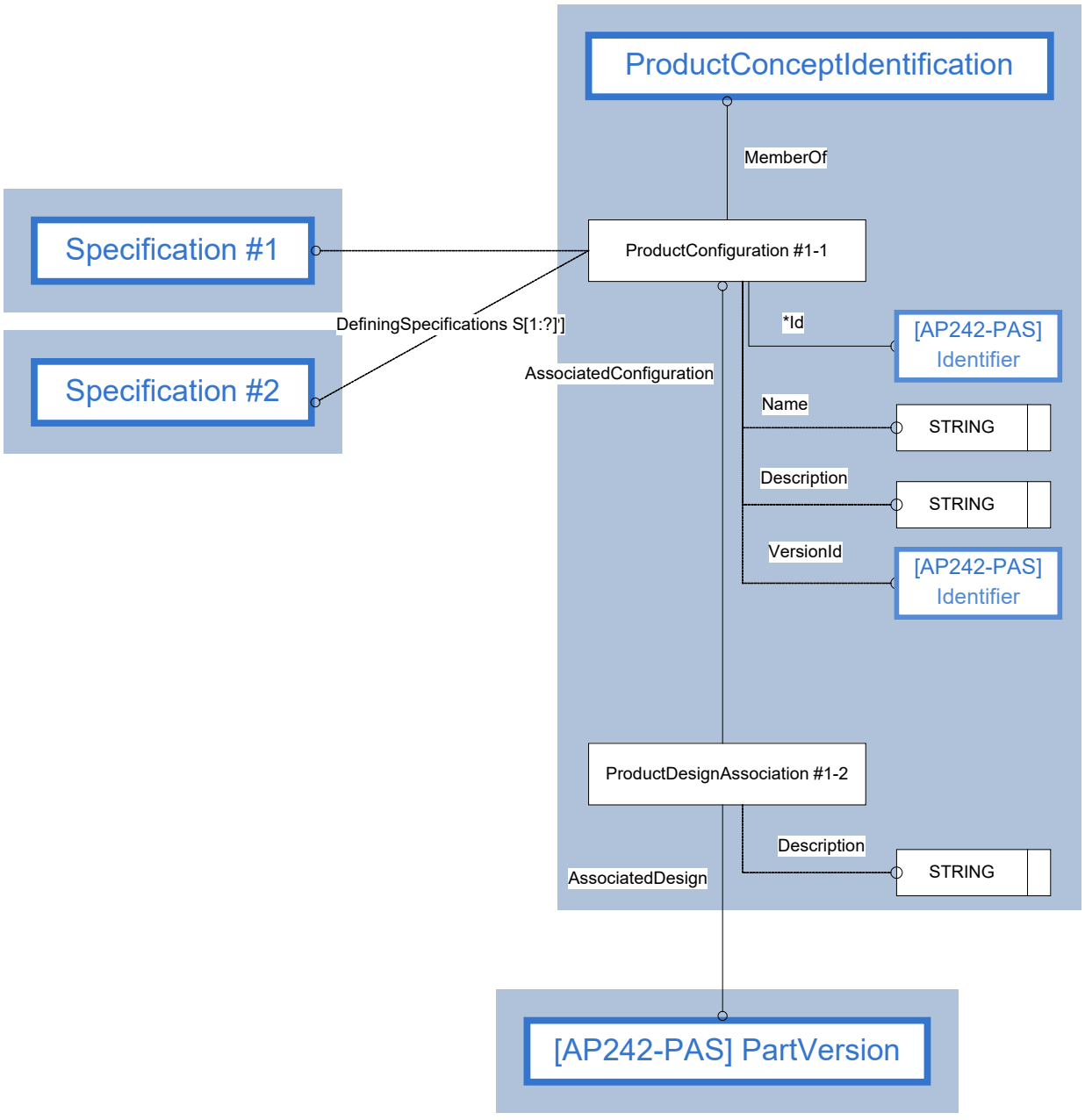


Figure 19: Instance Model ProductConceptConfigurationIdentification

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<ProductConcept uid="pc--19088" xsi:type="n0:ProductClass">
…
  <ProductConfiguration uid="pconf--19089">
    <Description>
      <CharacterString>as1 short description</CharacterString>
    </Description>
    <Id>
      <Identifier uid="pc-as1short--id1" id="as1 short" idRoleRef="rl--ii"
idContextRef="o--000000178"/>
    </Id>
    <Name>
      <CharacterString>as1 short name</CharacterString>
    </Name>
    <ProductDesignAssociation uid="pda--4711">
      <AssociatedDesign uidRef="pv--0000000017D374A0--id1"/>
    </ProductDesignAssociation>
    <VersionId id="A.1"/>
            …
  </ProductConfiguration>
</ProductConcept>

<Part uid="p--0000000017D374A0">
…
  <Id>
    <Identifier uid="pid--0000000017D374A0--id1" id="as1" idRoleRef="rl--ii"
idContextRef="o--000000178"/>
  </Id>
…
  <Versions>
    <PartVersion uid="pv--0000000017D374A0--id1">
…
    </PartVersion>
  </Versions>
</Part>
```

4.1.4.1 ENTITY ProductConfiguration

The ProductConfiguration entity is a key concept to support explicit and implicit configuration management. It represents the identification of a particular Configuration Item, i.e., variation of a product concept. A ProductConfiguration is defined with respect to the product concept, i.e., the class of similar products of which it is a member.

The ProductConfiguration defines a manufacturable end item, or something that is conceived and expected as such. The association between a ProductConfiguration and a corresponding part design is established using a ProductDesignAssociation. The valid use of component parts for planned units of manufacturing of a particular ProductConfiguration may be specified using configuration effectivity (see 4.2.1).

In case the ProductConfiguration is controlled by an Effectivity and/or Specifications, the use of the subtype EffectivityControlledProductConfiguration is recommended.

Attribute Name	Attribute Type
ClassifiedAs	OPTIONAL SET[1:?] OF Classification
DefiningGeometry	OPTIONAL GeometricModel
DefiningSpecifications	OPTIONAL SET[1:?] OF Specification

Description	OPTIONAL DescriptorSelect
Id	IdentifierSelect
Name	OPTIONAL DescriptorSelect
Occurrence	OPTIONAL SET[1:?] OF DefinitionBasedOccurrence
ProductDesignAssociation	OPTIONAL SET[1:?] OF ProductDesign Association
RetentionPeriod	OPTIONAL SET[1:?] OF RetentionPeriod
SameAs	OPTIONAL S [1:n] OF ProxySelect
ShapeElement	OPTIONAL SET[1:?] OF ShapeElement
VersionId	OPTIONAL IdentifierSelect
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
BreakdownVersionAssignment	OPTIONAL SET[1:?] of BreakdownVersionAssignment
CertificationAssignment	OPTIONAL SET[1:?] of CertificationAssignment
ContractAssignment	OPTIONAL SET[1:?] of ContractAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssign-ment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssign-ment
ObservationAssignment	OPTIONAL SET[1:?] of ObservationAssignment
OrganizationOrPersonInOrganiza-tionAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrgani-zationAssignment
ProductConfigurationRelationship	OPTIONAL SET[1:?] of ProductConfigurationRelationship
ProjectAssignment	OPTIONAL SET[1:?] of ProjectAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
RequirementAssignment	OPTIONAL SET[1:?] of RequirementAssignment
SecurityClassificationAssignment	OPTIONAL SET[1:?] of SecurityClassifica-tionAssign-ment
StateAssignment	OPTIONAL SET[1:?] of StateAssignment
StateDefinitionAssignment	OPTIONAL SET[1:?] of StateDefinitionAssignment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRelationship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment
WorkRequestAssignment	OPTIONAL SET[1:?] of WorkRequestAssignment

Table 7: "ProductConfiguration" Attributes

Attribute recommendations:

- **ClassifiedAs**: the classifications of the ProductConfiguration. The value of this attribute need not be specified except for nested files (see [242-PAS] 9.3) and incremental exchange (see [242-PAS] 9.4). Use “Classification” template (see [242-PAS]).
- **DefiningSpecifications**: deprecated and shall not be used anymore, but for STEP files former to Edition 4, it shall still be supported: in case of implicit configurations, the set of Specification objects necessary to discriminate the SpecificationAssignment within its ProductClass. Each definingSpecifications shall be associated with a ProductClass by a SpecificationAssignment with associationType not equal 'part usage'. The associated ProductClass shall be either the associated ProductClass of the ProductConfiguration or any higher level ProductClass related directly or indirectly by a ProductClassRelationship with a relationType 'hierarchy'. The value of this attribute need not be specified. Use “Specification” template.
- **Description:** the text or the set of texts that provide further information about the ProductConfiguration. The value of this attribute need not be specified. Use "Description" template (see [242-PAS]).
- **Id:** The id attribute specifies an identifier that distinguishes the ProductConfiguration. The identifier or set of identifiers for the ProductConfiguration. (shall be unique in relation with a specific ProductConcept). Use “Identifier” template (see [242-PAS]): Here the use of IdentifierString is not recommanded due to the use of ProductConfiguration.Id as ExchangeContext.IdentifierContext.
- **Name:** the words or set of words by which the ProductConfiguration is known. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]
- **VersionId**: the identification or set of identifications of a particular version of the ProductConfiguration. The value of this attribute need not be specified. Use IdentifierString type or "Identifier" template (see [242-PAS]).
- **Occurrence**: the instantiations of the ProductConfiguration. The element Occurrence itself cannot be instantiated. For the purpose of the Electrical Wiring Harness (see [242-EWH] 6.1, only the subtype “SingleOccurrence” (see [242-PAS] 7.1) shall be used in case the part number is not decided yet but only its characteristics (for example the voltage and power supply of a batterie).
- **ProductDesignAssociation*:*** to assign (optionally) one or multiple ProductDesignAssociations to define the explicit design that corresponds to this Configuration. Use the “ProductDesignAssociation” template (see 4.1.4.2).
- ***EffectivityAssignment:*** deprecated and shall not be used anymore, but for STEP files former to Edition 4, it shall still be supported: to assign (optionally) one or multiple Effectivities to define the Configuration. Use the “EffectivityAssignment” template (see 4.2.8.3).
- **ProductConfigurationRelationship:** to relate to another ProductConfiguration. Use the “ProductConfigurationRelationship” template, see 4.1.4.4.
- **PropertyValueAssignment**: to assign a PropertyValue to the ProductConfiguration. See [242-PAS] 6.2 for details, especially used for validation properties (see section 7.1.2) and in the area of Electrical Wiring Harness (see [242-EWH] 6.
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

There is no standard mapping for the id attribute of ProductConfiguration; however, the value should be unique in conjunction with the id attribute of the associated ProductConcept.

If the ProductConfiguration.Id is unique independently from the ProductClass, idContextRef may reference an Organization, otherwise it shall reference the ProductClass.Identifier .uid (as in "Identifier" template, see [242-PAS]).

If multiple versions of a ProductConfiguration are involved, since ProductConfiguration.Id and ProductConfiguration.VersionId are in the same object, all versionless attributes of ProductConfiguration (Id, Classification, …) shall be redundantly mapped to all versions, including the multiple identifiers.

Postprocessor Recommendations:

There are no specific postprocessor recommendations.

Remarks:

A ProductConfiguration can be established prior to the existence of a corresponding part design, i.e., a ProductDesignAssociation does not need to exist for a ProductConfiguration.

If the design of a ProductConfiguration is made of many distinct assemblies, more than one ProductDesignAssociation may exist to this ProductConfiguration.

Deprecated and shall not be used anymore, but for STEP files former to Edition 4, it shall still be supported: The DefiningSpecifications shall include all necessary Specifications in order to define a manufacturable end item. Usually, this is one particular variant of the product. The same applies to the Effectivities referenced by EffectivityAssignment: here no serial number intervals, date or event intervals are expected, but one given serial number, date/event and/or lot.

4.1.4.2 ENTITY EffectivityControlledProductConfiguration

An EffectivityControlledProductConfiguration is a type of ProductConfiguration who is controlled by an Effectivity.

EXAMPLE A ProductConfiguration may apply to a given Serialnumber(-interval), a given Lot, a given Date(-interval), a given Event(-interval), ..., or a conditional combination of them.

Additional attributes, derived from entity ProductConfiguration will not be repeated in this table

Attribute Name	Attribute Type
Definition	OPTIONAL Effectivity

Table 8: "EffectivityControlledProductConfiguration" Attributes

Attribute recommendations:

- ***Definition:*** by which the EffectivityControlledProductConfiguration is controlled. Use the "Effectivity" templates (see 4.2).

Preprocessor Recommendations:

If no Effectivity controls the ProductConfiguration, it is recommended to use the supertype object ProductConfiguration rather than EffectivityControlledProductConfiguration.

If EffectivityControlledProductConfiguration is used, the inherited attribute DefiningSpecifications shall not be used, nor EffectivityAssignments.

Postprocessor Recommendations:

There are no specific postprocessor recommendations.

4.1.4.3 ENTITY ProductDesignAssociation

A ProductDesignAssociation is a mechanism to associate a PartVersion with its corresponding ProductConfiguration. It specifies the explicit design that corresponds to the ProductConfiguration. The ProductDesignAssociation represents the statement that, in all definition contexts, the PartVersion is a valid way to implement the ProductConfiguration.

Attribute Name	Attribute Type
AssociatedDesign	PartVersion
ClassifiedAs	OPTIONAL SET[1:?] of Classification
ConfiguredAssemblyEffectivity	OPTIONAL SET[1:?] of ConfiguredAssemblyEffectivity
Description	OPTIONAL DescriptorSelect
AssociationObjectRelationship	OPTIONAL SET[1:?] of AssociationObjectRelationship
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment

Table 9: "ProductDesignAssociation" Attributes

Attribute recommendations:

- **AssociatedConfiguration**: the ProductConfiguration that represents the requirements for which a PartVersion as solution is designated.
- **AssociatedDesign**: the PartVersion that identifies a part which is a valid implementation for the ProductConfiguration, i.e. meets the requirements specified by the ProductConfiguration. Use “Part” template (see [242-PAS].
- **Description**: the text or the set of texts that provide further information about the ProductDesignAssociation. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- **ConfiguredAssemblyEffectivity*:*** to assign (optionally) one or multiple ConfiguredAssemblyEffectivities to define the occurrences in the full variant product structure that apply to this ProductConfiguration. Use the “ConfiguredAssemblyEffectivity” template (see 4.1.4.5).
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

The ProductDesignAssociation should only be used for the top node of the product structure

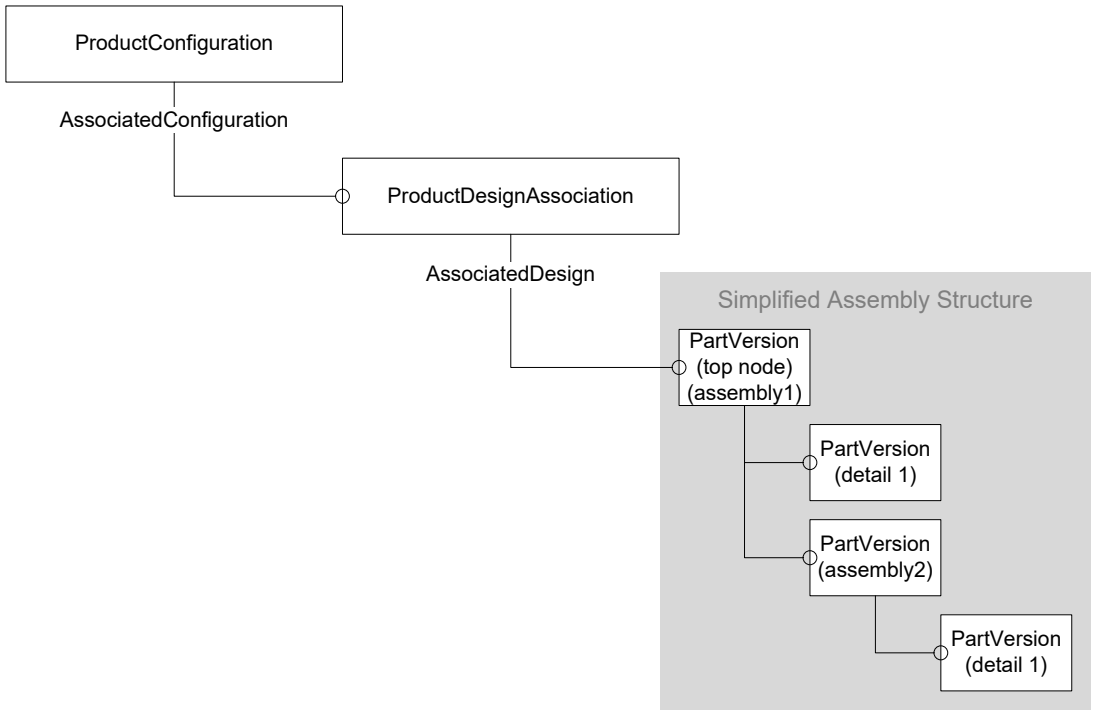


Figure 20: Simplified Example for ProductDesignAssociation for the top node

Postprocessor Recommendations:

There are no specific postprocessor recommendations.

4.1.4.4 ENTITY ProductConfigurationRelationship

This relationship enables to relate two ProductConfigurations.

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<ProductConfiguration uid="pconf--19089">
…
      <Id>
            <Identifier uid="pc-as1short--id1" id="as1 short" idRoleRef="rl--
ii" idContextRef="o--000000178"/>
      </Id>
…
      <VersionId id="A.1"/>
…
      <ProductConfigurationRelationship uid="pcr--1">
            <Related uidRef="pconf--19089b"/>
            <RelationType>
                  <ClassString>sequence</ClassString>
            </RelationType>
      </ProductConfigurationRelationship>
</ProductConfiguration>
<ProductConfiguration uid="pconf--19089b">
…
      <Id>
            <Identifier uid="pc-as1short--id1b" id="as1 short" idRoleRef="rl-
-ii" idContextRef="o--000000178"/>
      </Id>
…
      <VersionId id="B.1"/>
</ProductConfiguration>
```

Entity ProductConfigurationRelationship attributes	Attribute type
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
Id	OPTIONAL IdentifierSelect
Related	ProductConfiguration
RelationType	ClassSelect
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssign-ment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganizationAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
RelationshipObjectRelationship	OPTIONAL SET[1:?] of RelationshipObjectRelationship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 10: "ProductConfigurationRelationship" Attributes

Attribute recommendations

- ***RelationType***: the meaning of the relationship. Use ClassString type if one of the values below is used, otherwise use "Class" template (see [242-PAS]). According to the ISO AP242 Specification, where applicable, the following values shall be used:

RelationType	
'hierarchy'	the ProductConfigurationRelationship defines a relationship where the relating ProductConfiguration is on a higher level in the hierarchy of ProductConfiguration objects than the related ProductConfiguration;
'sequence'	The business object defines a logical sequence where the related configuration comes after the relating configuration. In case the Id is equal and VersionId of both ProductConfigurations is different, a version sequence is defined.

- ***Related***: the other object of **ProductConfiguration** that is part of the relationship
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

4.1.4.5 ENTITY ConfiguredAssemblyEffectivity

This object is an effectivity controlled association of a ProductDesignAssociation with Occurrences (and potentially also with AssemblyViewRelationships, DefinitionalPartViewUsages or PartShapeElements, not in scope of this document).

It shall be only used if the top assembly PartVersion node referenced by ProductDesignAssociation.associatedDesign contains more than one explicit configurations of the product.

In this case, it is recommended to associate the ConfiguredAssemblyEffectivity to Occurrences, telling that in a full variant product structure, all (and only these) occurrences apply to the ProductConfiguration where this ConfiguredAssemblyEffectivity is embedded into.

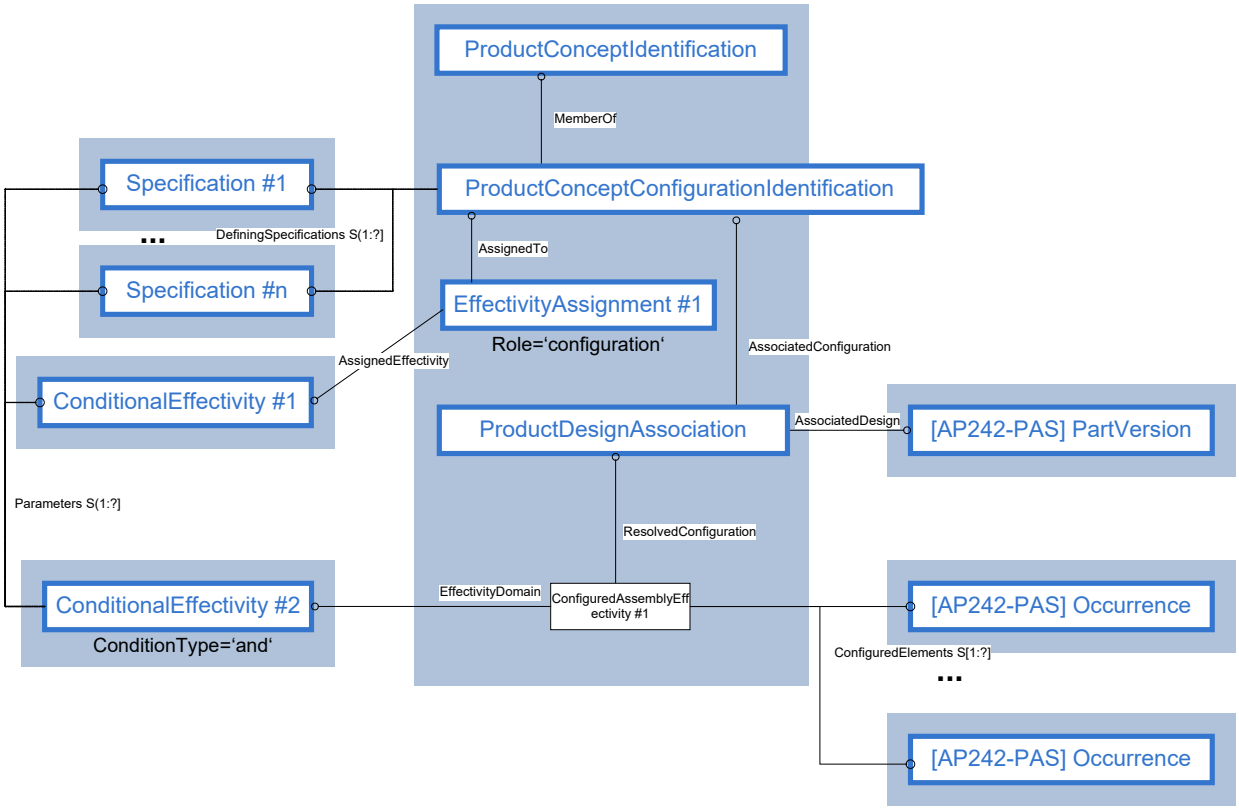


Figure 21: Template "ConfiguredAssemblyEffectivity"

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<ProductConcept uid="pc--19088" xsi:type="n0:ProductClass">
…
      <ProductConfiguration uid="pconf--19089">
…
            <ProductDesignAssociation uid="pda--4711">
                  <AssociatedDesign uidRef="pv--0000000017D374A0--id1"/>
                  <ConfiguredAssemblyEffectivity uid="cae--1">
                        <ConfiguredElements>
                              <Occurrence uidRef="pi--000000001EB04CF0--10"/>
                              <Occurrence uidRef="pi--0000000017D37010--11"/>
                              <Occurrence uidRef="pi--000000001EAA8110--14"/>
                        </ConfiguredElements>
                        <EffectivityDomain uidRef="sneff-11"/>
                  </ConfiguredAssemblyEffectivity>
```

```
            </ProductDesignAssociation>
...
        </ProductConfiguration>
...
</ProductConcept>
```

Entity ConfiguredAssemblyEffectiv-ity attributes	Attribute type
ClassifiedAs	OPTIONAL SET[1:?] of Classification
ConfiguredElements	SET[1:?] of AssemblyDesignSelect
Description	OPTIONAL DescriptorSelect
EffectivityDomain	OPTIONAL Effectivity
Id	OPTIONAL IdentifierSelect
Name	OPTIONAL DescriptorSelect
RetentionPeriod	OPTIONAL SET[1:?] OF RetentionPeriod
SameAs	OPTIONAL S [1:n] OF ProxySelect
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssignment
OrganizationOrPersonInOrganiza-tionAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganiza-tionAssignment
SecurityClassificationAssignment	OPTIONAL SET[1:?] of SecurityClassifica-tionAssignment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRelationship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 11: "ConfiguredAssemblyEffectivity" Attributes

Attribute recommendations

- ***ConfiguredElements***: the objects that are controlled by a ConfiguredAssemblyEffectiv-ity. Use "Occurrence" template (see [242-PAS]).
- ***EffectivityDomain***: the Effectivity that determines the validity of the ConfiguredAssem-blyEffectivity. The value of this attribute need not be specified. If no value is provided, the referenced ConfiguredElements are *never* effective for theProductConfiguration.
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange part-ners.

Preprocessor Recommendations:

- All the Occurrences referenced via ConfiguredElements shall be within the product struc-ture defined underneath the top assembly PartVersion node referenced by the ProductDesignAssociation the ConfiguredAssemblyEffectivity is embedded into.

- It is recommended to define one instance of ConfiguredAssemblyEffectivity for each ProductConfiguration.
- The Effectivity referenced by EffectivityDomain shall be consistent with the one defined on the ProductConfiguration, including both
  - the Specifications referenced by ProductConfiguration. (see section 4.1.4.1)
  - the Effectivity referenced by the EffectivityAssignment defined on the ProductConfiguration (see section 4.2.8.3).
- Leaving EffectivityDomain unset implies, that all the whole Occurrences within the top assembly PartVersion node referenced by the ProductDesignAssociation do apply except the ones referenced by ConfiguredElement. Since this is not supported by most PDM systems, it is not recommended.

4.1.5 Template “ProductClassRelationship”

A ProductClassRelationship is a relationship between two ProductClass objects. The meaning of this relationship is specified further by the attribute relationType.

Attribute Name	Attribute Type
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
Id	OPTIONAL IdentifierSelect
Related	ProductClass
RelationType	ClassSelect
RetentionPeriod	OPTIONAL SET[1:?] OF RetentionPeriod
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssign-ment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssign-ment
OrganizationOrPersonInOrganiza-tionAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrgani-zationAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
RelationshipObjectRelationship	OPTIONAL SET[1:?] of RelationshipObjectRelationship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 12: "ProductClassRelationship" Attributes

Attribute recommendations:

- ***RelationType***: the meaning of the relationship. Use ClassString type if one of the values below is used, otherwise use “Class” template (see [242-PAS] 4.6.4). According to the ISO AP242 Specification, where applicable, the following values shall be used:

RelationType	
‘derivation’	the ProductClassRelationship defines a relationship where the related ProductClass is derived from the relating ProductClass;
‘hierarchy’	the ProductClassRelationship defines a relationship where the relating ProductClass is on a higher level in the hierarchy of ProductClass objects than the related ProductClass;
‘substitution’	the ProductClassRelationship defines a relationship where the related ProductClass replaces the relating ProductClass
'sequence'	the ProductClassRelationship defines a logical sequence where the related ProductClass comes after the relating ProductClass. In case the Id is equal and VersionId of both ProductClasses is different, a version sequence is defined.

- ***Related***: the other object of ProductClass that is part of the relationship
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

The Instance Model: AP242 Domain Model XML entities and attributes

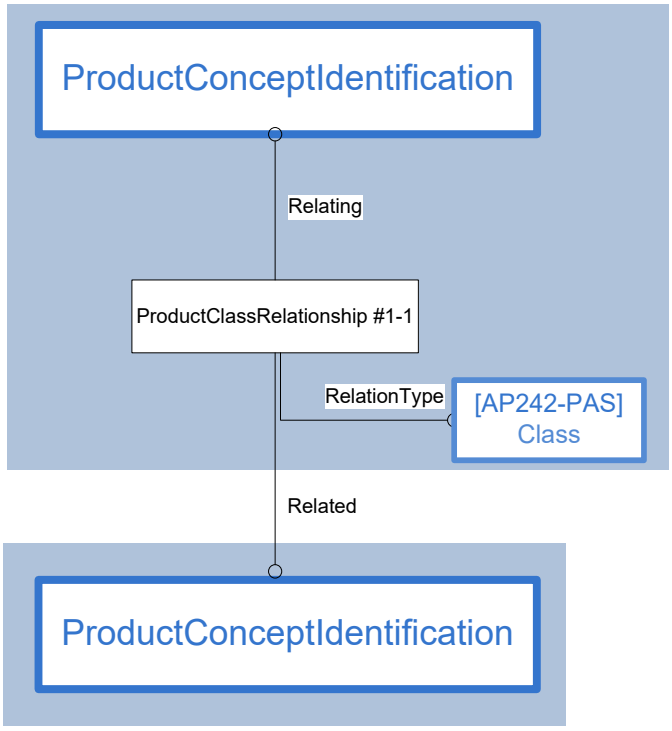


Figure 22: Instance Model ProductClassRelationship

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <ProductConcept uid="pc--19087" xsi:type="n0:ProductClass">
      <Id>
        <Identifier uid="pc-ent--id1" id="Smith.com" idRoleRef="rl--ii" id-
ContextRef="o--000000178"/>
      </Id>
      <Name>
        <CharacterString>Smith engineering</CharacterString>
      </Name>
      <LevelType>
        <ClassString>enterprise</ClassString>
      </LevelType>
      <ProductClassRelationship uid="pcr--1">
        <Related uidRef="pc--19088"/>
        <RelationType>
          <ClassString>hierarchy</ClassString>
        </RelationType>
      </ProductClassRelationship>
    </ProductConcept>
```

4.1.6 General Handling of Product Configurations

It is possible to have one ProductConcept with more than one ProductConfigurations.

4.1.7 Example Mapping an Item to AP242 Context

Some systems manage the context in the Assembly-Item itself and not in a separate context object. In most cases, the item is normally the top node of the product structure or an additional node on top of the product structure, but it may be any assembly node.

Most of the Effectivities along the product structure have to refer to an EndItem. For example mandatory for SerialEffectivities or ConfitionalEffectivities and optional for DateEffectivities.

Potentially, multiple levels within the product structure can be used as UsedItem, to tell that at each level, the 100% BoM can be computed.

The following example demonstrates the mapping of a 3Dexperience item to the AP242 ProductClass and the mapping to a Teamcenter EndItem for the use case “supplier exchange” (in this example the EndItem could be another Part or the top assembly Part used as a context).

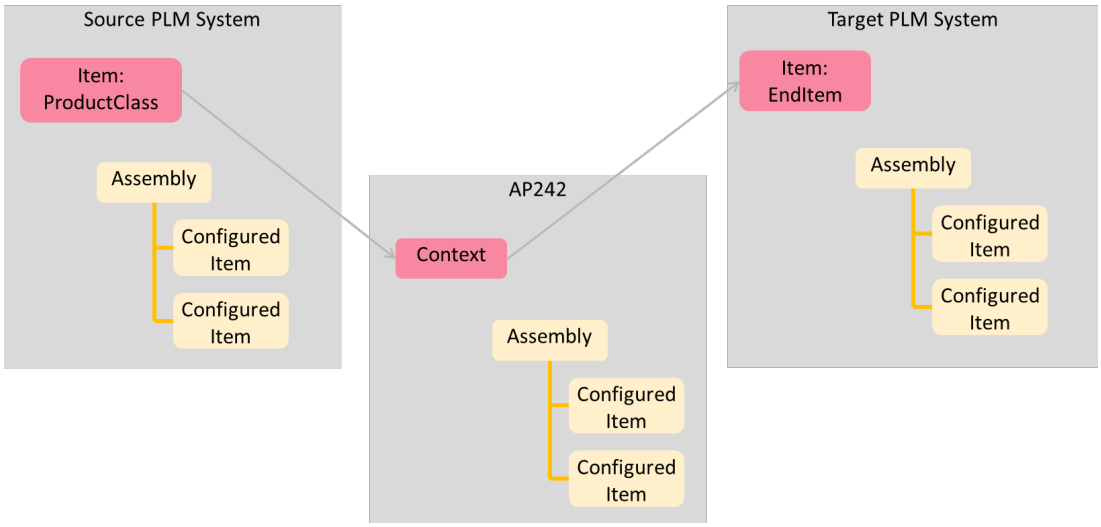


Figure 23: Context Mapping for Supplier Exchange

Since EndItems are not supported by many PDM Systems and usually, they are defined at one and only one assembly level within the product structure (just like a ProductClass), a dedicated mapping for EndItems is currently out of scope.

4.2 Configuration Composition Management

Configuration composition management is concerned with the specification of the different product configurations that exist for a given product concept, and the association with product data that is necessary to build those configurations. This includes the identification of the actual constituents that are to be included in a planned unit of production of a product configuration.

Effectivity is a key concept to control the valid use of product data. The AP242 Schema supports the association of effectivity information to different types of product data. Three different effectivity concepts are available:

- Configuration effectivity (serial, lot, time interval), describing the planned use of part versions (so-called revision effectivities) or part occurrences (i.e., occurrences of parts as sub-assemblies or component parts in some higher-level assembly, so-called occurrence effectivities) in the context of a Configuration Item defined for a ProductConcept.
- A more generic effectivity (DatedEffectivity), describing the validity period of part versions or part occurrences in which the associated product data may be used independent from any additional context (e.g., date, lifecycle or organization related) that further restricts the applicability of that effectivity.
- A conditional effectivity, describing the use of part versions or part occurrences based on specification expressions.

4.2.1 Common supertype “Effectivity”

Supertype of all effectivity kinds, it allows attachment of effectivity information to

- part versions (so-called revision effectivities)
- or to occurrences (so-called occurrence effectivities)

of component parts in the context of a particular ProductConfiguration Item or ProductConcept. This enables specification of the valid use of a part version or part occurrence in the context of a lot, serial number range, time period and/or specification (or many of them) of a particular product configuration. This controls the constituent parts that are planned to be used for manufacturing end items of a particular product configuration within a dated time period, for a certain lot or serial number range of the end items or according to specification expressions.

A configured assembly effectivity is an identification of the valid use of an aspect of product data tracked by date, event, serial number, lot size or specifications (see section 4.1.4.5).

Effectivity is supertype of:

- DatedEffectivity (for Date Effectivities and Milestone Effectivities)
- LotEffectivity
- SerialEffectivity
- TimeIntervalEffectivity
- ConditionalEffectivity

Effectivities are related by:

- EffectivityAssignment
- Condition

ENTITY Effectivity	**Attribute Type**
ClassifiedAs	OPTIONAL SET[1:?] of Classification
ConcernedOrganizations	OPTIONAL SET[1:?] of Organization
ConfigurationType	OPTIONAL ClassSelect
Description	OPTIONAL DescriptorSelect
EffectivityContext	OPTIONAL EffectivityContext
Id	OPTIONAL IdentifierSelect
InheritanceType	OPTIONAL ClassSelect
RetentionPeriod	OPTIONAL SET[1:?] OF RetentionPeriod
SameAs	OPTIONAL SET[1:?] of ProxySelect
VersionId	OPTIONAL Id
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
ConditionAssignment	OPTIONAL SET[1:?] of ConditionAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityRelationship	OPTIONAL SET[1:?] of EffectivityRelationship
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
OrganizationOrPersonInOrganizationAssign-ment	OPTIONAL SET[1:?] of OrganizationOrPersonInOr-ganizationAssignment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRelationship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 13: "Effectivity" Attributes

NOTE — The assignment of effectivities is often done during the approval process, i.e, when releasing product data for the next stage of the development process, it gets effectivity information assigned to define when and in which context these product data may be used.

Attribute recommendations:

- **Id:** The id attribute specifies the identifier that distinguishes the effectivity. Use “Identifier” template (see [242-PAS]). The value of this attribute need not be specified.
- **VersionId:** the identification or set of identifications of a specific version of the Effectivity. The value of this attribute need not be specified. Use IdentifierString type or "Identifier" template (see [242-PAS]).
- **EffectivityRelationship:** to relate to another Effectivity. Use the “EffectivityRelationship” template, see 4.2.9.
- **EffectivityContext:** the ProductClass for which the effectivity is valid (see “ProductConceptIdentification” template) Depending on the PDM system, and depending on the kind of Effectivity, the value of this attribute need or need not or shall not be specified.
- **ConfigurationType**: the valid usage of an Effectivity object that is applied to the business object as configured Element. The value of this attribute need not be specified. Use

ClassString if the value is recommended within this document, otherwise use “Class” template (see [242-PAS]). According to the ISO AP242 Specification, where applicable, the following values shall be used:

  - 'design' (currently out of scope): the object referenced as configured Element has to be designed and verified before it can actually be used in a given context. This context is specified by the SpecificationConditionAssignment and SpecificationAssignment objects referenced as the resolved Configuration. This type of Effectivity is applicable for AlternativeSolution or BreakdownElement objects
  - 'usage': the object referenced as the configured Element is controlled by an Effectivity. The SpecificationConditionAssignment and SpecificationAssignment objects specify the usage cases and are referenced as the resolved Configuration. This type of Effectivity is applicable for AlternativeSolution, Occurrence or PhysicalBreakdown objects

- Examples:
  - The presence of an optional third rear axle of a truck is controlled by specifications on the level of PhysicalBreakdown.
  - A 'sunroof' has to be provided, i.e., designed for a class of vehicles that has an optional 'sunroof' as part of its specification.

- **InheritanceType**: specifies whether or not an inheritance scheme for the configuration information in a hierarchical structure is applied to the business object referenced as the configuredElement. The levels within such a hierarchy are defined through BreakdownElementRealization objects or the attribute baseElement of AlternativeSolution or NextAssemblyOccurrenceUsage/NextAssemblyViewUsage objects. The value of this attribute need not be specified. Use ClassString if the value is recommended within this document, otherwise use “Class” template (see [242-PAS]). According to the ISO AP242 Specification, where applicable, the following values shall be used:
  - 'exception (currently out of scope)': No inheritance scheme is applicable and all required configuration information shall be attached locally at the business object. The value indicates that the configuration information may be inconsistent to the structural levels above it or that it is, on purpose, contradictory to it. Such a condition implies that an inheritance scheme shall not continue beyond this point in the product structure tree.
  - 'inherited': A scheme for inheritance of configuration information applies. The complete configuration information shall be collected from the different levels in the structure by evaluation of results. The results shall be evaluated using the logical AND to combine configuration information starting at the referenced configuredElement (i.e. along the lower levels of the structure) and using the logical OR to combine alternatives (i.e. for different nodes on the same level of the product structure, involving the same Part). In addition, this evaluation shall consider related effectivity information. 'inherited' only applies for objects for which the same value of Effectivity.ConfigurationType is defined. Nevertheless, any inherited configuration information of a higher level shall be consistent, i.e., it shall not repeat the inherited configuration information (this is already assumed if local configuration information is specified), and the inherited configuration information shall be included in the locally defined one.
    Example:
    level 1: navigation system

level 2: none (inherits ‘navigation system’)
level 2: ‘touch screen’ (inherits ‘navigation system’ and combined via ‘AND’ with ‘touch screen’).
It is a common understanding that the effectivity defined at level 1 have to be fulfilled before the effectivity defined at level 2 get evaluated. In other words: the upper-level wins.

- Examples:
  - (Currently out of scope) A situation where the inheritanceType 'exception' is applicable is a technical solution released for one particular customer without support in higher structure levels.
  - The following figure shows how inheritance is applied along the tree of a alternative solutions.

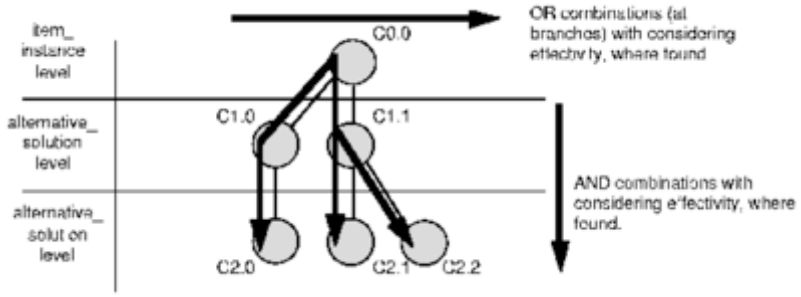


Figure 24: Inheritance along a tree of alternative solutions

The complete configuration information of a Occurrence can be obtained by adding any such information to AlternativeSolution objects which are linked through BreakdownElementRealization objects and baseElement attributes respectively. Whenever more than one higher level instance is present, the current information available is branched in as many branches as instances are present. For example, the total configuration information for the Occurrence given in the figure above could be expressed as follows: (C0.0 AND C1.0 AND C2.0) OR (C0.0 AND C1.1 AND C2.1) OR (C0.0 AND C1.1 AND C2.2).

- 'local': No inheritance scheme is applicable and all required configuration information shall be attached locally at the business object. Nevertheless, any potentially inherited configuration information of a higher level shall be consistent, i.e., it shall be included in the locally defined configuration information.
  Example:
  level 1: navigation system
  level 2: navigation system
  level 3: navigation system AND touch screen

- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

Even if defined as a BaseRootObject, it is recommended to not instantiate the effectivity instance as standalone instance, i.e., an effectivity should always be assigned to some product data by

using EffectivityAssignment. Furthermore, it is always recommended to instantiate an effectivity as one of its subtypes, unless:

- the EffectivityAssignment shall express that the assigned object/structure is configured and shall apply all the time (so-called null effectivity, semi-column or empty code rule)
- or the definition of the effectivity is based on some other effectivity using EffectivityRelationship.

Null effectivities may have an EffectivityContext. For example if the same product structure is shared by multiple ProductClasses: for ProductClass A, a SerialEffectivity is defined and for ProductClass B, a null effectivity is defined. SerialEffectivity.EffectivityContext would refer 'A' and (null) Effectivity.EffectivityContext would refer'B'.

An Identifier need not to be specified for a particular effectivity, except for shared effectivities (ie. that are referenced multiple times within a product structure). If used, the id value should be unique in conjunction with the product data the effectivity is assigned to, and if present, the context restricting the applicability of the effectivity to a certain usage.

If the source system does not support a Context of kind EffectivityContextSelect (Activity, Contract, Organization, PersonInOrganization, ProductClass, ProductConfiguration, Project), it is recommended to create an EffectivityAssignment with a Role "context" and to attach it to a PartVersion (possibly the top node of the configured product structure) representing the context (see template "EffectivityAssignment" 4.2.8.4).

If multiple versions of an Effectivity are needed, since Effectivity.Id and Effectivity.VersionId are in the same object, all versionless attributes of Effectivity (Id, EffectivityContext, …) shall be redundantly mapped to all versions, including the multiple identifiers.

'usage' and 'inherited' are the implicit default values for ConfigurationType and InheritanceType if they are not set in Effectivity.

Postprocessor Recommendations:

- If the target system requires an EffectivityContext (like in 3DExperience), but none is provided (if optional for example like in Teamcenter for DatedEffectivities or if none is supported for example like in Aras), the postprocessor has to decide which (general) context to use or to return an error.
- Some PDM systems do not support ConditionalEffectivities that combine simple effectivities (date, lot, serial, …) with specifications. If not supported an no simple splitting in multiple effectivities is possible, an error shall be returned.
- Some PDM systems do not support ConditionalEffectivities that combine Effectivities having different EffectivityContext (ProductClass, …). If not supported an no simple splitting in multiple effectivities is possible, an error shall be returned.

The behavior of the sender PDM system regarding inheritance of effectivities may not be supported by the receiving PDM system:

- If the receiving system supports inherited effectivities and the InheritanceType is set to 'local', it shall be checked if the effectivities may be imported (possibly redundantly) on each level of the product structure or if all redundancies shall be removed first.
- If the receiving system does not support configuration on multiple levels of the product structure:
    - If the exchanged data contains effectivities on multiple levels:
        - If the InheritanceType is set to 'inherited': for those product structure nodes where the effectivity is expected, the inherited effectivities shall be

retrieved ‘bottom up’ and all combined together with the local effectivity with ‘AND’

- If the InheritanceType is set to ‘local’, only those effectivities defined on the product structure nodes where the effectivity is expected, shall be mapped

In both cases, no effectivity information shall get lost, otherwise an error shall be returned.

- If the exchanged data contains effectivities on one level, but the receiving system does not support an Effectivty on the given PartVersion or PartOccurrence, an error shall be returned.

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<Effectivity uid="SE__600009" xsi:type="n0:Effectivity"/>
```

4.2.2 Template “DatedEffectivity”

The DatedEffectivity is a subtype of Effectivity that applies onwards from a point in time, or between two points in time which define the start and end of the DatedEffectivity. These points in time may be specified by:

- Date
- Date and time
- Event occurrence (currently recommended for the mapping of Milestone effectivities)

The Instance Model: AP242 Domain Model XML entities and attributes

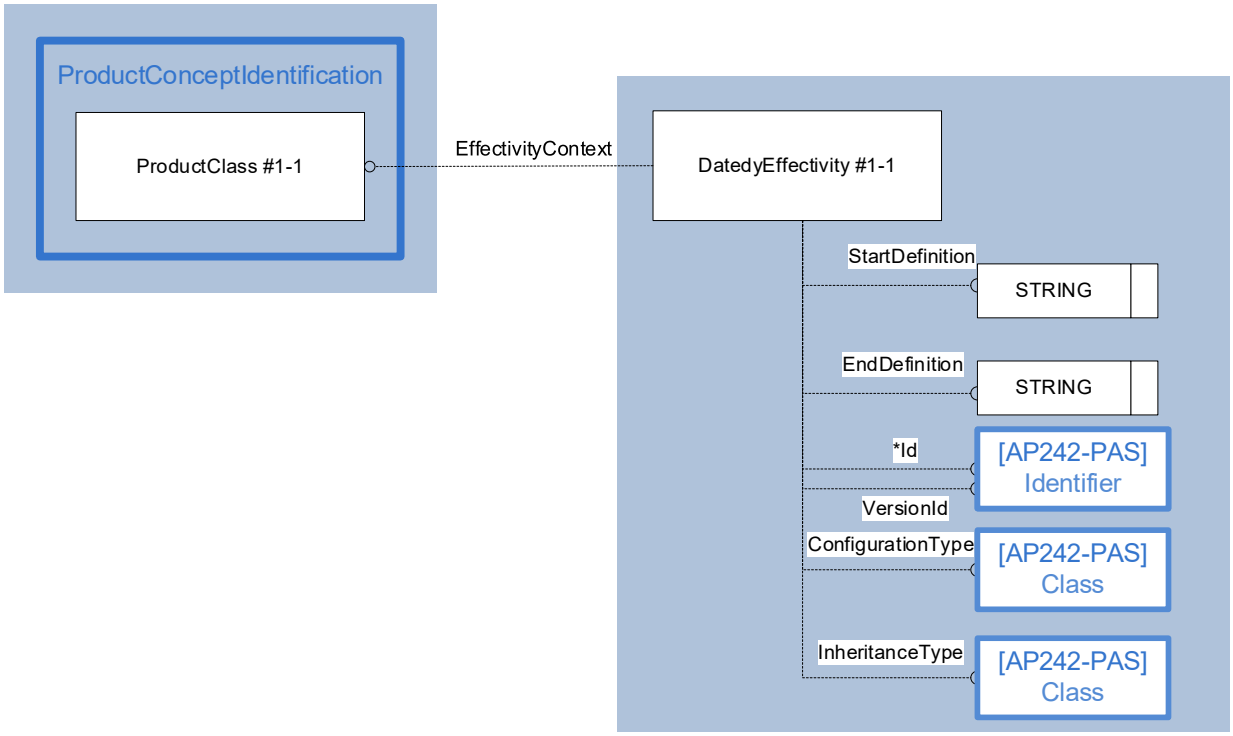


Figure 25: Instance Model DatedEffectivity

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<Effectivity uid="dveff-20161231" xsi:type="n0:DatedEffectivity">
  <VersionId id="A.1"/>
  <EndDefinition>
    <DateTimeString>2016-12-10T12:14:23</DateTimeString>
  </EndDefinition>
  <StartDefinition>
    <DateTimeString>2016-12-31T23:11:05</DateTimeString>
  </StartDefinition>
</Effectivity>
```

4.2.2.1 ENTITY DatedEffectivity

Additional attributes, derived from entity effectivity, will not be repeated in this table.

Attribute Name	Attribute Type
EndDefinition	OPTIONAL EventOrDateSelect
StartDefinition	OPTIONAL EventOrDateSelect

Table 14: "DatedEffectivity" Attributes

Attribute recommendations:

- **EndDefinition:** The EndDefinition specifies the date and time (or Milestone) when the effectivity of the associated product data ends. It defines the upper bound of the interval of applicability. The value of this attribute need not be specified. If a value for this attribute is not defined, the interval of applicability has no upper limit. Use “DateTime” template for DateTimeString without DateTimeAssignment (see [242-PAS]).
- **StartDefinition:** The StartDefinition specifies the Date and Time (or Milestone) when the effectivity of the associated product data begins to apply. It defines the lower bound of the interval of applicability. The value of this attribute need not be specified. If a value for this attribute is not defined, the interval of applicability has no lower limit. Use “DateTime” template for DateTimeString without DateTimeAssignment (see [242-PAS]) in case of a DatedEffectivity. Use “Event” in case of a Milestone Effectivity (see chapter 4.2.2.2).

Preprocessor Recommendations:

- The StartDefinition shall be prior to the EndDefinition.
- If a PDM system does not support an empty from-date, the dummy value 1970-01-01T00:00:00 may be used instead.
- Most PDM systems do not define an EffectivityContext on DatedEffectivities
- Using Events, the EndDefinition is meant ‘exclusive’, so the next effectivity to “Event1 to Event 4” is for example “Event 4 to Event7”.
- Ditto for dates: 1.7.2024 to 2.7.2024 applies from “1.7.2024-00:00:00.000 to 1.7.2024-23:59:59.999.
- As a consequence, StartDefinition and EndDefinition shall not have the same value.

Postprocessor Recommendations:

- The from-date value 1970-01-01T00:00:00 shall be interpreted as if from-date was empty.
- If the receiving PDM system does not expect a value in EffectivityContext, the context shall be ignored, or a warning shall be returned.

- If the receiving PDM system expects a value in EffectivityContext, but it is missing, the postprocessor has to decide which (general) context to use or to return an error.

4.2.2.2 ENTITY Event

An Event is the fact of the existence of a state at some point in time. It is recommended to use Event to map Milestone Effectivities

Attribute Name	Attribute Type
ActualStartDate	OPTIONAL DateTime
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
EventType	ClassSelect
Id	OPTIONAL IdentifierSelect
Offset	OPTIONAL xsd:duration
PlannedStartDate	OPTIONAL DateTime
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
EventRelationship	OPTIONAL SET[1:?] of EventRelationship
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganizationAssignment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRelationship

Table 15: "Event" Attributes

Attribute recommendations:

- **EventType:** specifies the kind of event. Use ClassString if the value is recommended within this document, otherwise use “Class” template (see [242-PAS]). Where applicable, the following values shall be used:
    - 'milestone': the event is a milestone.
- **Id:** The id attribute specifies the identifier that distinguishes the event. Use “Identifier” template (see [242-PAS]). Although the value of this attribute need not be specified, it is recommended to set it in case of a Milestone.
- **ActualStartDate:** the actualStartDate specifies the date when the Event actually started. The value of this attribute need not be specified. Use “DateTime” template for DateTimeString without DateTimeAssignment (see [242-PAS]).
- **PlannedStartDate:** the DateTimeString when the Event is or was planned to start. The value of the attribute need not be specified. Use “DateTime” template for DateTimeString without DateTimeAssignment (see [242-PAS]).
- **OrganizationOrPersonInOrganizationAssignment**: an organization or person in organization with a specific relation to the EVent according to the OrganizationOrPersonInOrganizationAssignment.role attribute. The value of this attribute need not be specified. Use "PersonInOrganization" template (see 4.6.19). See [242-PSMS] for more details.

- **Offset:** specifies the amount of time before or after the defined Event that shall be used to calculate the actual point in time. The value of this attribute need not be specified.
  The possibilities to define values for duration are explained in xsd:duration.
  A complex example for a negative duration:
  -P3Y6M4DT12H30M17.123456789012345678905
- **ActivityAssignment**: the Activities associated to the Event. The value of this attribute need not be specified. Use "Activity" template (see [242-CM] and [242-PSMS] for details).
- **SuppliedObjectRelationship**: to assign (optionally) one or multiple Events on supplier's sites (see [242-PSMS] for more details). Use the "SuppliedObjectRelationship" template in [242-PAS] section 4.6.22.
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

There are no specific Preprocessor recommendations

Postprocessor Recommendations:

If not supported by the target system, Milestone Effectivities shall be ignored, or a warning shall be returned.

The Instance Model: AP242 Domain Model XML entities and attributes

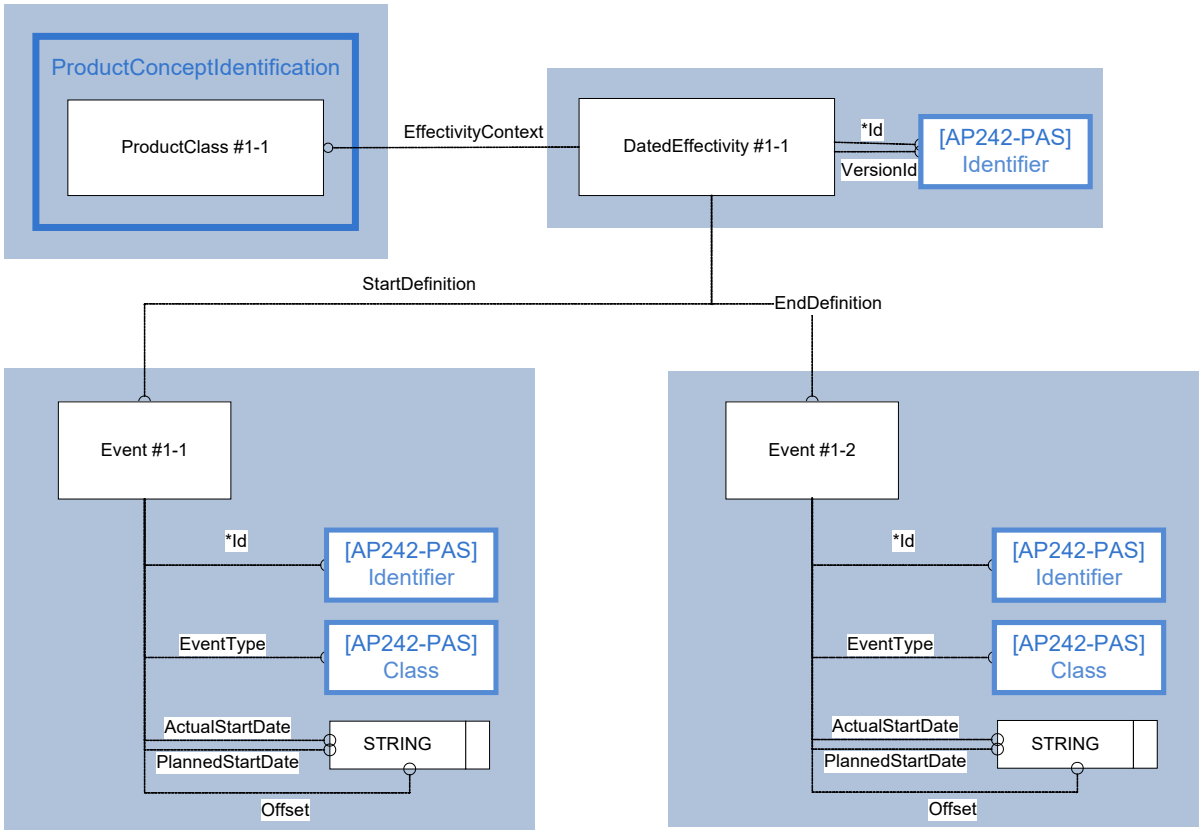


Figure 26: Instance Model Milestone Effectivity

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <Effectivity uid="dveff-M1" xsi:type="n0:DatedEffectivity">
      <VersionId id="A.1"/>
      <EndDefinition>
        <Event uidRef="ev--2"/>
      </EndDefinition>
      <StartDefinition>
        <Event uidRef="ev--1"/>
      </StartDefinition>
    </Effectivity>
    <Event uid="ev--1">
      <EventType>
        <ClassString>milestone</ClassString>
      </EventType>
      <Id>
        <Identifier uid="ev--1-id1" id="M1" idRoleRef="rl--ii" idContex-
tRef="o--000000178"/>
      </Id>
    </Event>
    <Event uid="ev--2">
      <EventType>
        <ClassString>milestone</ClassString>
      </EventType>
      <Id>
        <Identifier uid="ev--2-id1" id="M2" idRoleRef="rl--ii" idContex-
tRef="o--000000178"/>
      </Id>
    </Event>
```

4.2.3 Template “LotEffectivity”

The LotEffectivity is a subtype of Effectivity with an additional attribute LotSize. It defines the domain of applicability as a given batch of end items.

The Instance Model: AP242 Domain Model XML entities and attributes

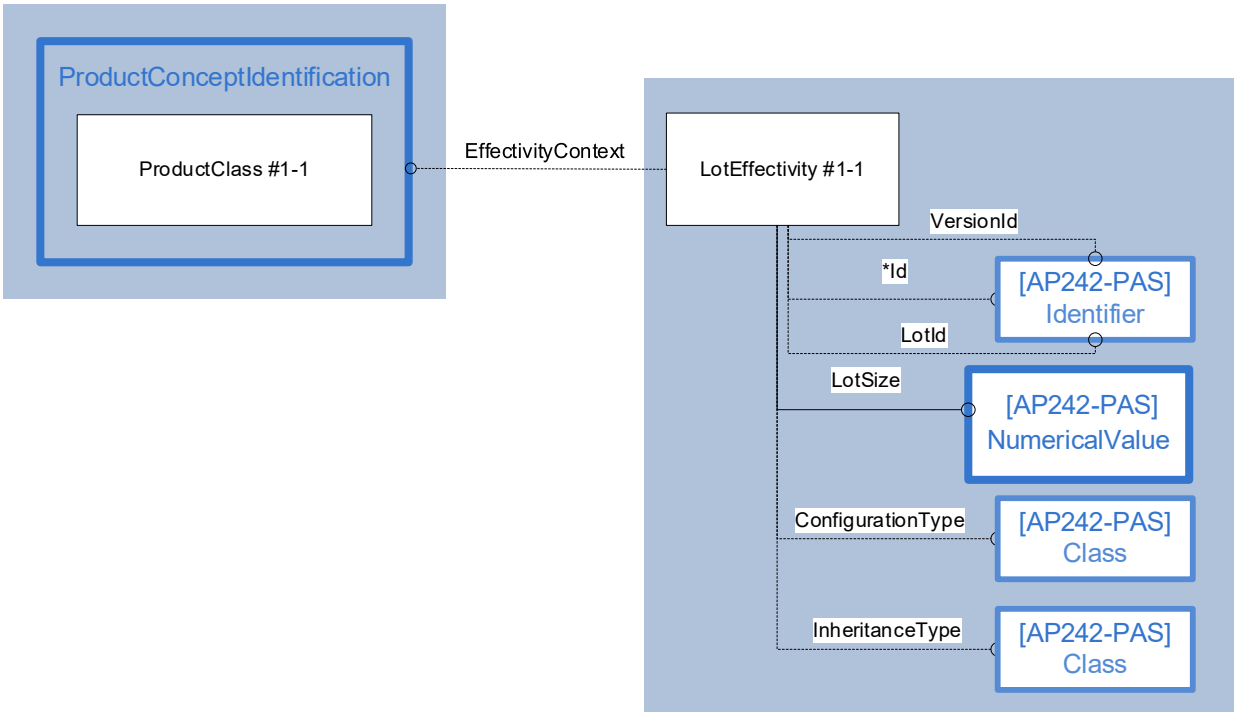


Figure 27: Instance Model LotEffectivity

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <Effectivity uid="loteff-x-1" xsi:type="n0:LotEffectivity">
      <EffectivityContext uidRef="pc--19088"/>
      <VersionId id="A.1"/>
      <LotId>
        <IdentifierString>123456</IdentifierString>
      </LotId>
      <LotSize uid="nv--l2" xsi:type="n0:NumericalValue">
        <Definition>
          <PropertyDefinitionString>lot size property</PropertyDefini-
tionString>
        </Definition>
        <Unit uidRef="i--1231"/>
        <ValueComponent>35</ValueComponent>
      </LotSize>
    </Effectivity>
```

4.2.3.1 ENTITY LotEffectivity

Additional attributes, derived from entity effectivity will not be repeated in this table

Attribute Name	Attribute Type
LotId	OPTIONAL IdentifierSelect
LotSize	ValueWithUnit

Table 16: "LotEffectivity" Attributes

Attribute recommendations:

- **LotId**: the identification of the batch of items, which is different from Effectivity identification. The value of this attribute need not be specified. Use “Identifier” template (see [242-PAS]).
- **LotSize:** The LotSize attribute specifies the size of the batch of end items for the manufacturing of which the effectivity applies. The lotsize definition (LotSize.Definition.PropertyDefinitionString) should be set to “lot size property”. A range of lots is not supported. Use NumericalValue template (see [242-PAS]).

Preprocessor Recommendations:

- It is not recommended to use other attributes of the super object to add effectivity information like StartDefinition or EndDefinition together with LotEffectivity, to avoid rule out each other.
- Detailed informations about the lot itself are not supported. The lot will only be represented by the defined Id.
- A range of lots is not supported in AP242. If a range is supported by the exporting system, the preprocessor has to multiply the LotEffectivity corresponding to the range.
- Most PDM systems require to define an EffectivityContext on LotEffectivities

Postprocessor Recommendations:

- If not supported by the target system, LotEffectivities shall be ignored, or a warning shall be returned.
- If the receiving PDM system does not expect a value in EffectivityContext, the context shall be ignored, or a warning shall be returned.
- If the receiving PDM system expects a value in EffectivityContext, but it is missing, the postprocessor has to decide which (general) context to use or to return an error.

4.2.4 Template “SerialEffectivity”

A SerialEffectivity is a type of Effectivity for which the domain of applicability is defined as a possibly open-ended interval of serial numbers.

The Instance Model: AP242 Domain Model XML entities and attributes

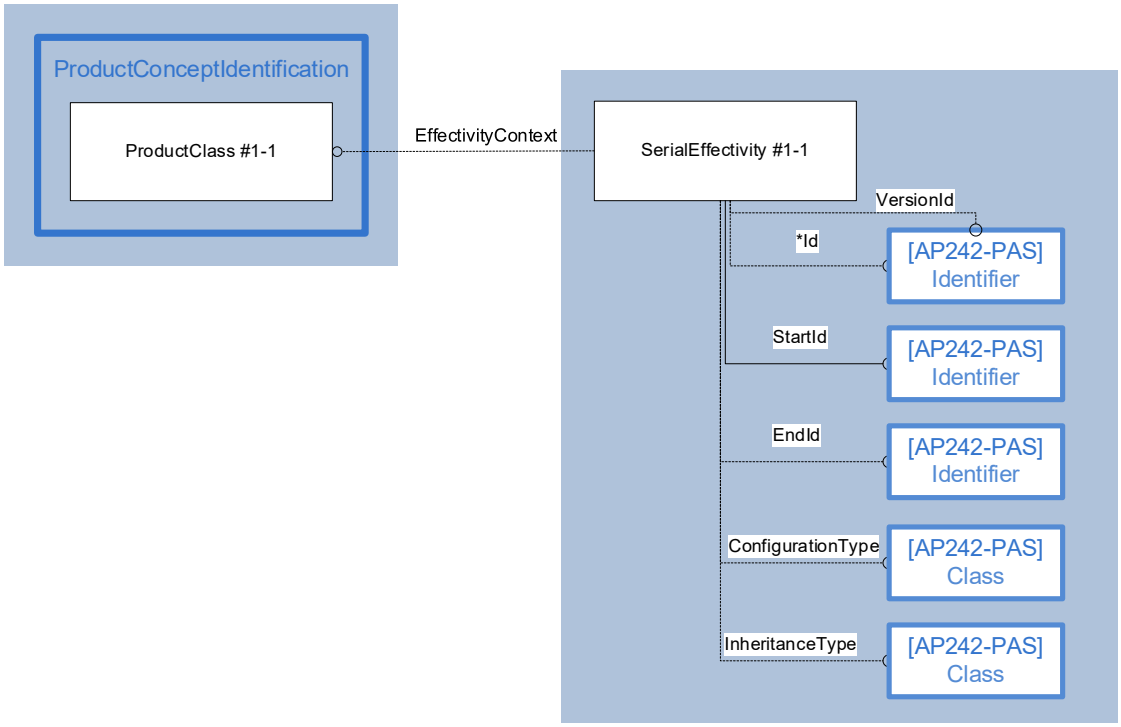


Figure 28: Instance Model SerialEffectivity

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<Effectivity uid="sneff-33-2" xsi:type="n0:SerialEffectivity">
  <EffectivityContext uidRef="pc--19088"/>
  <VersionId id="A.1"/>
  <EndId>
    <IdentifierString>3</IdentifierString>
  </EndId>
  <StartId>
    <IdentifierString>3</IdentifierString>
  </StartId>
</Effectivity>
```

4.2.4.1 ENTITY SerialEffectivity

Additional attributes, derived from entity effectivity will not be repeated in this table

Attribute Name	Attribute Type
EndId	OPTIONAL IdentifierSelect
StartId	IdentifierSelect

Table 17: “SerialEffectivity” Attributes

Attribute recommendations:

- **EndId:** The EndId attribute identifies the last valid serial number for which the effectivity applies (may be equal to the StartId). The value of this attribute need not be specified. If a

value for this attribute is not defined, the interval of applicability has no serial number defined as upper bound, i.e., the effectivity is expected to apply to all serial numbers greater than the StartId. Use IdentifierString type or “Identifier” template (see [242-PAS]).

- **StartId:** The StartId attribute identifies the first valid serial number for which the effectivity applies. Use IdentifierString type or “Identifier” template (see [242-PAS]).

Preprocessor Recommendations:

- The StartId shall be prior to the EndId.
- Most PDM systems do require an EffectivityContext on SerialEffectivities
- Most PDM systems require to define an EffectivityContext on SerialEffectivities
- The EndId ist meant ‘inclusive’, so the next effectivity to “1 to 4” is for example “5 to 7”. This allows to express “1 to 1”.

Postprocessor Recommendations:

- If the receiving PDM system does not expect a value in EffectivityContext, the context shall be ignored, or a warning shall be returned.
- If the receiving PDM system expects a value in EffectivityContext, but it is missing, the postprocessor has to decide which (general) context to use or to return an error.

4.2.5 Template “TimeIntervalEffectivity”

A TimeIntervalEffectivity is a type of Effectivity for which the domain of applicability is defined as duration.

The Instance Model: AP242 Domain Model XML entities and attributes

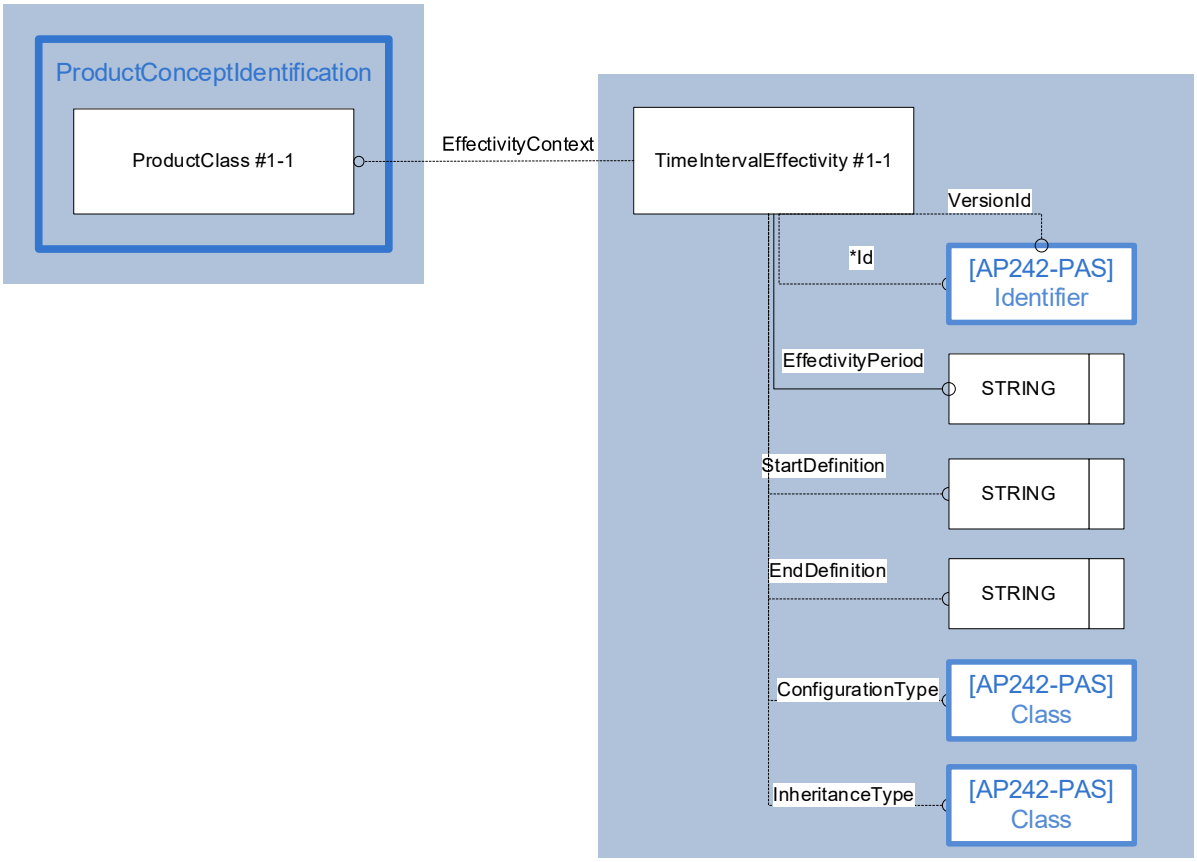


Figure 29: Instance Model TimeIntervallEffectivity

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <Effectivity uid="tieff-20160301P3D" xsi:type="n0:TimeIntervalEffectiv-
ity">
      <EffectivityContext uidRef="pc--19088"/>
      <VersionId id="A.1"/>
      <StartDefinition>
        <DateTimeString>2016-12-31T00:00:00</DateTimeString>
      </StartDefinition>
      <EffectivityPeriod>P3D</EffectivityPeriod>
    </Effectivity>
```

4.2.5.1 ENTITY TimeIntervallEffectivity

Additional attributes, derived from entity effectivity, will not be repeated in this table.

Attribute Name	Attribute Type
EffectivityPeriod	xsd:duration

Table 18: "TimeIntervallEffectivity" Attributes

Attribute recommendations:

- **EffectivityPeriod:** The EffectivityPeriod attribute specifies the time intervall that defines the domain of validity. The possibilities to define values for duration are explained in xsd:duration.
  A complex example for a negative duration:
  -P3Y6M4DT12H30M17.123456789012345678905
- **EndDefinition:** Use a negative duration together with EndDefinition (in this case, StartDefinition shall not be set). Use "DateTime" template for DateTimeString without DateTimeAssignment (see [242-PAS]).
- **StartDefinition:** Use a positive duration together with StartDefinition (in this case, EndDefinition shall not be set). Use "DateTime" template for DateTimeString without DateTimeAssignment (see [242-PAS]).

Preprocessor Recommendations:

The usage of TimeIntervalEffectivity is not recommended since not supported by most of the PDM systems.

Postprocessor Recommendations:

TimeIntervalEffectivities shall be ignored, or a warning shall be returned.

4.2.6 Template "VersionBranchEffectivity"

A VersionBranchEffectivity is a type of Effectivity for which the domain of applicability is defined as a possibly open-ended interval of versions. A 'start' value shall always be specified as 'Root', an 'end' value may be specified as 'Leaf'.

Currently, it is recommended that the referenced versioned object is a ProductClass.

The Instance Model: AP242 Domain Model XML entities and attributes

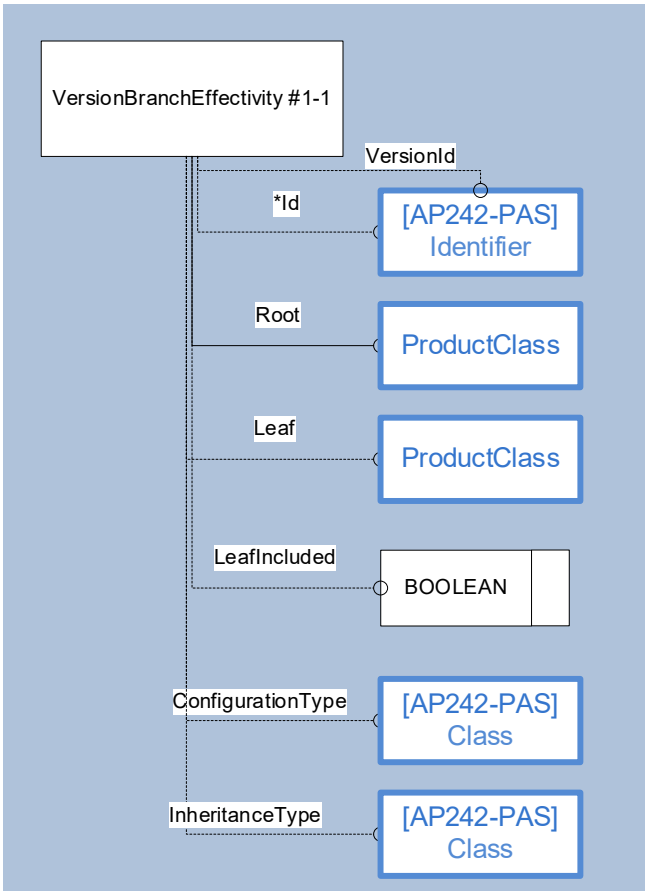


Figure 30: Instance Model VersionBranchEffectivity

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

T.B.S.

4.2.6.1 ENTITY VersionBranchEffectivity

Additional attributes, derived from entity effectivity will not be repeated in this table

Attribute Name	Attribute Type
LeafIncluded	OPTIONAL BOOLEAN
Leaf	OPTIONAL VersionedObjectSelect
Root	VersionedObjectSelect

Table 19: "VersionBranchEffectivity" Attributes

Attribute recommendations:

- **LeafIncluded**: indicates, whether the Leaf is included in the effectivity range. The value of this attribute need not be specified.
- **Leaf:** the last valid version. The value of this attribute need not be specified. If the value for this attribute is not specified, the interval of applicability has no upper bound. See section 4.1.3.1 for ProductClass template.
- **Root:** the first valid version. See section 4.1.3.1 for ProductClass template.

Preprocessor Recommendations:

- The Root shall be prior to the Leaf along the Relationship that builds the branches.
- Root and Leaf shall reference two objects having the same type (currently only ProductClass).
- Root and Leaf shall reference two versions of the same object, i.e. ProductClass.Id shall be the same.
- ProductClass.Scope shall relate to the top-level assembly node under wich the VersionBranchEffectivities referring to the ProductClass as Root/Leaf are defined.
- It is not recommended to define an EffectivityContext on VersionBranchEffectivities, since the Root/Leaf (ProductClass) already defines the context.
- Normally, the Leaf is meant 'exclusive', so the next effectivity to "Version1 to Version4" is for example "Version4 to Version7" (LeafIncluded shall be set to FALSE), but in some cases, "inclusive" is necessary, for example to express "Version1 to Version1" since a Version may have multiple successors (in different branches), so using an excluded Leaf could be rather complex. In this case, LeafIncluded shall be set to TRUE; Root and Leaf may be equal.

Postprocessor Recommendations:

A value in EffectivityContext shall be ignored, or a warning shall be returned.

If **LeafIncluded** is not set, the **Leaf** is excluded.

4.2.7 Template "ConditionalEffectivity"

The ConditionalEffectivity is a specialization of Effectivity for which the domain of applicability is defined by an explicit condition.

The Instance Model: AP242 Domain Model XML entities and attributes

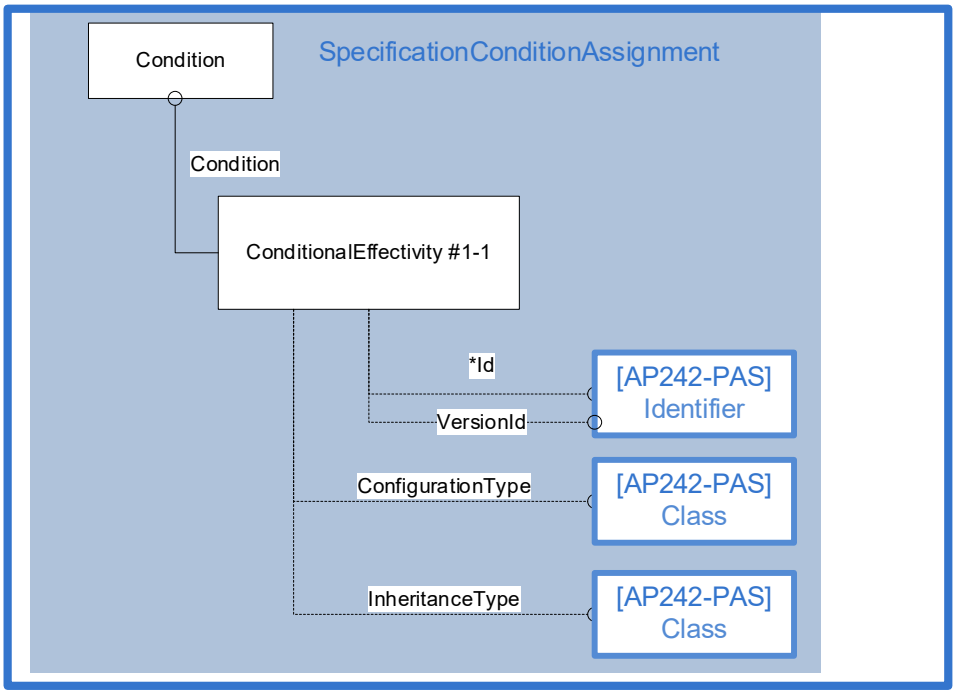


Figure 31: Instance Model ConditionalEffectivity

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<Effectivity uid="condeff-Spec001" xsi:type="n0:ConditionalEffectivity">
  <VersionId id="A.1"/>
  <Condition uidRef="spece--19133"/>
</Effectivity>
```

4.2.7.1 ENTITY ConditionalEffectivity

Additional attributes, derived from entity effectivity, will not be repeated in this table.

Attribute Name	Attribute Type
Condition	Condition

Table 20: "ConditionalEffectivity" Attributes

Attribute recommendations:

- **Condition:** Reference to a condition. See “SpecificationConditionAssignment” template in chapter 4.3.4.

Preprocessor Recommendations:

The use of OneOfCondition is not recommended, since not supported by most of the PDM systems.

It is not recommended to use other attributes of the supertype object to add effectivity information like StartDefinition or EndDefinition together with ConditionalEffectivity, to avoid rule out each other.

It should not be used to parametrize a product configuration by EffectivityAssignment. In this case, rather use ProductConfiguration.definingSpecifications.

ConditionalEffectivities are always indirectly associated to a ProductClass (via SpecificationConditionAssignment), so the use of ConditionalEffectivity.EffectivityContext would be redundant and is therefore not recommended.

Postprocessor Recommendations:

A value in EffectivityContext shall be ignored, or a warning shall be returned.

4.2.7.2 ENTITY ConditionalConfiguration

A ConditionalConfiguration adds usage information to a ConditionalEffectivity so that it may be evaluated in combination with further ConditionalEffectivities.

The Instance Model: AP242 Domain Model XML entities and attributes

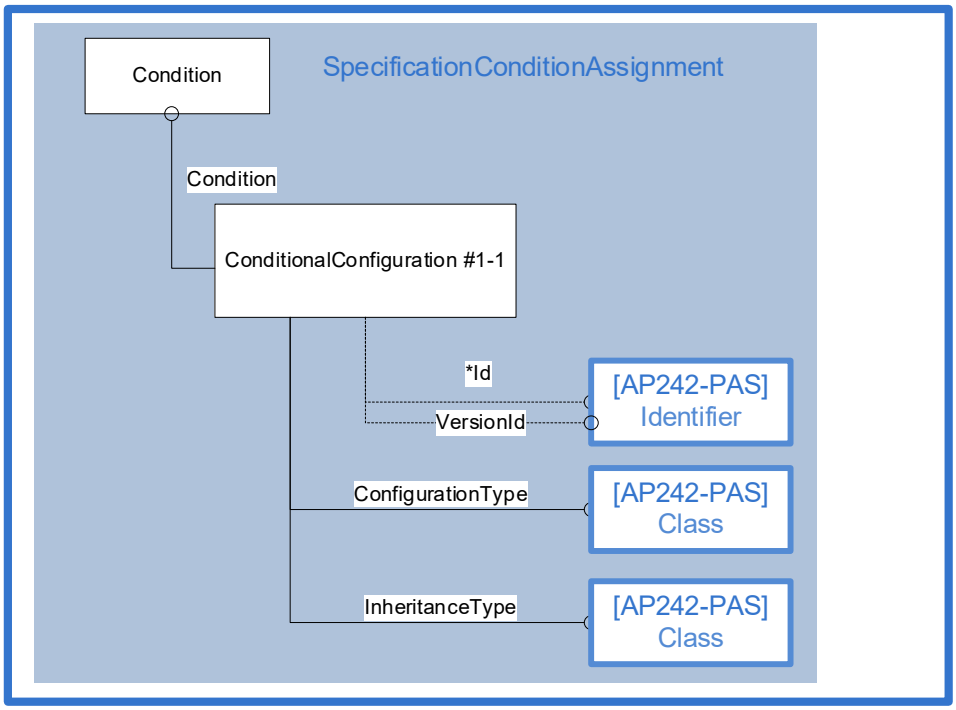


Figure 32: Instance Model ConditionalConfiguration

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <Effectivity uid="condconf-Spec001" xsi:type="n0:ConditionalConfigura-
tion">
      <VersionId id="A.1"/>
      <Condition uidRef="spece--19133"/>
      <ConfigurationType>
        <ClassString>usage</ClassString>
      </ConfigurationType>
      <InheritanceType>
        <ClassString>local</ClassString>
      </InheritanceType>
    </Effectivity>
```

Additional attributes, derived from entity ConditionalEffectivity, will not be repeated in this table.

Attribute Name	Attribute Type
ConfigurationType	ClassSelect
InheritanceType	ClassSelect
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssignment
SecurityClassificationAssignment	OPTIONAL SET[1:?] of SecurityClassificationAssignment

Table 21: "ConditionalConfiguration" Attributes

Attribute recommendations:

- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

The same Preprocessor Recommendations than Effectivity are valid (see “Effectivity template” 4.2.1).

Postprocessor Recommendations:

The same Postprocessor Recommendations than Effectivity are valid (see “Effectivity template” 4.2.1).

4.2.8 Template “EffectivityAssignment”

An EffectivityAssignment is an object that associates effectivity with the product or activity data whose effectivity is controlled by the associated Effectivity.

It is recommended to use EffectivityAssignments for following Entities:

- PartVersion (so-called revision effectivities)
- NextAssemblyOccurrenceUsage (so-called occurrence effectivities)
- ProductConfiguration (to define the configuration)

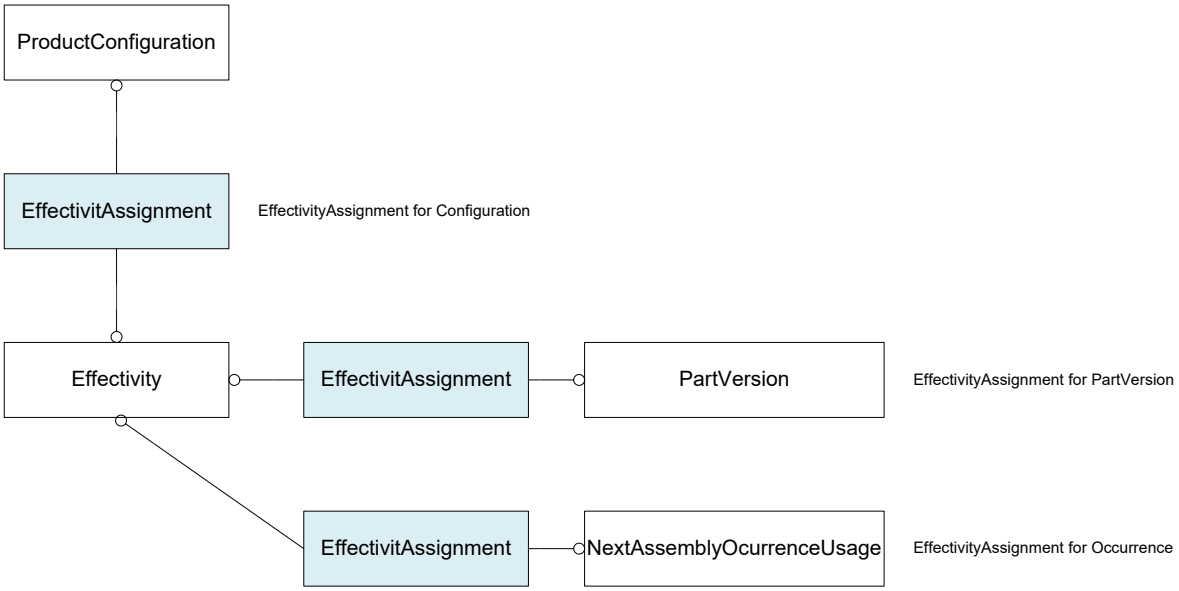


Figure 33: Simplified Example for supported EffectivityAssignments

Rules:

- In case of serial/timeinterval/date range effectivities, the PartVersion or NextAssemblyOccurrenceUsage is valid for the configuration if the effectivity assigned to the configuration (single value of range) is within the given range.
- In case of ConditionalEffectivities, the boolsche expression shall be evaluated for all of:
    - the serial/timeinterval/date/lot effectivities assigned to the Configuration

the Specifications assigned to the Configuration via its attribute definingSpecifications

- Special cases:
  - The PartVersion is valid for the Configuration, if both have an assignment to the same effectivity
  - The NextAssemblyOccurrenceUsage is valid for the Configuration, if both have an assignment to the same effectivity

4.2.8.1 EffectivityAssignment for PartVersion

If the effectivity of a part is always the same for all usages of the part, Effectivities may be assigned to the PartVersions of the part structure definition, to define the validity for assembly structure links.

The Instance Model: AP242 Domain Model XML entities and attributes

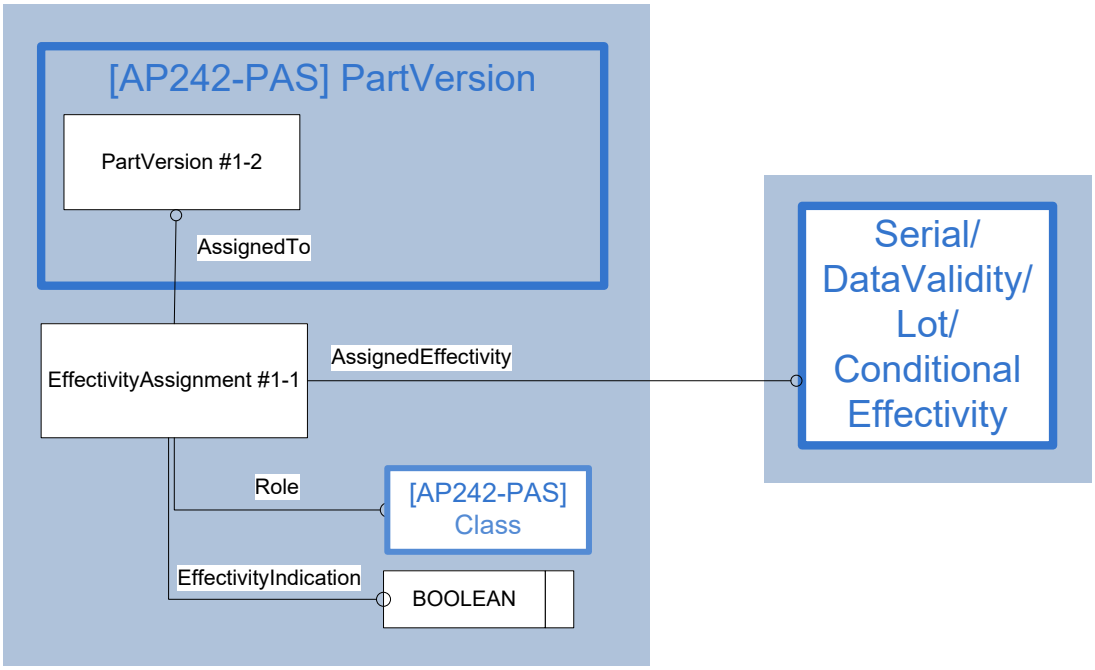


Figure 34: Instance Model EffectivityAssignment for PartVersions

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <Part uid="p--0000000017D374A0">
…
      <Versions>
        <PartVersion uid="pv--0000000017D374A0--id1">
…
          <EffectivityAssignment uid="ea--19215v">
            <AssignedEffectivity uidRef="condeff-Spec001"/>
            <EffectivityIndication>true</EffectivityIndication>
            <Role>
              <ClassString>actual</ClassString>
            </Role>
          </EffectivityAssignment>
…
        </PartVersion>
      </Versions>
    </Part>
```

4.2.8.2 EffectivityAssignment for Assembly Links

If the effectivity of a part may be different, depending on each usage of the part, it is recommended to assign Effectivities from the Entity NextAssemblyOcurrenceUsage of the part structure definition, to define the validity for assembly structure links.

The Instance Model: AP242 Domain Model XML entities and attributes

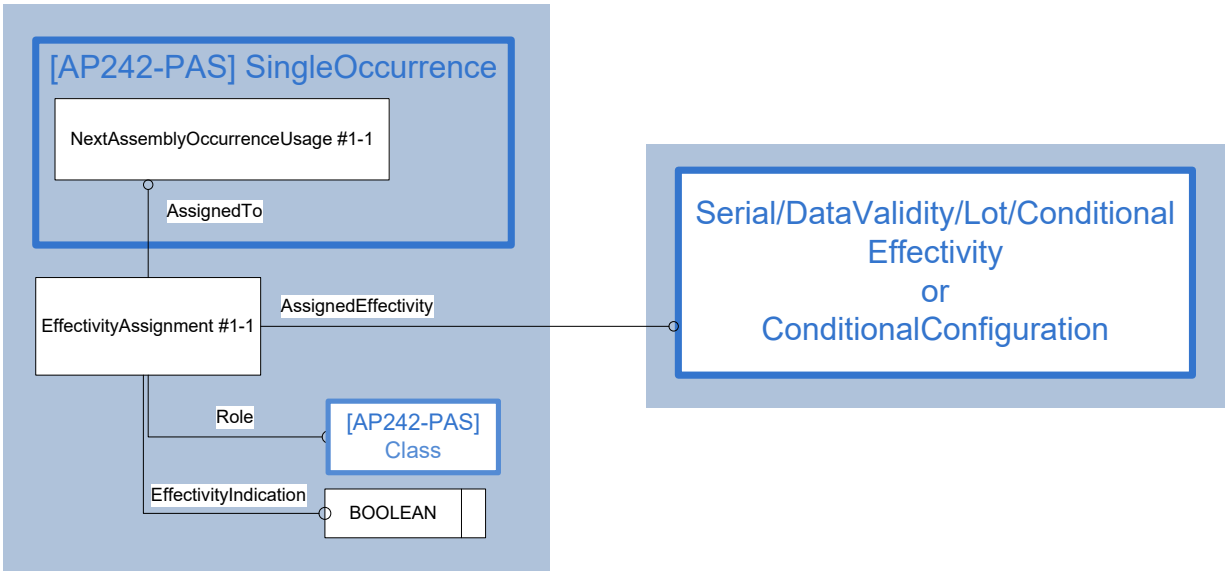


Figure 35: Instance Model EffectivityAssignment for assembly links

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <Part uid="p--0000000017D36260">
      <Id>
…
      <Versions>
        <PartVersion uid="pv--0000000017D36260--id8">
…
          <Views>
            <PartView xsi:type="n0:AssemblyDefinition" uid="pvv--
0000000017D36260--id8">
…
              <ViewOccurrenceRelationship uid="pvvid--000000001EA37D20--20"
xsi:type="n0:NextAssemblyOccurrenceUsage">
                <Related uidRef="pi--000000001EA37D20--20"/>
                <RelationType>
                  <ClassString>next assembly occurrence</ClassString>
                </RelationType>
                <EffectivityAssignment uid="ea--19218">
                  <AssignedEffectivity uidRef="dveff-20161231"/>
                  <EffectivityIndication>true</EffectivityIndication>
                  <Role>
                    <ClassString>actual</ClassString>
                  </Role>
                </EffectivityAssignment>
                <Placement>
                  <CartesianTransformation uid="cto--000000001EA37D20--20">
```

```
                    <RotationMatrix>1.000000 0.000000 0.000000 0.000000
1.000000 0.000000 0.000000 0.000000 1.000000</RotationMatrix>
                    <TranslationVector>-30.000000 0.000000 0.000000</Transla-
tionVector>
                  </CartesianTransformation>
                </Placement>
              </ViewOccurrenceRelationship>
…
              <AssemblyType>
                <ClassString>design assembly</ClassString>
              </AssemblyType>
            </PartView>
          </Views>
…
        </PartVersion>
      </Versions>
    </Part>
```

4.2.8.3 EffectivityAssignment for Configurations

It is recommended to assign effectivities to the configurations to define the lot/serial/date validity of the links dependent on the selected configuration. If more than one lot/serial/date apply, they shall be either applied through several EffectivityAssignments (implicitly combined with 'AND') or into an AndCondition.

The specifications that shall apply to the configuration shall be mapped in DefiningSpecifications, alls combined implicitly with AND and again combined implicitly with AND with the EffectivityAssignment(s).

An Issue TCSC410303-839 has beed defined in Bugzilla to avoid the use of EffectivityAssignment and add Serial/Lot/Dated/TimeIntervalEffectivities to the definition of a ProductConfiguration.

The Instance Model: AP242 Domain Model XML entities and attributes

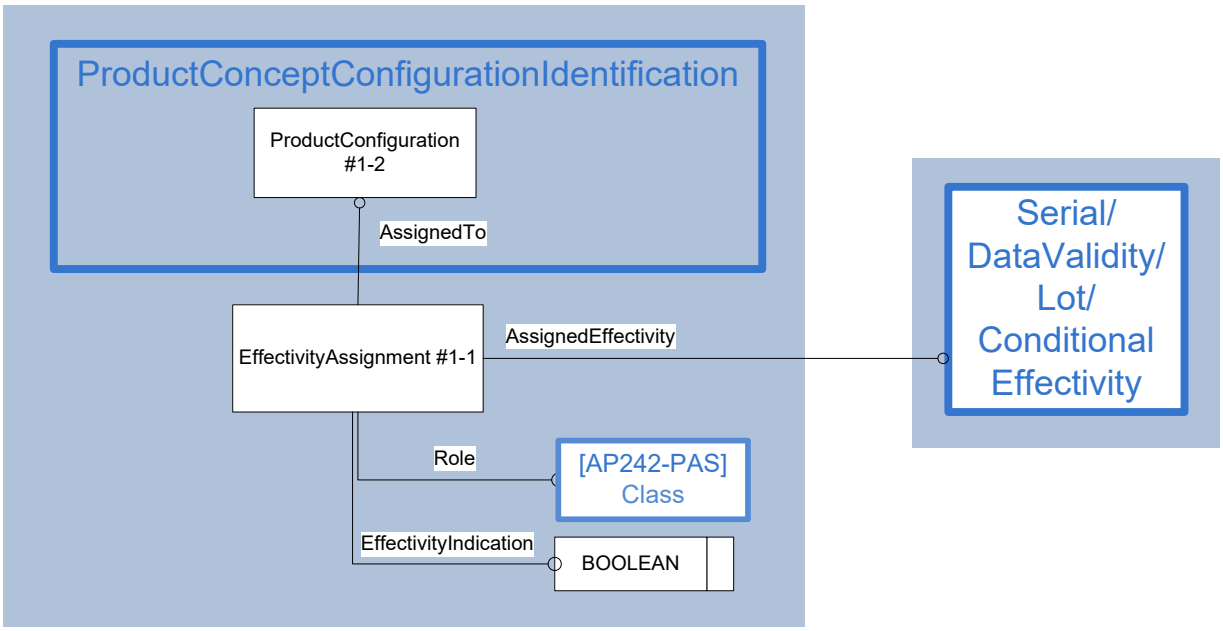


Figure 36: Instance Model EffectivityAssignment for configurations

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<ProductConcept uid="pc--19088" xsi:type="n0:ProductClass">
…
    <ProductConfiguration uid="pconf--19089">
      <Description>
        <CharacterString>as1 short description</CharacterString>
      </Description>
      <Id>
        <Identifier uid="pc-as1short--id1" id="as1 short" idRoleRef="rl--ii"
idContextRef="o--000000178"/>
      </Id>
      <MemberOf uidRef="pc--19088"/>
      <Name>
        <CharacterString>as1 short name</CharacterString>
      </Name>
      <VersionId id="A.1"/>
      <EffectivityAssignment uid="ea--19225">
        <AssignedEffectivity uidRef="sneff-11"/>
        <EffectivityIndication>true</EffectivityIndication>
        <Role>
          <ClassString>configuration</ClassString>
        </Role>
      </EffectivityAssignment>
      <EffectivityAssignment uid="ea—19225-2">
        <AssignedEffectivity uidRef="sneff-11-2"/>
        <EffectivityIndication>true</EffectivityIndication>
        <Role>
          <ClassString>configuration</ClassString>
        </Role>
      </EffectivityAssignment>
    </ProductConfiguration>
…
</ProductConcept>
```

4.2.8.4 ENTITY EffectivityAssignment

Attribute Name	Attribute Type
AssignedEffectivity	Effectivity
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
EffectivityIndication	BOOLEAN
Id	OPTIONAL Identifier
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
Role	ClassSelect
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
AssignmentObjectRelationship	OPTIONAL SET[1:?] of AssignmentObjectRelationship
ConditionAssignment	OPTIONAL SET[1:?] of ConditionAssignment

DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssignment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganizationAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 22: "EffectivityAssignment" Attributes

Attribute recommendations:

- **AssignedEffectivity**: the instance of Effectivity that is assigned. Use “Effectivity” template in chapters 4.2.2 to 4.2.6.
- **Description:** the text or the set of texts that provide further information about the EffectivityAssignment. The value of this attribute need not be specified. Use "Description" template (see [242-PAS]).
- **EffectivityIndication**: the indication whether the assignedEffectivity defines a period of effectivity (value 'TRUE') or a period of ineffectivity (value 'FALSE') for the assignedTo objects. In the first case, the use of assignedTo objects is or was valid during the considered period. In the second case, the use of assignedTo objects is or was not valid during the considered period. The use of ‘TRUE’ is recommended.
- **Role:** the meaning of the assignment. Use ClassString if the value is recommended within this document, otherwise use “Class” template (see [242-PAS]). According to the ISO AP242 Specification, where applicable, the following values shall be used:
    - 'actual': The actual period during which the assignedEffectivity lasted: EffectivityAssignments or PartVersions (see chapter 4.2.8.1) and Assembly Links (see chapter 4.2.8.2)
    - ‘configuration’: The effectivity (mostly one single serial number, date or a list of specifications) used to define a product variant: EffectivityAssignments for ProductConfiguration (see chapter 4.2.8.3)
    - 'planned' (currently out of scope): The period associated with the assignedEffectivity defines a planned period of time during which the assignedTo objects are or were supposed to be effective;
    - 'required' (currently out of scope): The AssignedTo objects shall be kept effective for this period.
    - ‘context’: This value mean’s that the AssignedTo object represent the context of the AssignedEffectivity (for systems not supporting Product Context elements). The AssignedTo objet shall be a PartVersion.

- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

- If more than one configuration is used within the product concept, the configurations have to be assigned with different effectivities to define the filters for the configuration
- It is recommended to assign at most:
    - One LotEffectivity xor
        - One SerialEffectivity with StartNumber=EndNumber and/or
        - One DatedEffectivity with StartDefinition=EndDefinition
    - And/or one or many Specifications (via the attribute definingSpecifications)

  to a ProductConfiguration, since this is the way most of the PDM systems support ProductConfigurations
- Some PDM systems support the definition of serial/date ranges for ProductConfigurations: doing so, a >100% product structure gets selected (for overview purpose).
- If a PDM system (like VPM V4) interpretes a PartVersion or PartOccurrence without any effectivity as never valid, a DatedEffectivity having StartDefinition='1970-01-01T00:00:00' and no EndDefinition shall be mapped during export, together wird EffectivityIndication=FALSE.
- It is not recommended to reuse an Effectivity for several PartVersions, for several Assembly Links or for several ProductConfigurations
- If the source system does not support EffectivityAssignment.Role, it should be set to 'actual'.
- Usually, the BOM consists either of EffectivityAssignment for Occurrences xor EffectivityAssignment for PartVersions

Postprocessor Recommendations:

- If a so-called null effectivity (see section 4.2.1), or no effectivities are assigned to the assembly structure, these assembly structure links have to be considered as always applying.
- If only one configuration is defined with a null effectivity or no effectivity is assigned to the configuration, all links with all effectivities within the data structure have to be considered as applying.
- If the receiving system does not support the EffectivityAssignment for PartVersions, it has to be ignored or a warning shall be returned.
- If the receiving system does not support the EffectivityAssignment for Occurrences, it has to be ignored or a warning shall be returned.
- It is not recommended to map EffectivityAssignment for PartVersions as EffectivityAssignment for Occurrences or vice versa.
- If the receiving system does not support effectivityIndication=FALSE, it shall negate the effectivity as if it came with effectivityIndication=TRUE and was embedded into a NonEqualsCondition effectivity.
- If the receiving system (like VPM V4) interpretes a PartVersion or PartOccurrence without any effectivity as never valid, DatedEffectivities having StartDefinition='1970-01-

01T00:00:00' and no EndDefinition together wird EffectivityIndication=FALSE shall be interpreted as an empty effectivity.

- If the receiving system does not support EffectivityAssignment.Role, it should be ignored, or a warning shall be returned.
- Since not recommended, reuse of an Effectivity for several PartVersions, for several Assembly Links or for several ProductConfigurations, has to be mapped to a distinct Effectivity in the target system.
- If multiple EffectivityAssignments apply to an object, they are implicitly combined with 'OR'.

4.2.9 Template "EffectivityRelationship"

This relationship enables to relate two Effectivities.

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<Effectivity uid="loteff-x-2" xsi:type="n0:LotEffectivity">
      <EffectivityContext uidRef="pc--19088"/>
      <VersionId id="A.1"/>
      <EffectivityRelationship uid="effr--1">
            <Related uidRef="loteff-x-2b"/>
            <RelationType>
            <ClassString>sequence</ClassString>
            </RelationType>
      </EffectivityRelationship>
…
</Effectivity>
<Effectivity uid="loteff-x-2b" xsi:type="n0:LotEffectivity">
      <EffectivityContext uidRef="pc--19088"/>
      <VersionId id="A.2"/>
…
</Effectivity>
```

Entity EffectivityRelationship attributes	Attribute type
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
Id	OPTIONAL IdentifierSelect
Related	Effectivity
RelationType	ClassSelect
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment

EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssignment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganiza-tionAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
RelationshipObjectRelationship	OPTIONAL SET[1:?] of RelationshipObjectRelationship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 23: "EffectivityRelationship" Attributes

Attribute recommendations

- ***RelationType***: the meaning of the relationship. Use ClassString type if one of the values below is used, otherwise use "Class" template (see [242-PAS]). According to the ISO AP242 Specification, where applicable, the following values shall be used:

RelationType	
'constraint'	The time period between the start and end definition of the related Effectivity shall be within the time period of the relating Effectivity;
'inheritance'	The related Effectivity shall not have neither 'start' nor 'end' definitions specified, but inherits the effectivity bounds from the relating Effectivity.

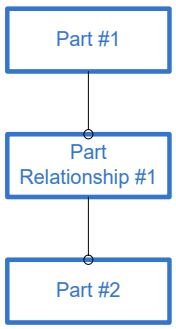
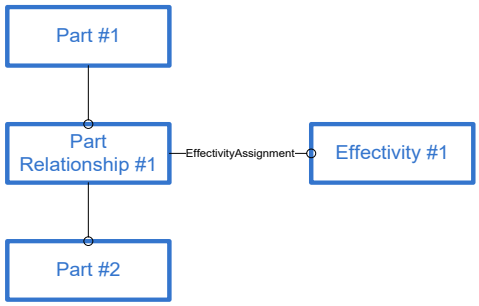
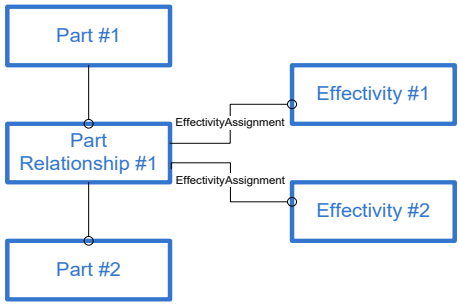
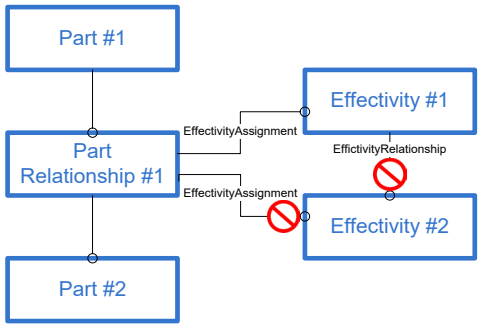
Additionally, the following value is recommended:

'sequence'	The business object defines a logical sequence where the related effectivity comes after the relating effectivity. In case the Id is equal and VersionId of both Effectivities is different, a version sequence is defined.

- ***Related***: the other object of **Effectivity** that is part of the relationship
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

4.2.10 General Handling of Effectivities

There are different possibilities to assign effectivity information to the product. This chapter will give an overview of the possibilities with the example to define the validity of an assembly structure link. The diagrams are not compliant to the instantiation diagrams defined in chapter 1.1.10 and are strongly reduced to the minimum information needed to show the dependencies between the units.

Simplified Diagrams	Recommended interpretation
	The assembly structure link or PartVersion is always valid. See special handling of PDM systems that assume the contrary. There is no effectivity assigned
	The assembly structure link or PartVersion is valid if the effectivity #1 is valid. The effectivity could be one of the introduced effectivities except for the ConditionalEffectivity
	The assembly structure link or PartVersion is valid if the effectivity #1 or effectivity #2 is valid. If more than one effectivity is assigned, it has to be handeld as an implicit **OR** operation between all of them.
	It is not recommended to use EffectivityAssignment and EffectivityRelationship together for assigning the additional effectivity information of Effectivit#2
Combined Effectivities	Combined Effectivities with ConditionalEffectivities are supported

4.3 Product Specification

4.3.1 Template “Specification”

A Specification shall exist only within the context of a ProductConceptIdentification (via SpecificationAssignment) to define its ‘availability’ within one (or several) ProductClass.

Additionaly, they are assigned to ProductConfigurations in which they influence if the Effectivities on PartVersion/Occurrence are valid or not for this ProductConfiguration

The specifications define the characteristics of the product and discriminate one product from other members of the same ProductClass.

A specification belongs to a SpecificationCategory that completes the semantics of the specification.

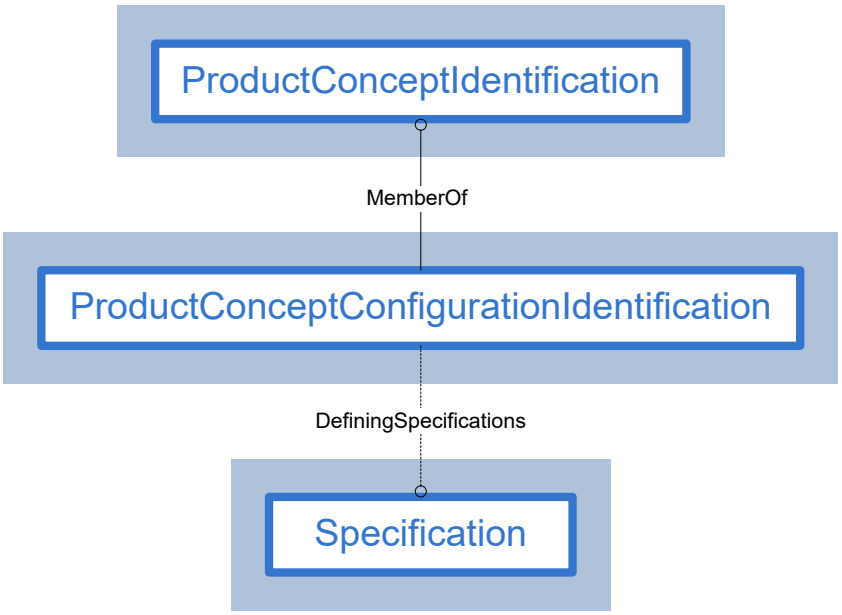


Figure 37: Specifications to define configurations

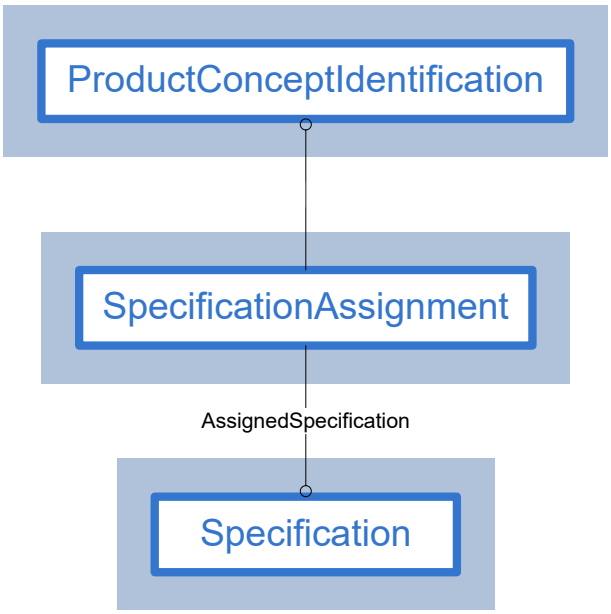


Figure 38: Specifications defined as available for configuration given ProductClass

The Instance Model: AP242 Domain Model XML entities and attributes

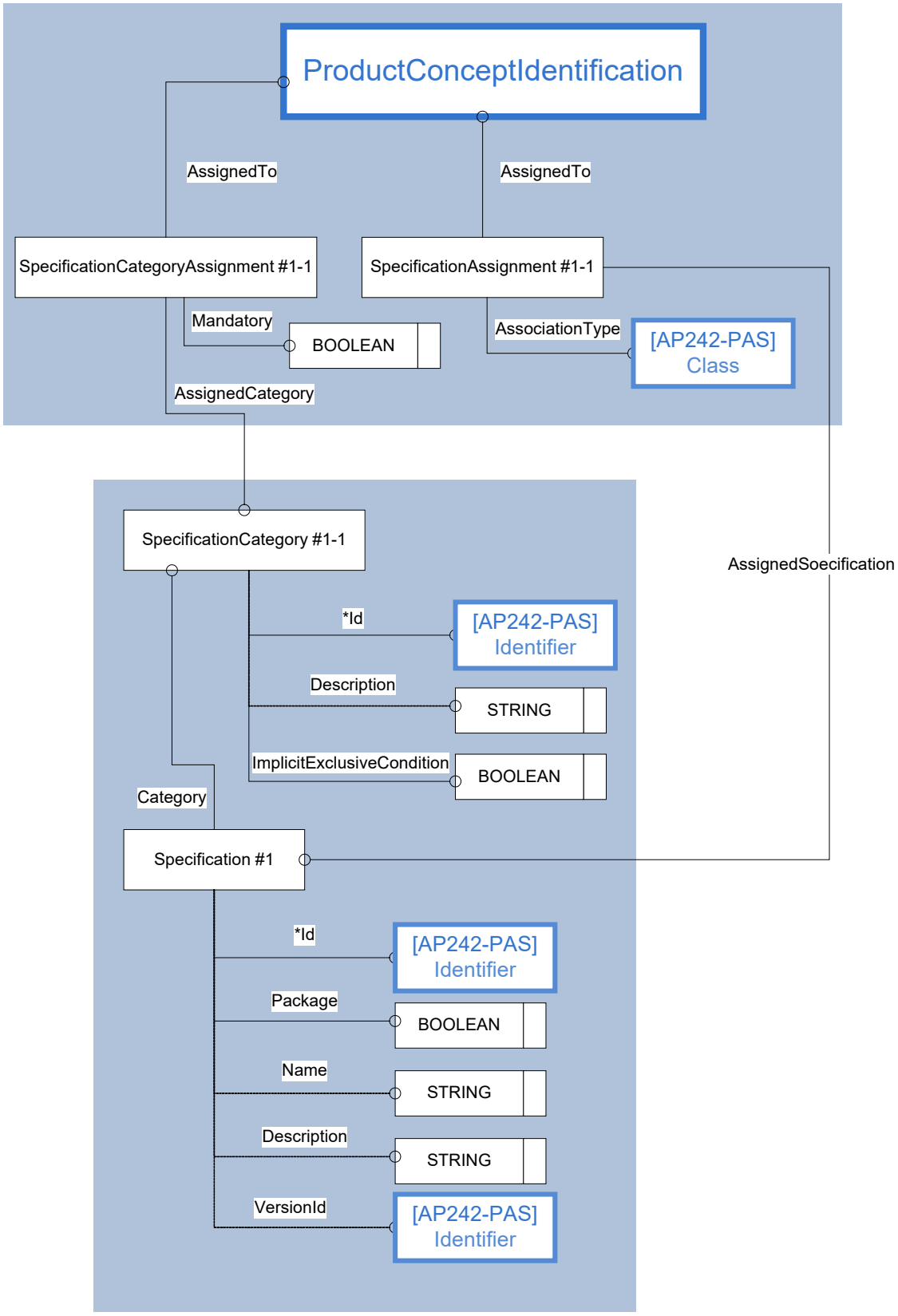


Figure 39: Instance Model Specification

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<ProductConcept uid="pc--19088" xsi:type="n0:ProductClass">
      <Description>
        <CharacterString>as class description</CharacterString>
      </Description>
      <Id>
        <Identifier uid="pc-asclass--id1" id="as class" idRoleRef="rl--ii"
idContextRef="o--000000178"/>
      </Id>
      <Name>
        <CharacterString>as class name</CharacterString>
      </Name>
      <LevelType>
        <ClassString>platform</ClassString>
      </LevelType>
      <VersionId id="A.1"/>
…
      <SpecificationAssignment uid="csa--19126">
        <AssignedSpecification uidRef="spec--19191"/>
        <AssociationType>
          <ClassString>availability</ClassString>
        </AssociationType>
      </SpecificationAssignment>
      <SpecificationAssignment uid="csa--19127">
        <AssignedSpecification uidRef="spec--19192"/>
        <AssociationType>
          <ClassString>availability</ClassString>
        </AssociationType>
      </SpecificationAssignment>
      <SpecificationCategoryAssignment uid="cca--19124">
        <AssignedCategory uidRef="specc--19123"/>
        <Mandatory>true</Mandatory>
      </SpecificationCategoryAssignment>
</ProductConcept>
<SpecificationCategory uid="specc--19123">
      <Description>
        <CharacterString>car body description</CharacterString>
      </Description>
      <Id>
        <Identifier uid="specc--carbody-id1" id="car body" idRoleRef="rl--ii"
idContextRef="o--000000178"/>
      </Id>
      <ImplicitExclusiveCondition>true</ImplicitExclusiveCondition>
      <Specification uid="spec--19191">
        <Description>
          <CharacterString>Spec001 description</CharacterString>
        </Description>
        <Id id="Spec001"/>
        <Name>
          <CharacterString>Spec001 name</CharacterString>
        </Name>
        <Package>false</Package>
        <VersionId id="A.1"/>
      </Specification>
      <Specification uid="spec--19192">
        <Description>
          <CharacterString>Spec002 description</CharacterString>
        </Description>
        <Id id="Spec002"/>
```

```
        <Name>
          <CharacterString>Spec002 name</CharacterString>
        </Name>
        <Package>false</Package>
        <VersionId id="A.1"/>
      </Specification>
</SpecificationCategory>
```

4.3.1.1 ENTITY Specification

Attribute Name	Attribute Type
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
Id	IdentifierSelect
Name	OPTIONAL DescriptorSelect
Package	BOOLEAN
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
VersionId	OPTIONAL IdentifierSelect
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssign-ment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUs-ageRightAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssign-ment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPer-sonInOrganiza-tionAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefini-tionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssign-ment
SecurityClassificationAssignment	OPTIONAL SET[1:?] of SecurityClassifica-tionAssignment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRela-tionship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssign-ment

Table 24: "Specification" Attributes

Attribute recommendations:

- **ClassifiedAs**: the classifications of the Specification. The value of this attribute need not be specified except for nested files (see [242-PAS] 9.3) and incremental exchange (see [242-PAS] 9.4). Use “Classification” template (see [242-PAS]).
- **Description:** the text by which the specification is described. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- **Id:** the identifier that distinguishes the specification. Use IdentifierString or “Identifier” template (see [242-PAS]).
- **Name:** the text by which the specification is known. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- **Package:** specifies whether this specification represents a package of specification objects or not. Such a specification combines those specification objects that shall be offered to the market as a set. In the case where package is 'true', there shall be exactly one SpecificationInclusion per ProductClass considered, that refers to this Specification by its attribute ifCondition. The Specification objects that are members of the package, shall be specified in the attribute includedSpecification of the SpecificationInclusion.
- **VersionId**: the identification or set of identifications of a particular version of the Specification. The value of this attribute need not be specified. Use IdentifierString type or "Identifier" template (see [242-PAS]).

- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

If the Specification.Id is unique independently from the SpecificationCategory, IdentifierString may apply in case only one Identifier shall be exchanged. If more than one Identifier is involved, or if two SpecificationCategories may contain Specifications having the same name, and these are different Specifications, then the template “Identifier” shall apply.

Additionally, if the Specification.Id is unique independently from the SpecificationCategory, idContextRef may reference an Organization, otherwise it shall reference the SpecificationCategory.Identifier.uid (as in “Identifier” template, see [242-PAS]).

Since SpecificationInclusion is not yet in scope of this document, at the moment it is recommended to set Package to ‘false’ and to mention all specifications in the effectivities (no specification will be implicitly included).

Postprocessor Recommendations:

Ignore at the moment the value of Package and any related SpecificationInclusion.

If multiple versions of a Specification are needed, since Specification.Id and Specification.VersionId are in the same object, all versionless attributes of Specification (Id, EffectivityContext, …) shall be redundantly mapped to all versions, including the multiple identifiers.

4.3.1.2 ENTITY SpecificationCategory

A SpecificationCategory is a definition of a set of specification objects serving the same purpose (for example all engines that may be alternatively chosen (one and only one) radio specifications that may be alternatively chosen (zero or one), or all confort specifications that may be ordered (zero to many).

Attribute Name	Attribute Type
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
Id	IdentifierSelect
ImplicitExclusiveCondition	BOOLEAN
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
Specification	OPTIONAL SET[1:?] of Specification
SpecificationCategoryHierarchy	OPTIONAL SET[1:?] of SpecificationCategoryHierarchy
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssignment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganiza-tionAssignment
SecurityClassificationAssignment	OPTIONAL SET[1:?] of SecurityClassificationAssignment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRelationship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 25: "SpecificationCategory" Attributes

Attribute recommendations:

- **ClassifiedAs**: the classifications of the SpecificationCategory. The value of this attribute need not be specified except for nested files (see [242-PAS] 9.3) and incremental exchange (see [242-PAS] 9.4). Use “Classification” template (see [242-PAS]).
- **Description:** the text by which the SpecificationCategory is described. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- **Id:** the identifier that distinguishes the SpecificationCategory. Use “Identifier” template (see [242-PAS]).
- **ImplicitExclusiveCondition:** specifies whether the specification objects within the SpecificationCategory are mutually exclusive for the production of one particular product. A value of 'true' indicates that the referenced objects are mutually exclusive for the production of the particular product.
- **Specification:** embeds the Specifications belonging to this category.
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

If the source system does not support ImplicitExclusiveCondition, it should be set to 'false'.

Postprocessor Recommendations:

If the target system does not support ImplicitExclusiveCondition, it should be ignored, or a warning shall be returned.

4.3.2 Template "SpecificationAssignment"

A SpecificationAssignment exists only within the context of one ProductConcept to define the meaning of a specification within a ProductClass.

Attribute Name	Attribute Type
AssignedSpecification	Specification
AssociationType	ClassSelect
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
AssignmentObjectRelationship	OPTIONAL SET[1:?] of AssignmentObjectRela-tionship
ConditionAssignment	OPTIONAL SET[1:?] of ConditionAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUsageRightAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssignment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganiza-tionAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 26: "SpecificationAssignment" Attributes

Attribute recommendations:

- **AssignedSpecification:** the Specification that is associated with the ProductClass (see Template “Specification”).
- **AssociationType:** the kind of availability of a particular specification in a ProductClass. Use ClassString if the value is recommended within this document, otherwise use “Class” template (see [242-PAS]). According to the ISO AP242 Specification, where applicable, the following values shall be used:
  - 'availability': The specification is a potential characteristic of any product belonging to a high level ProductClass. It is not specified if this is an option or a standard. This builds the option pool as defined in chapter 4.4.

    EXAMPLE: 'front wheel drive' or 'four-wheel drive' are available for a product class 'Voyager‘

  - 'identification' (currently out of scope): The specification is a characteristic that enables to distinguish the associated ProductClass from other ProductClass objects. This is a kind of 'non replaceable standard'. This value is not applicable for a top level node in a hierarchy of ProductClass objects. This identification is part of the identification of all sub classes of this ProductClass;

    EXAMPLE: The car body type 'limousine' is identifying product class 'S40'.

  - 'non replaceable standard' (currently out of scope): The specification is a characteristic of all products belonging to the ProductClass;

    EXAMPLE: 'South East Asia climate zone' is a 'non-replaceable standard' for a car that is produced for use in that particular geographic area.

  - 'option' (currently out of scope): The specification is a characteristic of a product if explicitly chosen. The Specification replaces another Specification of the same SpecificationCategory if the replaced specification is associated with the ProductClass as 'replaceable standard';

    EXAMPLE: 'sunroof' or 'radio with CD' are common options proposed to the customer.

  - 'part usage' (currently out of scope): The specification is a characteristic for the usage of the components of an AlternativeSolution, the usage of an Occurrence, or for the application of a ProcessPlan or a ProcessOperationOccurrence in the products of the associated ProductClass;

  - 'replaceable standard' (currently out of scope): The specification is a default characteristic of the products belonging to the ProductClass as long as no other Specification of the same SpecificationCategory is chosen.
- **Description:** the text by which the SpecificationAssignment is described. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

Until further use cases are in scope of this document, use SpecificationAssignment only with AssociationType=’availability’.

Postprocessor Recommendations:

Ignore at the moment any SpecificationAssociations not having AssociationType= ‘availability’.

4.3.3 Template “SpecificationCategoryAssignment”

A SpecificationCategoryAssignment exists only within the context of one ProductConcept to define the meaning of a SpecificationCategory within a ProductClass.

Attribute Name	Attribute Type
AssignedCategory	Specification
ClassifiedAs	OPTIONAL SET[1:?] of Classification
Description	OPTIONAL DescriptorSelect
Mandatory	BOOLEAN
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
AssignmentObjectRelationship	OPTIONAL SET[1:?] of AssignmentObjectRela-tion-ship
ConditionAssignment	OPTIONAL SET[1:?] of ConditionAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssign-ment
OrganizationOrPersonInOrganizationAssign-ment	OPTIONAL SET[1:?] of OrganizationOrPersonInOr-ganiza-tionAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefini-tionAssign-ment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssign-ment
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 27: "SpecificationCategoryAssignment" Attributes

Attribute recommendations:

- **AssignedCategory:** the SpecificationCategory that is associated with the ProductClass (see Template “Specification”).
- **Description:** the text by which the SpecificationAssignment is described. The value of this attribute need not be specified. Use “Description” template (see [242-PAS]).
- **Mandatory**: specifies whether the Specification objects referring to the associated SpecificationCategory have to be used or may be used (optional) for products within the referenced ProductClass. A value of 'true' indicates that the usage is mandatory.

  Example: the SpecificationCategory ‘radio’ may be associated ‘optional’ to the ProductClass of a car; the SpecificationCategory ‘engine’ is an example for a ‘mandatory’ association.
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

If the source system does not support Mandatory, it should be set to 'false'.

Consistency between the SpecificationCategoryAssignments and the SpecificationAssignments: the SpecificationCategory and its Specifications should be associated to the same ProductClass(es).

If the source system allows to associate a SpecificationCategory to more than one ProductClass, then multiple SpecificationCategoryAssignments may use the same SpecificationCategory.

Postprocessor Recommendations:

If the target system does not support Mandatory, it should be ignored, or a warning shall be returned.

If the target system does not support the reuse of the same SpecificationCategory in multiple ProductClasses, then they shall be duplicated in each ProductClass.

4.3.4 Template "SpecificationConditionAssignment"

A SpecificationConditionAssignment exists only within the context of one ProductConcept.

It is not used to restrict the scope of ConditionalEffectivities to a given ProductClass (see the 'Preprocessor recommendations' below).

Supported (and recommended) conditions are:

- AndCondition
- OrCondition
- NotCondition
- EqualsCondition
- Condition (with no subtype) only for textual conditions, see section

The usage of these conditions is not recommended:

- OneOfCondition
- NotEqualsCondition

4.3.4.1 Example for an AND Condition

The following diagram will show an example how to add an AND-Condition for two effectivities or specifications.

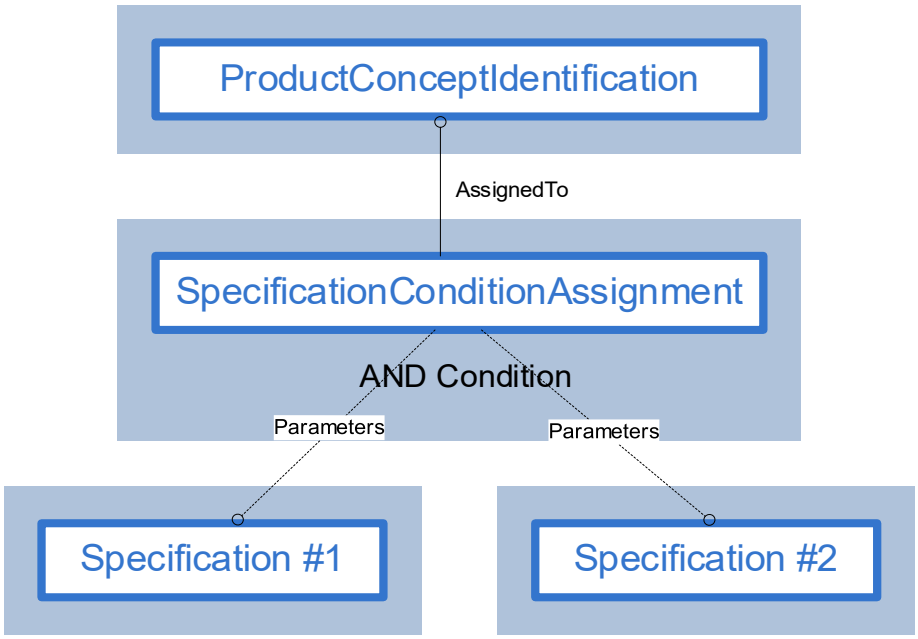


Figure 40: Example for an AND Condition for specifications

The Instance Model: AP242 Domain Model XML entities and attributes

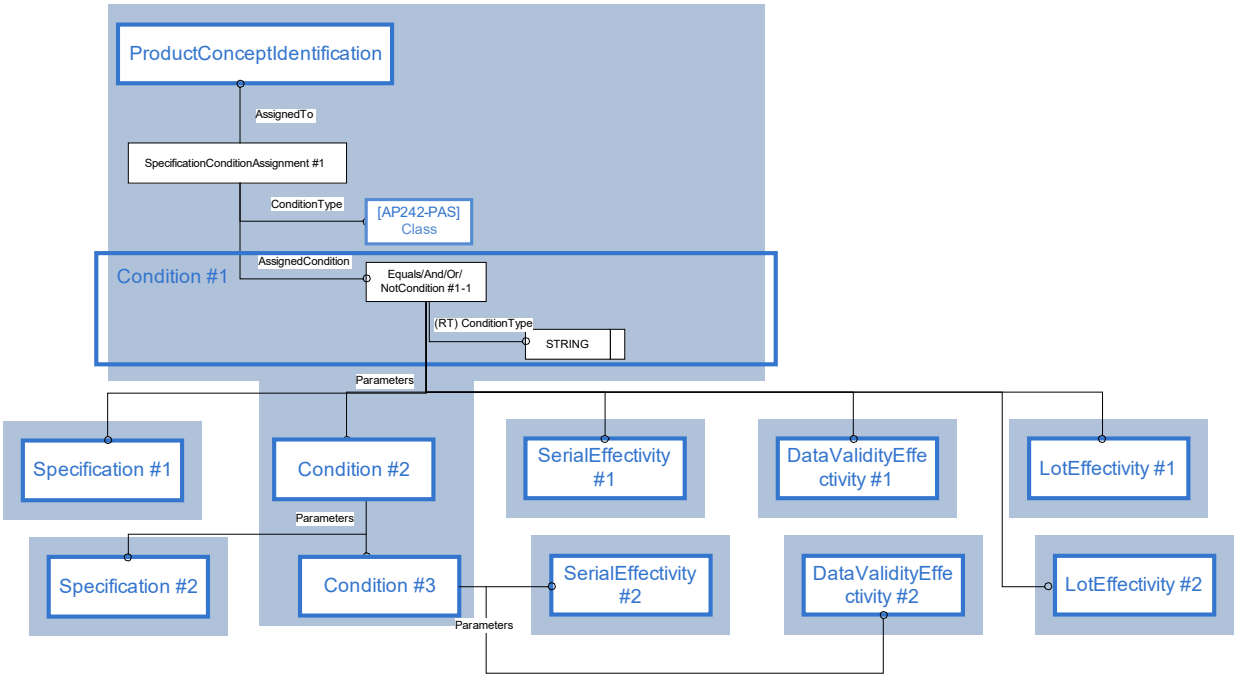


Figure 41: Instance Model SpecificationConditionAssignment

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
    <ProductConcept uid="pc--19088" xsi:type="n0:ProductClass">
…
      <SpecificationConditionAssignment uid="cda--19134">
        <AssignedCondition uid="spece--19133" xsi:type="n0:EqualsCondition">
          <ConditionType>
            <ClassString>equals</ClassString>
          </ConditionType>
          <Parameters>
            <Specification uidRef="spec--19191"/>
          </Parameters>
        </AssignedCondition>
        <ConditionType>
          <ClassString>part usage</ClassString>
        </ConditionType>
      </SpecificationConditionAssignment>
      <SpecificationConditionAssignment uid="cda--19135">
        <AssignedCondition uid="spece--19134" xsi:type="n0:EqualsCondition">
          <ConditionType>
            <ClassString>equals</ClassString>
          </ConditionType>
          <Parameters>
            <Specification uidRef="spec--19192"/>
          </Parameters>
        </AssignedCondition>
        <ConditionType>
          <ClassString>part usage</ClassString>
        </ConditionType>
      </SpecificationConditionAssignment>
    </ProductConcept>
    <Effectivity uid="condeff-Spec001" xsi:type="n0:ConditionalEffectivity">
      <Condition uidRef="spece--19133"/>
    </Effectivity>
     <Effectivity uid="condeff-Spec002" xsi:type="n0:ConditionalEffectivity">
      <Condition uidRef="spece--19134"/>
    </Effectivity>
```

More complex example to map the following condition: (for a more compact mapping, see section 4.3.5)

(Shimano AND (NOT Recourbe))

OR (Selle_en_mousse AND Bequille AND (NOT Magura) AND (NOT Recourbe))

OR (Droit AND (NOT Magura))

```
    <Effectivity xsi:type="n0:ConditionalConfiguration" uid="Eff__87">
      <Condition uidRef="AC__28"/>
      <ConfigurationType>
        <ClassString>usage</ClassString>
      </ConfigurationType>
      <InheritanceType>
        <ClassString>local</ClassString>
      </InheritanceType>
    </Effectivity>
    <ProductConcept xsi:type="n0:ProductClass" uid="PC__26">
      <Id>
        <Identifier uid="I__25" id="Ville" idRoleRef="rl-ii" idContextRef="o-
000000178"/>
      </Id>
```

```
<Name>
  <CharacterString>Ville</CharacterString>
</Name>
<SpecificationConditionAssignment uid="SCA__27">
  <AssignedCondition uid="AC__28" xsi:type="n0:OrCondition">
    <ConditionType>
      <ClassString>or</ClassString>
    </ConditionType>
    <Parameters>
      <Condition uid="AC__30" xsi:type="n0:AndCondition">
        <ConditionType>
          <ClassString>and</ClassString>
        </ConditionType>
        <Parameters>
              <Specification uidRef="SP__34"/>
          <Condition uid="AC__40" xsi:type="n0:NotCondition">
            <ConditionType>
              <ClassString>not</ClassString>
            </ConditionType>
            <Parameters>
              <Specification uidRef="SP__44"/>
            </Parameters>
          </Condition>
        </Parameters>
      </Condition>
      <Condition uid="Arl—ii0" xsi:type="n0:AndCondition">
        <ConditionType>
          <ClassString>and</ClassString>
        </ConditionType>
        <Parameters>
            <Specification uidRef="SP__52"/>
          <Condition uid="Arl—ii7" xsi:type="n0:NotCondition">
            <ConditionType>
              <ClassString>not</ClassString>
            </ConditionType>
            <Parameters>
              <Specification uidRef="SP__59"/>
            </Parameters>
          </Condition>
        </Parameters>
      </Condition>
      <Condition uid="AC__64" xsi:type="n0:AndCondition">
        <ConditionType>
          <ClassString>and</ClassString>
        </ConditionType>
        <Parameters>
          <Specification uidRef="SP__68"/>
          <Specification uidRef="SP__74"/>
          <Condition uid="AC__79" xsi:type="n0:NotCondition">
            <ConditionType>
              <ClassString>not</ClassString>
            </ConditionType>
            <Parameters>
              <Specification uidRef="SP__44"
            </Parameters>
          </Condition>
          <Condition uid="AC__79-1" xsi:type="n0:NotCondition">
            <ConditionType>
              <ClassString>not</ClassString>
            </ConditionType>
            <Parameters>
```

```
                        <Specification uidRef="SP__59"/>
                    </Parameters>
                  </Condition>
                </Parameters>
              </Condition>
            </Parameters>
          </AssignedCondition>
          <ConditionType>
            <ClassString>part usage</ClassString>
          </ConditionType>
        </SpecificationConditionAssignment>
      </ProductConcept>
      <SpecificationCategory uid="SC__32">
        <Description>
          <CharacterString>Freinage description</CharacterString>
        </Description>
        <Id>
          <Identifier uid="I__31" id="Freinage" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
        </Id>
        <ImplicitExclusiveCondition>true</ImplicitExclusiveCondition>
        <Specification uid="SP__59">
          <Id>
            <Identifier uid="I__58" id="Magura" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
          </Id>
          <Name>
            <CharacterString>Magura</CharacterString>
          </Name>
          <Package>false</Package>
        </Specification>
        <Specification uid="SP__34">
          <Id>
            <Identifier uid="I__33"id="Shimano" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
          </Id>
          <Name>
            <CharacterString>Shimano</CharacterString>
          </Name>
          <Package>false</Package>
        </Specification>
      </SpecificationCategory>
      <SpecificationCategory uid="SC__42">
        <Description>
          <CharacterString>Cintre description</CharacterString>
        </Description>
        <Id>
          <Identifier uid="I__41" id="Cintre" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
        </Id>
        <ImplicitExclusiveCondition>true</ImplicitExclusiveCondition>
        <Specification uid="SP__44">
          <Id>
            <Identifier uid="I__43" id="Recourbe" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
          </Id>
          <Name>
            <CharacterString>Recourbe</CharacterString>
          </Name>
          <Package>false</Package>
        </Specification>
```

```
      <Specification uid="SP__52">
        <Id>
          <Identifier uid="I__51" id="Droit" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
        </Id>
        <Name>
          <CharacterString>Droit</CharacterString>
        </Name>
        <Package>false</Package>
      </Specification>
    </SpecificationCategory>
    <SpecificationCategory uid="SC__66">
      <Description>
        <CharacterString>Confort description</CharacterString>
      </Description>
      <Id>
        <Identifier uid="I__65" id="Confort" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
      </Id>
      <ImplicitExclusiveCondition>true</ImplicitExclusiveCondition>
      <Specification uid="SP__68">
        <Id>
          <Identifier uid="I__67" id="Selle en mousse" idRoleRef="rl—ii" id-
ContextRef="o—000000178"/>
        </Id>
        <Name>
          <CharacterString>Selle en mousse</CharacterString>
        </Name>
        <Package>false</Package>
      </Specification>
      <Specification uid="SP__74">
        <Id>
          <Identifier uid="I__73" id="Bequille" idRoleRef="rl—ii" idContex-
tRef="o—000000178"/>
        </Id>
        <Name>
          <CharacterString>Bequille</CharacterString>
        </Name>
        <Package>false</Package>
      </Specification>
    </SpecificationCategory>
```

4.3.4.2 ENTITY SpecificationConditionAssignment

Attribute Name	Attribute Type
AssignedCondition	Condition
ClassifiedAs	OPTIONAL SET[1:?] of Classification
ConditionType	ClassSelect
Description	OPTIONAL DescriptorSelect
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
ActivityMethodAssignment	OPTIONAL SET[1:?] of ActivityMethodAssign-ment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment

AssignmentObjectRelationship	OPTIONAL SET[1:?] of AssignmentObjectRela-tionship
ConditionAssignment	OPTIONAL SET[1:?] of ConditionAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssignment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUs-ageRightAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssignment
OrganizationOrPersonInOrganizationAssignment	OPTIONAL SET[1:?] of OrganizationOrPersonInOrganizationAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefinitionAssignment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssignment
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 28: "SpecificationConditionAssignment" Attributes

Attribute recommendations:

- **AssignedCondition:** Specifies the Condition (see chapter 4.3.4.3) that is assigned to the ProductClass.
- **ConditionType:** specifies the meaning of the association. Use ClassString if the value is recommended within this document, otherwise use "Class" template (see [242-PAS]). According to the ISO AP242 Specification, where applicable, the following values shall be used:
    - 'design case' (currently out of scope): The Condition specifies a condition when a given object has to be designed and verified. This value of the conditionType is for information only and shall not be interpreted when querying design cases or usage cases. For such a query, the value of the attribute configurationType of ConditionalConfiguration shall be evaluated. This value may be used to precise when a given BreakdownElement has to be studied by the design department so that it provides solutions appropriate for the case specified by the attributes assignedCondition and assignedTo.
    - 'identification' (currently out of scope): The Condition specifies a condition that enables to distinguish the assigned ProductClass from other ProductClass objects. This value is not applicable for a top-level node in a hierarchy of ProductClass objects. This identification is part of the identification of all sub classes of this ProductClass;
    - 'part usage': The Condition specifies a condition for the usage of the components of an AlternativeSolution, the usage of an Occurrence or for the application of a ProcessPlan or a ProcessOperationOccurrence in the products of the assignedTo

ProductClass. In this case, the SpecificationConditionAssignment shall be referenced by at least one ConditionalConfiguration object;

  - 'validity' (currently out of scope): The Condition specifies a condition that is used to verify a SpecificationAssignment for the assignedTo ProductClass. That means that the Condition evaluates to 'true' if the set of Specification objects is valid; otherwise, it evaluates to 'false' with the meaning that the specified object is invalid for the ProductClass. The assignedCondition is valid for all products belonging to the assignedTo ProductClass if conditionType is 'identification' or 'validity'.

- **Description:** the text by which the SpecificationConditionAssignment is described. The value of this attribute need not be specified. Use "Description" template (see [242-PAS]).
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Preprocessor Recommendations:

- If relevant, the scope of an effectivity shall be mapped to Effectivity.EffectivityContext (see section 4.2). Embedding an SpecificationConditionAssignment in a given ProductClass doesn't specify its scope. It is just a mandatory aspect of the XML instantiation of the Domain Model, it has no semantic. However, it is recommended to use the following ProductClass to embed the SpecificationConditionAssignment:
  - If there is only one ProductClass with one version (VersionId is not used), all SpecificationConditionAssignments may be embedded into this ProductClass
  - If there is only one ProductClass and it is versioned, any version of the ProductClass may be chosen to embed the SpecificationConditionAssignments.
    - In case of VersionBranchEffectivities, it may be of advantage to embed the SpecificationConditionAssignments in the newest version of one of the ProductClasses used by the VersionBranchEffectivities.
  - If there are multiple ProductClasses (so-called 'dictionaries' in case of Specifications), the SpecificationConditionAssignments shall be embedded in the same ProductClass than the SpecificationAssignments and the SpecificationCategoryAssignments of the Specifications used by the SpecificationConditionAssignments.
    - If one SpecificationConditionAssignment uses Specifications from several different Dictionaries, it may be of advantage to embed the SpecificationConditionAssignment in a parent ProductClass related with each Dictionary (child ProductClass) via a ProductClassRelationship with RelationType='hierarchy'.
- Until further use cases are in scope of this document, use SpecificationConditionAssignment only with ConditionType='validity'.

Postprocessor Recommendations:

- Ignore at the moment any SpecificationConditionAssociations not having ConditionType= 'part usage'.
- The ProductClass in which a SpecificationConditionAssociation is embedded has no semantic. If relevant, the scope of an Effectivity shall be mapped from Effectivity.EffectivityContext.

4.3.4.3 ENTITY Condition

A condition is a definition of the precedent that must be fulfilled before a statement or relationship becomes valid.

Following subtypes of condition are supported: EqualsCondition, OneOfCondition (not recommended), OrCondition, AndCondition, NotCondition, NotEqualsCondition (not recommended)

The parameters against which the condition is to be evaluated are identified by ConditionParameter. The target or consequence of a condition is represented by ConditionAssignment.

Attribute Name	Attribute Type
ClassifiedAs	OPTIONAL SET[1:?] of Classification
ConditionType	ClassSelect
Description	OPTIONAL DescriptorSelect
Id	OPTIONAL IdentifierSelect
Parameters	SET[1:?] OF ParameterSelect
RetentionPeriod	OPTIONAL SET[1:?] of RetentionPeriod
SameAs	OPTIONAL SET[1:?] of ProxySelect
ActivityAssignment	OPTIONAL SET[1:?] of ActivityAssignment
AnalysisAssignment	OPTIONAL SET[1:?] of AnalysisAssignment
ApprovalAssignment	OPTIONAL SET[1:?] of ApprovalAssignment
ConditionAssignment	OPTIONAL SET[1:?] of ConditionAssignment
DateAndPersonAssignment	OPTIONAL SET[1:?] of DateAndPersonAssign-ment
DateTimeAssignment	OPTIONAL SET[1:?] of DateTimeAssignment
DocumentAssignment	OPTIONAL SET[1:?] of DocumentAssignment
EffectivityAssignment	OPTIONAL SET[1:?] of EffectivityAssignment
EventAssignment	OPTIONAL SET[1:?] of EventAssignment
InformationUsageRightAssignment	OPTIONAL SET[1:?] of InformationUs-ag-eRightAssignment
ModelPropertyAssignment	OPTIONAL SET[1:?] of ModelPropertyAssign-ment
OrganizationOrPersonInOrganizationAssign-ment	OPTIONAL SET[1:?] of OrganizationOrPer-sonInOrganizationAssignment
PropertyDefinitionAssignment	OPTIONAL SET[1:?] of PropertyDefini-tionAssign-ment
PropertyValueAssignment	OPTIONAL SET[1:?] of PropertyValueAssign-ment
SuppliedObjectRelationship	OPTIONAL SET[1:?] of SuppliedObjectRelation-ship
TimeIntervalAssignment	OPTIONAL SET[1:?] of TimeIntervalAssignment

Table 29: “Condition” Attributes

Attribute recommendations:

- **ConditionType:** the meaning of the condition. When using the subtypes of Condition, the value of ConditionType shall comply to the name of the subtype, i.e. 'equals', 'and', 'or', 'not'. 'one of' and 'not equals' are not recommended.
- **Description:** the text by which the Condition is described. The value of this attribute need not be specified. Use "Description" template (see [242-PAS]).
- **Parameters:** specifies the set of specifications, SerialEffectivities, DatedEffectivities, LotEffectivities or embedded conditions that represent the parameters against which the condition is to be evaluated.

  Comments:

  - *Conditions have to be embedded either in SpecificationConditionAssigment.AssignedCondition or in Condition.Parameters*
  - *Combined Effectivities with ConditionalEffectivities are supported. Remark: if no Specification is involved in the Condition, even if the effectivities have no context, the Condition has to be defined within a SpecificationConditionAssignment (again within a ProductClass) => in such a case, a ProductClass with Id='/NULL' has to be created.*
- Other attributes than these are not covered by these Recommended Practices; their use is discouraged as it would depend on mutual agreements between data exchange partners.

Example of combined effectivities within a condition:

Spec001 AND #3-3

```
    <ProductConcept uid="pc–19088" xsi:type="n0:ProductClass">
…
      <SpecificationConditionAssignment uid="cda–19136">
        <AssignedCondition uid="spece–19136" xsi:type="n0:AndCondition">
          <ConditionType>
            <ClassString>and</ClassString>
          </ConditionType>
          <Parameters>
            <Specification uidRef="spec–19192"/>
            <Effectivity uidRef="sneff-33-2"/>
          </Parameters>
        </AssignedCondition>
        <ConditionType>
          <ClassString>part usage</ClassString>
        </ConditionType>
      </SpecificationConditionAssignment>
  …
  </ProductConcept>
```

Preprocessor Recommendations:

Although not defined as an abstract supertype, it is recommended to instantiate only the subtypes of Condition, except for textual conditions (see section 4.3.5).

The usage in 'Parameters' of further objects defined in ParameterSelect (other than Specification, SerialEffectivity, DatedEffectivity, LotEffectivity and the subtypes of Condition) is not recommended. Especially the usage of ConditionalEffectivity is not recommended.

The usage of EqualsConditions is recommended only for the top-level Condition referenced directly by a ConditionalEffectivity.Condition, but not for 'Parameters'.

EqualsConditions and NotConditions should have only one value in 'Parameters'.

OrConditions and AndConditions should have at least two values in 'Parameters'.

Postprocessor Recommendations:

Further objects defined in ParameterSelect (other than Specification, SerialEffectivity, DatedEffectivity, LotEffectivity and the subtypes of Condition) in 'Parameters' shall be ignored, or a warning shall be returned.

4.3.5 Template "ConditionParameter"

Complex conditions (as in the example in section 4.3.4.1) may be mapped in a more compact way, as a text. For this purpose, a Condition (without any subtype) can reference a ConditionParameter via Parameters (instead of referencing further Conditions/Specifications). The plain text of the condition may be mapped to ConditionParameter.Description.

The Instance Model: AP242 Domain Model XML entities and attributes

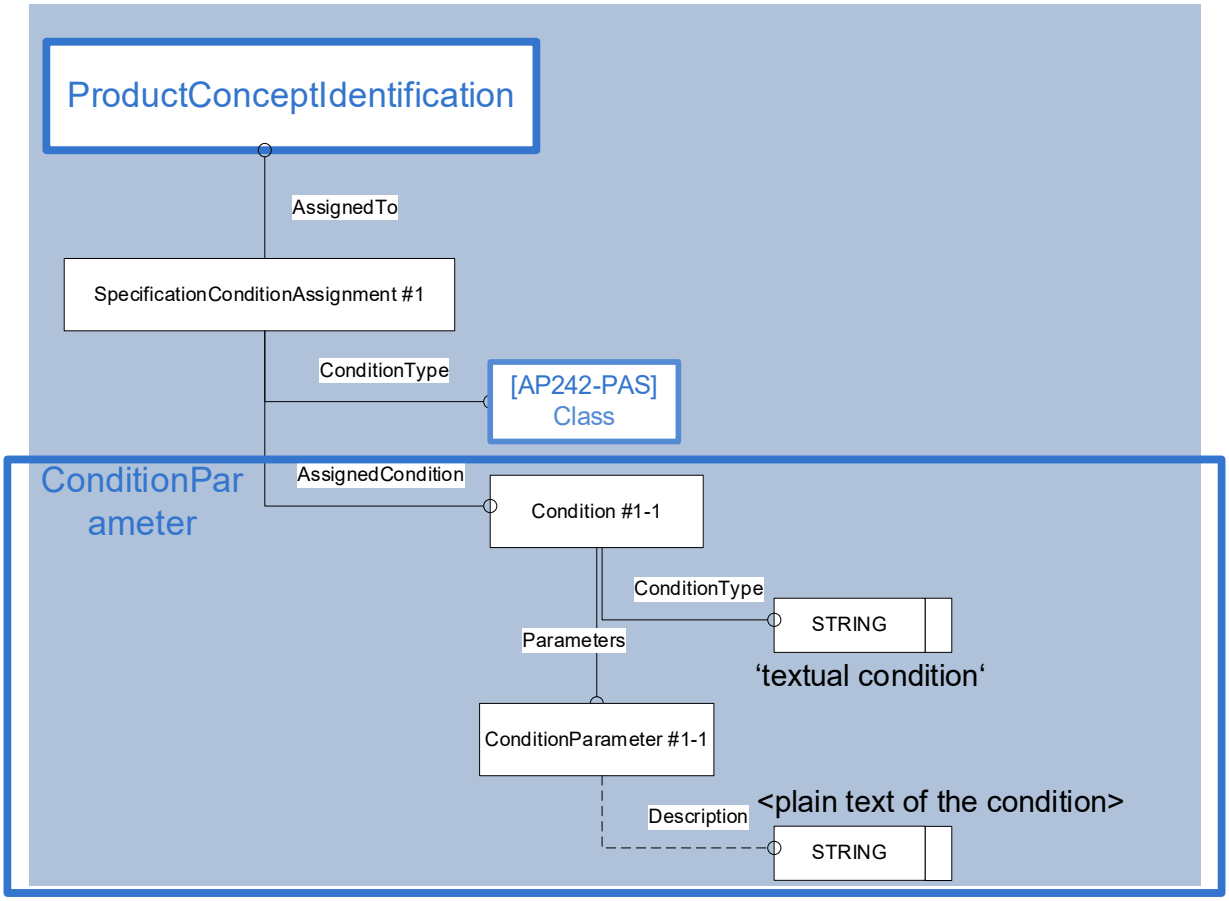


Figure 42: Instance Model ConditionParameter

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

Example to map the following condition:

(Shimano AND (NOT Recourbe))

OR (Selle_en_mousse AND Bequille AND (NOT Magura) AND (NOT Recourbe))

OR (Droit AND (NOT Magura))

```
<Effectivity xsi:type="n0: ConditionalConfiguration" uid="Eff__87">
  <EffectivityContext>PC__26</EffectivityContext>
    <Condition uidRef="AC__28"/>
    <ConfigurationType>
      <ClassString>usage</ClassString>
    </ConfigurationType>
    <InheritanceType>
      <ClassString>local</ClassString>
    </InheritanceType>
  </Effectivity>
<Condition uid="AC__28">
  <ConditionType>
    <ClassString>textual condition</ClassString>
  </ConditionType>
  <Parameters>
    <ConditionParameter uid="CP__1"/>
      <Description>
        <CharacterString>(Shimano AND (NOT Recourbe)) OR (Selle_en_mousse AND
Bequille AND (NOT Magura) AND (NOT Recourbe)) OR (Droit AND (NOT Ma-
gura))</CharacterString>
      </Description>
    </ConditionParameter>
  </Parameters>
</Condition>
```

Preprocessor Recommendations:

- The plain text may contain only Specification.Id,ids and Dated/SerialEffectivities combined with logical operators (AND, OR, NOT) and parenthesis (). The expression shall be parsed according to the priority rule “NOT before AND before OR”.
- The following syntax applies to range effectivities:
    - DatedEffectivity: xml:date-[xml:date] (Dates) or xml:string-[xml:string] (Events) similar to DatedEffectivity.Start/EndDefinition in section 4.2.2
    - SerialEffectivity: xml:string-[xml:string] similar to SerialEffectivity.Start/EndId in section 4.2.4
    - VersionBranchEffectivity: xml:string1:*[*xml:string2-[xml:string3]*]* similar to VersionBranchEffectivity Root/Leaf (string2 and string3) in section 4.2.6, where string1 is the ProductClass.Id.id. For example: ‘C205:[A.1-B.3]’
    - The upper value of the range is [optional]
- This syntax applies if all the Effectivities shall have the same EffectivityContext, and all the Specifications shall belong to the same Dictionary (ProductClass)
- Context prefix: if multiple Dictionaries are involved within one textual condition, each range effectivity / specification shall be preceeded by the Id.id of the context object (for example ProductClass.Id.id) + ‘:’. For example: ‘C205:1-10’.
- All the specification(categories) used by the textual condition shall exist within the ProductClass used as EffectivityContext (if the textual condition does not contain context prefixes) or shall exist within the context given as context prefix (as SpecificationAssignments / SpecificationCategoryAssignments).

Postprocessor Recommendations:

- An empty string or invalid syntax within the textual condition, as well as non-existing Specifications in the given context shall cause an error during import.

4.4 Usage of Option Pool (Dictionary)

An option pool defines all options with the possible values that could be used to configure a product within a product context.

The configuration options used in the effectivities along a Product structure need to have a context (i.e. a ProductClass in which they are available). This ProductClass is called Option Pool or Dictionary.

Some PDM systems support only a global dictionary (applies to all ProductClasses), others only local dictionaries (apply to one single ProductClass), others both.

The complete option pool itself must not be part of the AP242 file containing the configured product structure but shall be part of a contract between the exchange partners. The contract should also contain a rule to match the relation between the context and the option set.

For the use case “archiving” the option pool should also be defined in the AP242 file containing the configured product structure.

For the use case “internal PLM” the context should be common within all involved systems. Therefore, the Id attribute can be used as foreign key to get all information for the context as well as option pool.

Common Recommendations:

Create contract with the partner to define the supported option pool

Send options inside of each AP242 File together with the context

Preprocessor Recommendations:

Reduce the used options to the options that are part of the contract for the data exchange.

For the use case “Internal PLM” the **Id** attribute of the ProductClass could be used to match the relation between the context and the option set.

For the use case “supplier exchange”, the **Name** attribute has to used to match the relation to the option pool.

For the use case “archiving”, the option pool itself should be part of the AP242 file.

A global dictionary shall be mapped to a ProductClass.Id ‘global dictionary’.

Postprocessor Recommendations:

If the target system does not support local dictionaries, an appropriate logic has to be implemented like:

- all incoming options are imported without context,
- prefix the option names with the name of the incoming dictionary
- …

If the target system does not support a global dictionary, an appropriate logic has to be implemented like:

Duplicate all incoming options in each involved local dictionary.

5 Use cases for the exchange of configured product structures

5.1 *Export of effectivity-based single product configuration as explicit assembly structure without effectivities*

This use case applies when the data receiver has no PDM functionality to filter the data applying to a given ProductConfiguration.

Preprocessor Recommendations:

The sender performs the filtering on his side and sends only the result. The ProductConfiguration definition is also sent, so the receiver can handle many of them.

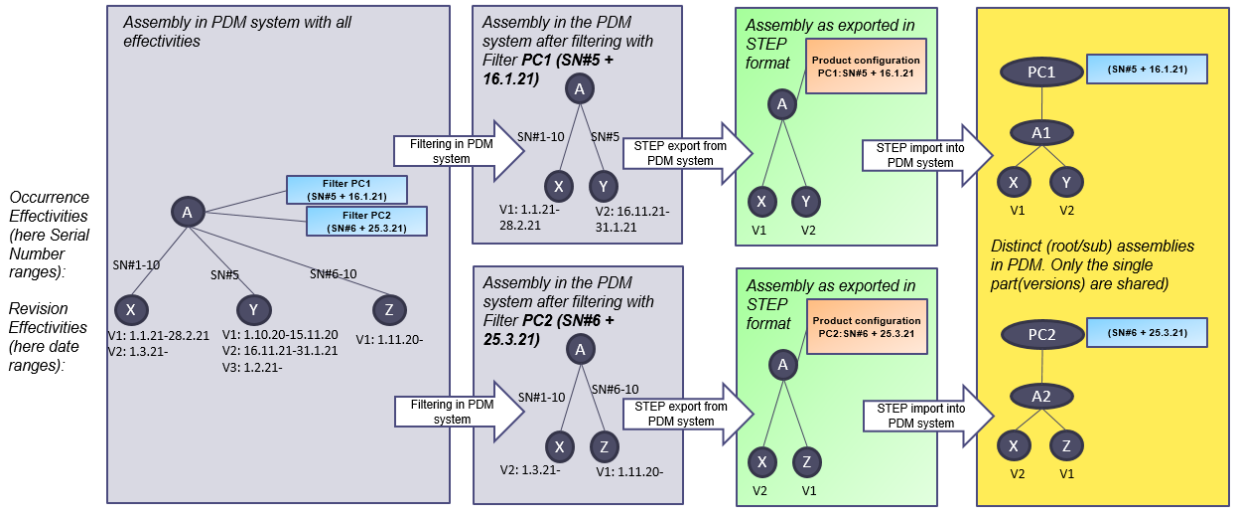


Figure 43: Example for overall process overview for Export of effectivity based single product configuration as explicit assembly structure without effectivities

Postprocessor Recommendations:

There are several ways for the receiver to import the ProductConfiguration definition:

- Create a Filter object and attach it to the top-level part node of the exchanged explicit assembly (if possible in his system)
- Create a dummy part node and store the ProductConfiguration definition:
  - Part.Id/VersionId = automatically generated
  - Name = Name of ProductConfiguration
  - Description = textual representation of the effectivities defining the ProductConfiguration

  and attach the top-level part node of the exchanged explicit assembly under it.

In all cases, all assembly nodes of the exchanged assembly structure have to be created each time. No sharing across many product configurations is possible, since the assembly doesn't support variance.

The single parts may be shared across many product configurations as long as all versions of the part are exchangeable (no revision effectivities) and the receiver shall retrieve always the newest one.

5.2 Export of effectivity-based single product configuration as explicit assembly structure with effectivities

Like the previous section, this use case applies when the data receiver has no PDM functionality to filter the data applying to a given ProductConfiguration.

The difference is, that for information, the effectivities defined along the product structure in the sender PDM system are also sent.

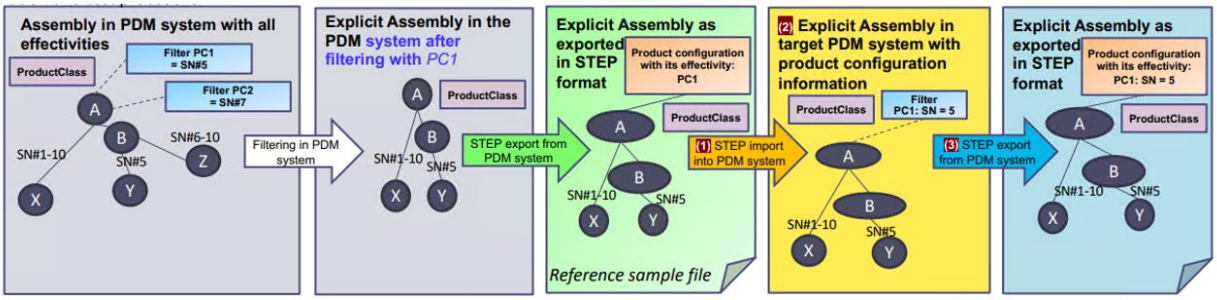


Figure 44: Example for overall process overview for Export of effectivity based single product configuration as explicit assembly structure with effectivities

Preprocessor Recommendations: see previous section

Postprocessor Recommendations: see previous section

5.3 Export of specification-based single product configuration as explicit assembly structure without effectivities

Not specified yet.

5.4 Export of dictionaries

Not specified yet.

6 Managing nested file structure

The recommendations how to split large product structure in a so-called nested structure are defined in the Core Document [242-PAS]. Regarding Configuration Management information, the following additional recommendations are made:

- ProductConfigurations, their AssignedEffectivities and ProductDesignAssociations as well as Specifications and SpecificationCategories shall be stored in the XML File where the ProductClass (as their definition scope) is defined, rather than in the file where the PartVersion referenced by ProductDesignAssociation via AssociatedDesign is defined. Doing so, the Part is independent from which ProductConfigurations are defined on it.

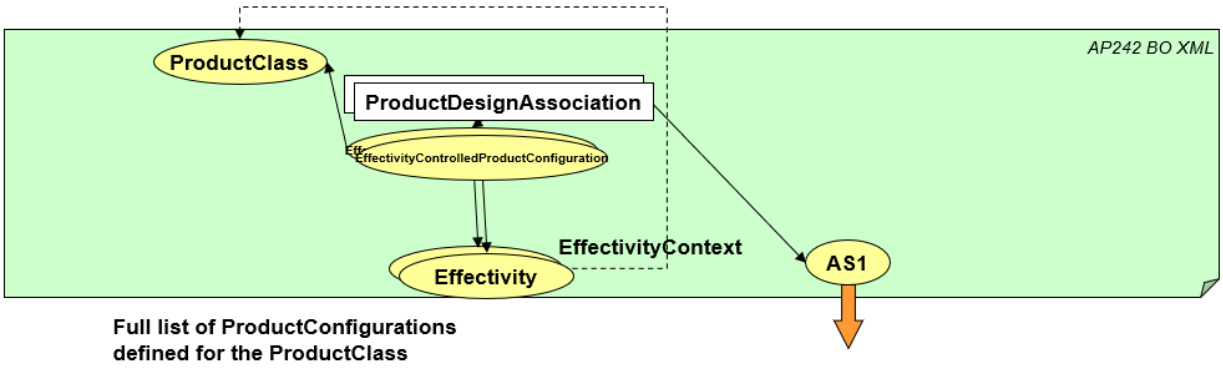


Figure 45: Example for Nested Structure with ProductConfigurations making no use of Specifications

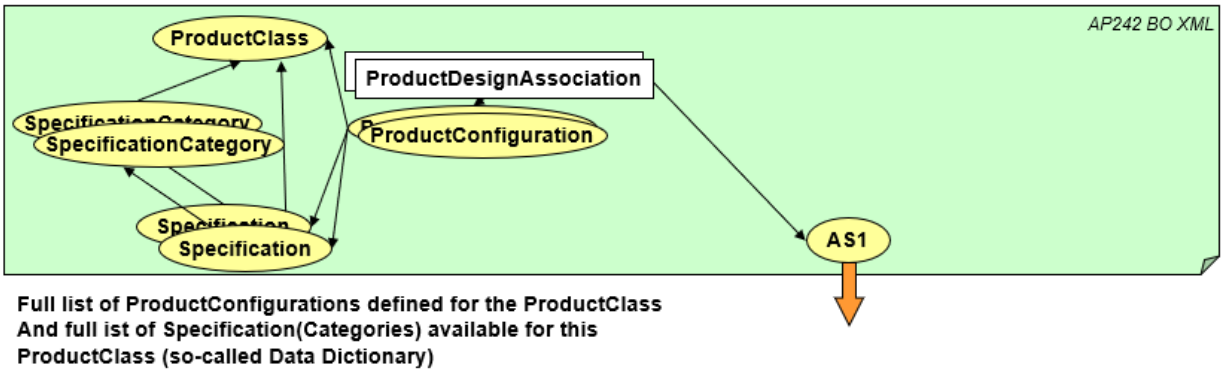


Figure 46: Example for Nested Structure with ProductConfigurations making only use of Specifications

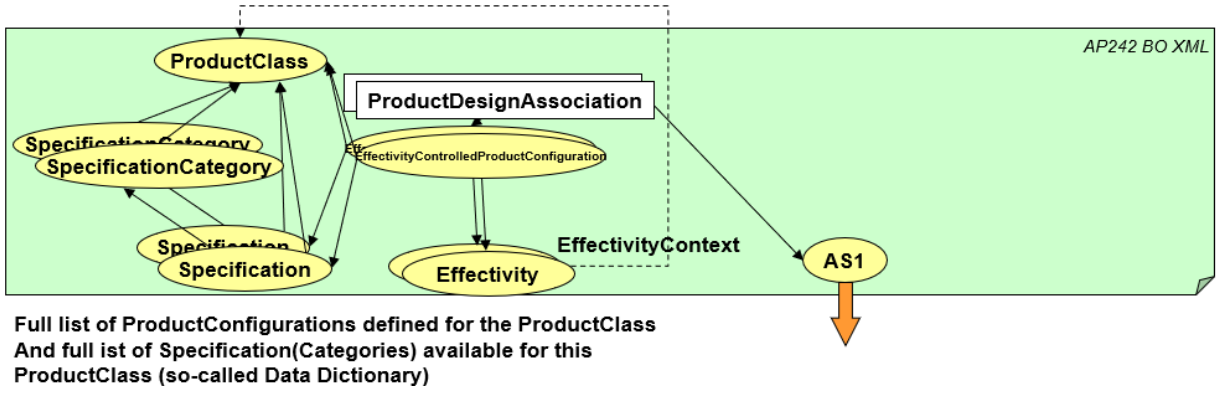


Figure 47: Example for Nested Structure with ProductConfigurations making use of Effectivities and Specifications

- EffectivityAssignments/Conditions and Effectivities on PartVersion/NextAssemblyOccurrenceUsage/NextAssemblyViewUsage and all their attributes shall be defined in the superordinate assembly node file, rather than in the subordinate component file. Doing so, the component file is independent from where and how often it is built into product structures.

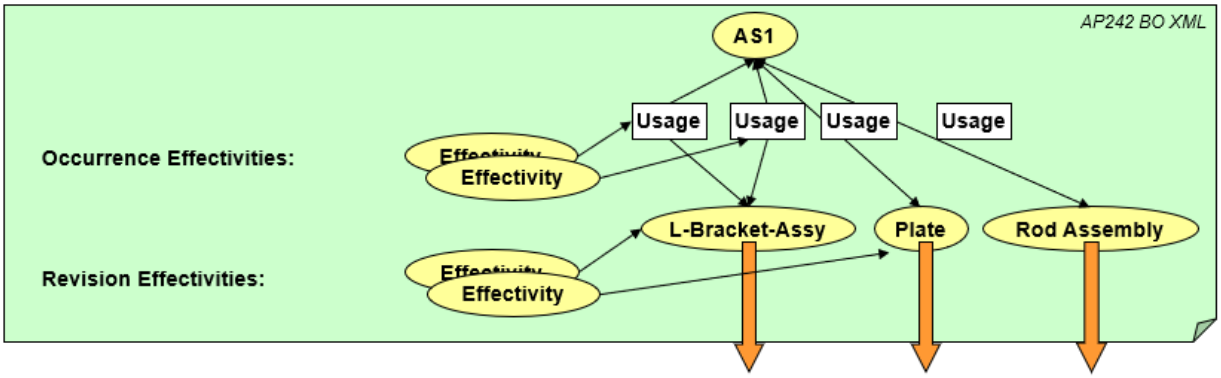


Figure 48: Example for Nested Structure with Effectivities (having no context) making no use of Specifications nor Conditions

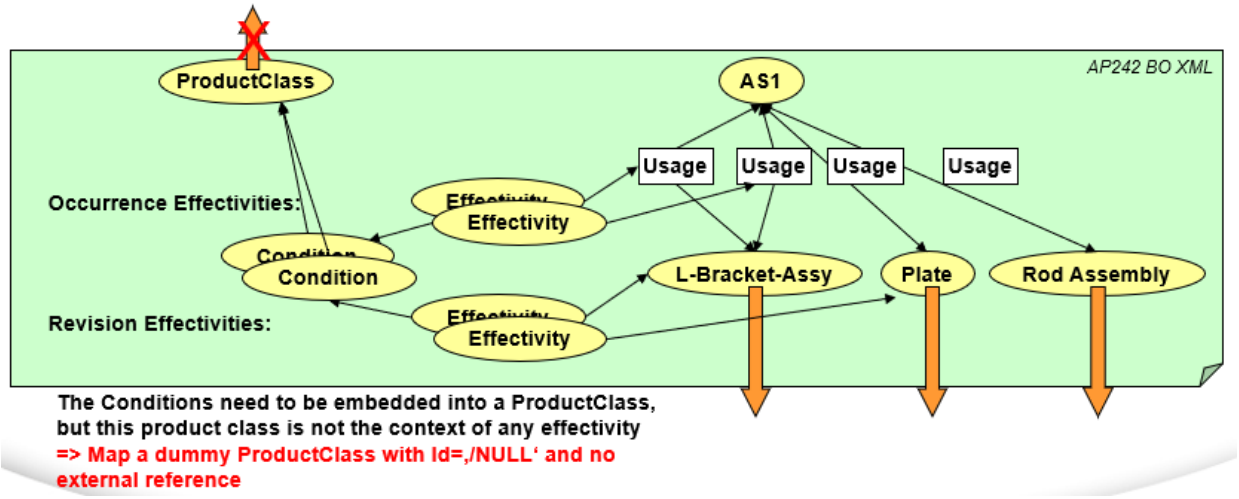


Figure 49: Example for Nested Structure with Effectivities (having no context) with Conditions but making no use of Specifications

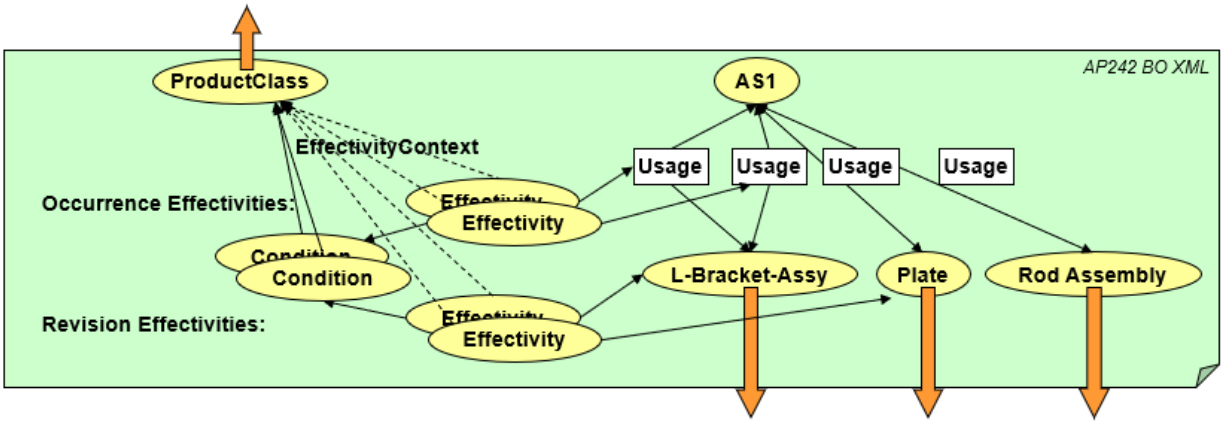


Figure 50: Example for Nested Structure with Effectivities (having a context) making no use of Specifications

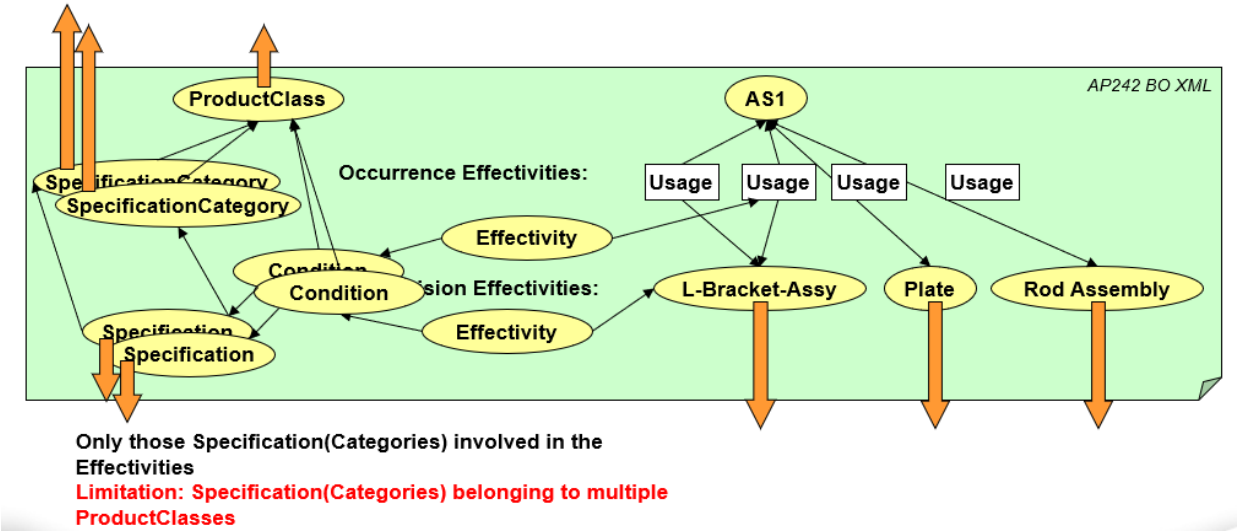


Figure 51: Example for Nested Structure with Effectivities making only use of Specifications

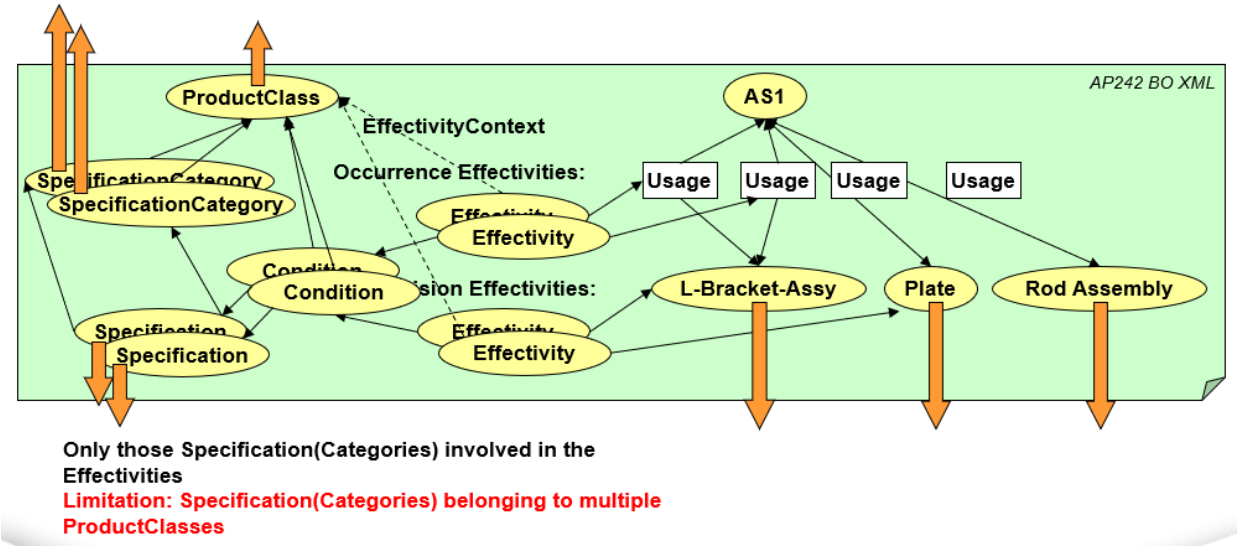


Figure 52: Example for Nested Structure with Effectivities making use of Effectivities and Specifications

There are two possible ways to map the nested references:

- Before AP242 Edition 4: the so-called '*specified* Reference' mechanism, see [242-PAS] section 9.3.
- With AP242 Edition 4: the so-called External Element Reference (EER) mechanism, see [242-PAS] section 9.3.

In all cases, ProductClass(es), ProductConfiguration(s), Specification(s) and SpecificationCategory(ies) may be referenced:

- as EffectivityContext of the Effectivities (defined on the PartVersion/NextAssemblyOccurrenceUsages/NextAssemblyViewUsage of the part described by the nested file)

- as SpecificationAssignment / SpecificationCategoryAssignment / SpecificationConditionAssignment involved in the above Effectivities
- as ProductClassRelationship.Related (in case the nested file describes a ProductClass having subordinated ProductClasses, and not a Part)

Using the 'specified reference' mechanism, a referenced component is mapped by :

1. mapping a minimum set of entities and attributes (subset of those mapped in the referenced XML file or in the monolithic mapping):
   - ProductClass.id with idRoleRef and idContextRef,
   - ProductClass.VersionId (if relevant for the data exchange),
   - ProductConfiguration.Id
   - ProductConfiguration.VersionId (if relevant for the data exchange),
   - Specification.id with idRoleRef and idContextRef,
   - Specification.VersionId (if relevant for the data exchange),
   - Specification.Category (since mandatory)
   - Specification.Package (!? since mandatory)
   - SpecificationCategory.id with idRoleRef and idContextRef,
   - SpecificationCategory.Description ('/NULL' since mandatory)
   - SpecificationCategory.ImplicitExclusiveCondition (!? since mandatory)
2. mapping the reference to the ProductClass XML file as DocumentAssignment between the ProductClass and a DigitalFile (no need to define DocumentAssignments on the referenced ProductConfiguration/Specifications/SpecificationCategories). In case a SpecificationCategory or Specification is associated to more than one ProductClass, they shall be mapped redundantly to each ProductClass
3. mapping a dedicated Classification 'specified reference' on the ProductClass(es), ProductConfiguration(s), Specification(s) and SpecificationCategory(ies)

Using the EER mechanism, a referenced component is mapped by:

1. mapping an ExternalRefBaseObject of type ProductConfiguration or Specification

The only exception is display in Figure 49: if no Effectivities have a Context nor use Specifications, the Conditions (which have to be defined within a ProductClass) may by mapped to a locally defined dummy ProductClass (having Id='/NULL').

If the nested files contain the product structure filtered for one ProductConfiguration, in all nested files (the one containing the ProductClass/ProductConfiguration definition and those containing the part/assembly nodes):

- according to section 4.1.2, ExchangeContext.IdentificationContext shall reference ProductConfiguration.Id in all nested files, no matter if the nested file contains effectivities or not. As a consequence, the ProductConfiguration and the Product Class shall be referenced as nested file in each nested file).

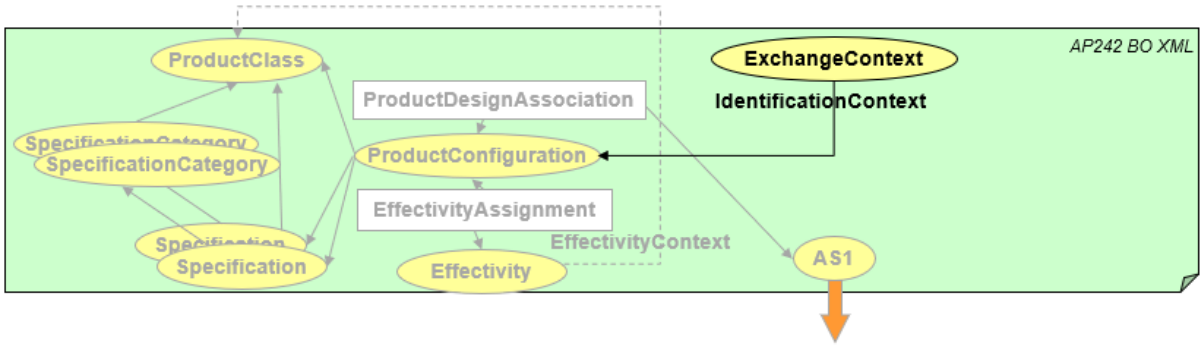


Figure 53: Example for Nested Structure of the product structure filtered for one ProductConfiguration

and in those nested files containing the part/assembly nodes, ProductConfiguration shall be mapped as reference.

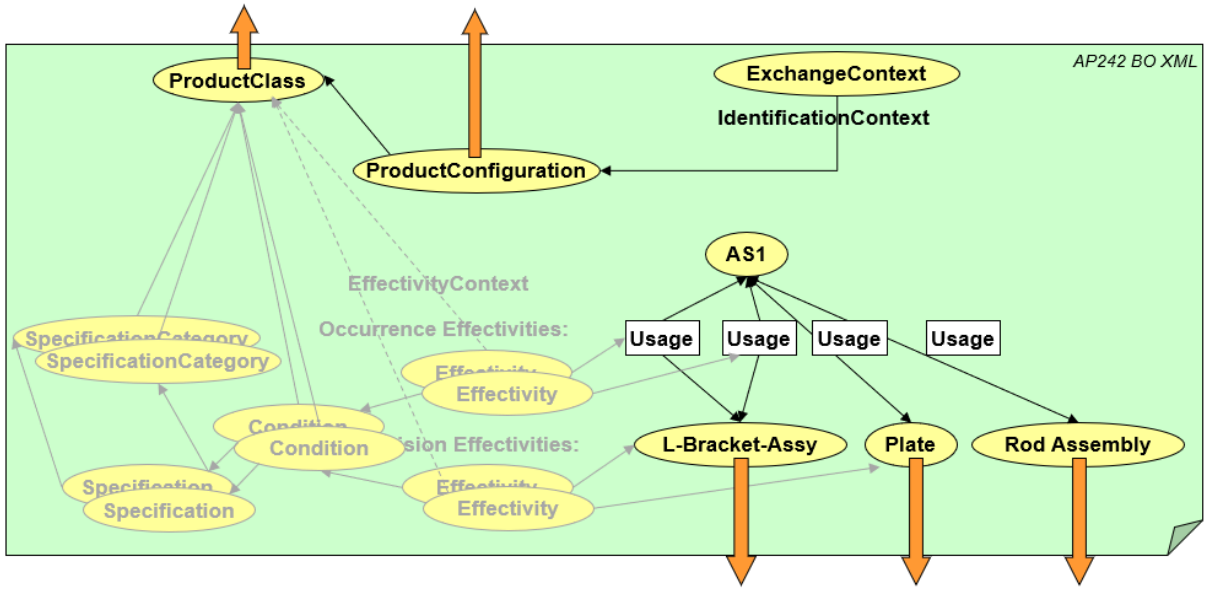


Figure 54: Example for Nested Structure of the product structure filtered of one ProductConfiguration

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<ExchangeContext uid="ID_987">
  <DefaultLanguage>en-US</DefaultLanguage>
  <DefaultUnit uidRef="ID_x20fbf17c1c0"/>
  <Description>
    <CharacterString>AP242 BO Model XML Assembly Structure exchange</Charac-
terString>
  </Description>
  <IdentificationContext uidRef="ID_982"/>
</ExchangeContext>
…
```

```
<Classification uid="nested_pv">
  <Class>
    <ClassString>specified reference</ClassString>
  </Class>
</Classification>
…
<FormatProperty uid="nested_ff">
  <CharacterCode>
    <ClassString>UTF-8</ClassString>
  </CharacterCode>
  <DataFormat>
    <ClassString>ISO 10303-242 Domain Model XML</ClassString>
  </DataFormat>
</FormatProperty>
…
<ProductConcept uid="ID_969" xsi:type="bom:ProductClass">
  <ClassifiedAs>
    <Classification uidRef="nested_pv"/>
  </ClassifiedAs>
  <Id>
    <Identifier uid="ID_964" id="Configuration test" idRoleRef="ID_962" id-
ContextRef="ID_963"/>
  </Id>
  <ProductConfiguration uid="ID_986">
    <ClassifiedAs>
      <Classification uidRef="nested_pv"/>
    </ClassifiedAs>
    <Id>
      <Identifier uid="ID_982" id="Conf03" idRoleRef="ID_962" idContex-
tRef="ID_964"/>
    </Id>
    <VersionId>
      <Identifier uid="ID_985" id="A.1" idRoleRef="ID_962" idContex-
tRef="ID_982"/>
    </VersionId>
  </ProductConfiguration>
  <DocumentAssignment uid="ID_91_da">
    <AssignedDocument uidRef="ID_dict"/>
    <Role>
      <ClassString>mandatory</ClassString>
    </Role>
  </DocumentAssignment>
  <VersionId>
    <Identifier uid="ID_967" id="/NULL" idRoleRef="ID_962" idContex-
tRef="ID_964"/>
  </VersionId>
</ProductConcept>
…
<Classification uid="nested_pc">
  <Class>
    <ClassString>dictionary</ClassString>
  </Class>
</Classification>
…
<File uid="ID_dict" xsi:type="bom:DigitalFile">
  <FileContent uid="ID_90_cp" xsi:type="bom:ContentProperty">
    <GeometryTypes>
      <Classification uidRef="nested_pc"/>
    </GeometryTypes>
  </FileContent>
  <FileFormat uidRef="nested_ff"/>
```

```
  <FileType>
    <ClassString>structured product data</ClassString>
  </FileType>
  <Id>
    <Identifier uid="ID_dict_id" id="120232-VDA-A.1.stpx" idRoleRef="ID_962"
idContextRef="ID_965"/>
  </Id>
  <Locations>
    <ExternalItem uid="ID_dict_ei">
      <Id id="120232-VDA-A.1.stpx"/>
      <Source>
        <IdentifierString>.</IdentifierString>
      </Source>
    </ExternalItem>
  </Locations>
</File>
```

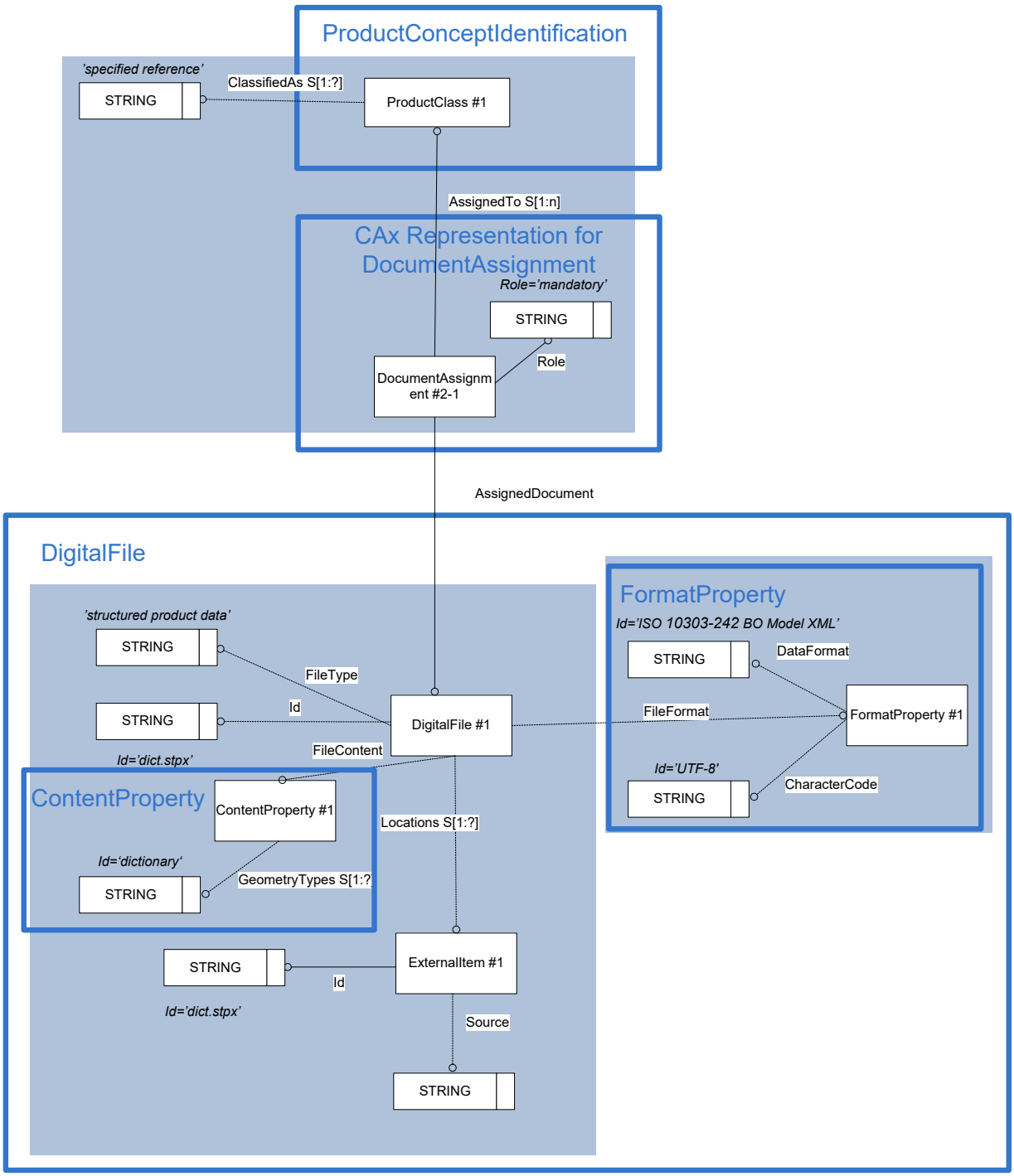


Figure 55: External reference to a ProductClass within a nested XML File containing EffectivityAssignments using 'specified reference' mechanism

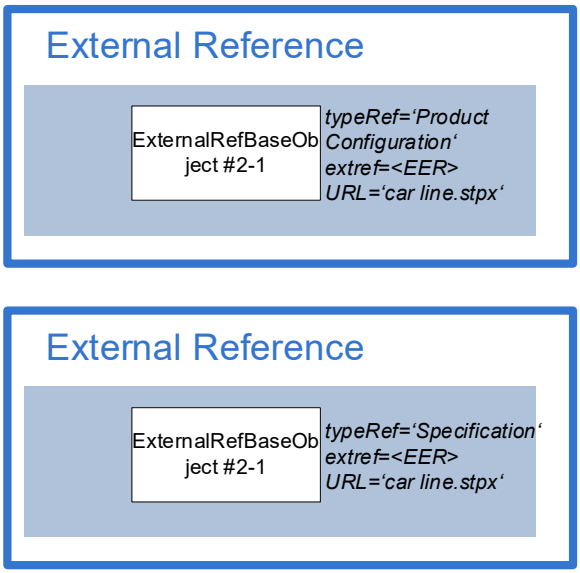


Figure 56: External reference to a ProductConfiguration or a Specification within a nested XML File containing EffectivityAssignments using EER mechanism

The Instance Model: STEP exchange file format (ISO10303 AP242 Domain Model XML syntax)

```
<Part uid="ID_208">
…
  <Versions>
    <PartVersion uid="ID_129">
…
      <Views>
        <PartView uid="ID_130" xsi:type="bom:AssemblyDefinition">
…
          <ViewOccurrenceRelationship uid="ID_250" xsi:type="bom:NextAssem-
blyOccurrenceUsage">
            <Related uidRef="ID_252"/>
            <RelationType>
              <ClassString>next assembly occurrence</ClassString>
            </RelationType>
            <EffectivityAssignment uid="EA__603591">
              <AssignedEffectivity uidRef="CC__603588"/>
              <EffectivityIndication>true</EffectivityIndication>
              <Role>
                <ClassString>actual</ClassString>
              </Role>
            </EffectivityAssignment>
…
          </ViewOccurrenceRelationship>
…
        </PartView>
      </Views>
…
    </PartVersion>
  </Versions>
</Part>


<Effectivity uid="DVE__603585" xsi:type="bom:DatedEffectivity">
  <EffectivityContext uidRef="pc--19088"/>
  <EndDefinition>
```

```
    <DateTimeString>2017-12-31T23:59:59.0Z</DateTimeString>
  </EndDefinition>
</Effectivity>
<Effectivity uid="CC__603588" xsi:type="bom:ConditionalConfiguration">
  <Condition uidRef="AC__603586"/>
  <ConfigurationType>
    <ClassString>usage</ClassString>
  </ConfigurationType>
  <InheritanceType>
    <ClassString>local</ClassString>
  </InheritanceType>
</Effectivity>
<SpecificationCategory uid="SC__603581">
  <Id>
    <Identifier id="FreerideMTB" idContextRef="ID_64" idRoleRef="ID_66"
uid="SC__603581--id"/>
  </Id>
  <ImplicitExclusiveCondition>true</ImplicitExclusiveCondition>
  <Specification uid="S__603583">
    <Id>
      <Identifier id="FreerideMTB_Black" idContextRef="ID_64" idRol-
eRef="ID_66" uid="S__603583--id"/>
    </Id>
    <Package>false</Package>
  </Specification>
  <Specification uid="S__603592">
    <Id>
      <Identifier id="FreerideMTB_Red" idContextRef="ID_64" idRoleRef="ID_66"
uid="S__603592--id"/>
    </Id>
    <Package>false</Package>
  </Specification>
</SpecificationCategory>
<Classification uid="nested_pv">
  <Class>
    <ClassString>specified reference</ClassString>
  </Class>
</Classification>
<File uid="ID_90_df" xsi:type="bom:DigitalFile">
  <FileContent uid="ID_90_cp" xsi:type="bom:ContentProperty">
    <GeometryTypes>
      <Classification uidRef="nested_pc"/>
    </GeometryTypes>
  </FileContent>
  <FileFormat uidRef="nested_ff"/>
  <FileType>
    <ClassString>structured product data</ClassString>
  </FileType>
  <Id>
    <Identifier uid="ID_90_df_id" id="dictionary.stpx" idRoleRef="ID_66" id-
ContextRef="ID_64">
    </Identifier>
  </Id>
  <Locations>
    <ExternalItem uid="ID_90_ei">
      <Id id="dictionary.stpx"/>
      <Source>
        <IdentifierString>./</IdentifierString>
      </Source>
    </ExternalItem>
  </Locations>
```

```
</File>
<ProductConcept uid="pc--19088" xsi:type="bom:ProductClass">
  <ClassifiedAs>
    <Classification uidRef="nested_pv"/>
  </ClassifiedAs>
  <Id>
    <Identifier uid="pc-asclass--id1" id="Configuration test" idRol-
eRef="ID_66" idContextRef="ID_64"/>
  </Id>
  <DocumentAssignment uid="ID_91_da">
    <AssignedDocument uidRef="ID_90_df"/>
    <Role>
      <ClassString>mandatory</ClassString>
    </Role>
  </DocumentAssignment>
  <VersionId id="A.1"/>
  <SpecificationAssignment uid="SA__603584">
    <AssignedSpecification uidRef="S__603583"/>
    <AssociationType>
      <ClassString>availability</ClassString>
    </AssociationType>
  </SpecificationAssignment>
  <SpecificationAssignment uid="SA__603593">
    <AssignedSpecification uidRef="S__603592"/>
    <AssociationType>
      <ClassString>availability</ClassString>
    </AssociationType>
  </SpecificationAssignment>
  <SpecificationCategoryAssignment uid="SCA__603582">
    <AssignedCategory uidRef="SC__603581"/>
    <Mandatory>true</Mandatory>
  </SpecificationCategoryAssignment>
  <SpecificationConditionAssignment uid="SCA__603587">
    <AssignedCondition uid="AC__603586" xsi:type="bom:AndCondition">
      <ConditionType>
        <ClassString>and</ClassString>
      </ConditionType>
      <Parameters>
        <Specification uidRef="S__603583"/>
        <Effectivity uidRef="DVE__603585"/>
      </Parameters>
    </AssignedCondition>
    <ConditionType>
      <ClassString>part usage</ClassString>
    </ConditionType>
  </SpecificationConditionAssignment>
</ProductConcept>
```

The referenced XML file containing the ProductClass shall contain its full definition and the full definition of all available Specifications and SpecificationCategories (attached to the ProductClass via SpecificationAssignment and SpecificationCategoryAssignment), but no Effectivities, no SpecificationConditionAssignments and no Conditions, except the ones needed to define the ProductConfigurations.

Since the uids are only unique within one stpx file, the uids used in the referenced ProductClass XML file are independent from the uids used in the assembly node nested file.

```
<ProductConcept uid="PCE__603569" xsi:type="n0:ProductClass">
  <Id>
    <Identifier id="Configuration test" idContextRef="O__3" idRoleRef="C__5"
uid="PCE__603569--id"/>
  </Id>
  <Name>
    <CharacterString>Configuration test</CharacterString>
  </Name>
  <SpecificationAssignment uid="SAE__603584">
    <AssignedSpecification uidRef="SE__603583"/>
    <AssociationType>
      <ClassString>availability</ClassString>
    </AssociationType>
  </SpecificationAssignment>
  <SpecificationAssignment uid="SAE__603593">
    <AssignedSpecification uidRef="SE__603592"/>
    <AssociationType>
      <ClassString>availability</ClassString>
    </AssociationType>
  </SpecificationAssignment>
  <SpecificationCategoryAssignment uid="SCAE__603582">
    <AssignedCategory uidRef="SCE__603581"/>
    <Mandatory>true</Mandatory>
  </SpecificationCategoryAssignment>
</ProductConcept>
<SpecificationCategory uid="SCE__603581">
  <Description>
    <CharacterString>Freeride Mountain Bike description</CharacterString>
  </Description>
  <Id>
    <Identifier id="FreerideMTB" idContextRef="O__3" idRoleRef="C__5"
uid="SCE__603581--id"/>
  </Id>
  <ImplicitExclusiveCondition>true</ImplicitExclusiveCondition>
  <Specification uid="SE__603583">
    <Category uidRef="SCE__603581"/>
    <Id>
      <Identifier id="FreerideMTB_Black" idContextRef="O__3" idRoleRef="C__5"
uid="SE__603583--id"/>
    </Id>
    <Name>
      <CharacterString>Freeride Mountain Bike Black</CharacterString>
    </Name>
    <Package>false</Package>
  </Specification>
  <Specification uid="SE__603592">
    <Category uidRef="SCE__603581"/>
    <Id>
      <Identifier id="FreerideMTB_Red" idContextRef="O__3" idRoleRef="C__5"
uid="SE__603592--id"/>
    </Id>
    <Name>
      <CharacterString>Freeride Mountain Bike Red</CharacterString>
    </Name>
    <Package>false</Package>
  </Specification>
</SpecificationCategory>
```

Limitations of the current recommendation:

- Effectivities and Conditions shall be repeated in the XML files and not treated as referenced objects.
- Events (used for Milestone Effectivities) shall be repeated in the XML files and not treated as referenced object.

In order to compute reproduceable UUIDs v5 at each partner, it is recommended to generate the UUIDs out of the key attributes of the DomainModel objects:

- From ProductConfiguration: <<ProductConfiguration.Id[Identifier].Id><ProductConfiguration.Id[Identifier].idRoleRef.ClassString><ProductConfiguration.Id[Identifier].idContextRef.Id>> (only one role and context according to the Rec. Pracs. for PAS section 9.3)
- And from ProductConfiguration: <<ProductConfiguration.VersionId[Identifier].Id><ProductConfiguration.VersionId[Identifier].idRoleRef.ClassString><ProductConfiguration.VersionId[Identifier].idContextRef.Id>> (shall have the same role/context than ProductConfiguration.Id[Identifier])

Or

- From Specification: <<Specification.Id[Identifier].Id><Specification.Id[Identifier].idRoleRef.ClassString><Specification.Id[Identifier].idContextRef.Id>> (only one role and context according to the Rec. Pracs. for PAS section 9.3)
- And from Specification: <<Specification.VersionId[Identifier].Id><Specification.VersionId[Identifier].idRoleRef.ClassString><Specification.VersionId[Identifier].idContextRef.Id>> (shall have the same role/context than Specification.Id[Identifier])

Preprocessor Recommendation:

- in case of nested files using EER, all objects of kind ProductConfiguration, Specification shall be mapped with their attribute ‘uuid’.

7 Statistics and Validation Properties

As the development and testing of Configuration Management capabilities is just starting, no established statistics for testing or validation properties are available yet. It is a task for the test rounds in the involved implementor forums to define and improve these.

This section is intended to gather initial ideas for such statistics and properties, and to facilitate the discussion of these across the groups.

The exporting system will calculate the respective values based on the native model and write them as IntegerValues into the STEP file.

If some capabilities are missing in the STEP tool to export all data contained in the native model, they shall not be counted in the validation properties.

In all cases, the verification of the validation properties shall be possible without having the native model.

As defined within the CAx-IF; the receiving system upon import will re-create the information from the file, and derive the same attributes based on (mapping) the results. These will be compared to the data given in the file, and if the values match, the import will be deemed successful.

Note: like in [242-PAS] 6.2, It is recommended that all the properties attached to an object are spread over two instances of PropertyValueAssignment. One instance shall collect the properties that describe the object (having PropertyValueAssignment.ClassifiedAs=’<object> properties’);

the other instance shall collect the properties that describe the validation properties of the same object (having PropertyDefinition.PropertyType='configuration validation property' and PropertyValueAssignment.ClassifiedAs='validation properties').

The validation properties shall be mapped similarly to the assembly validation properties defined in the section 13.1 in [242-PAS].

The PropertyDefinition.PropertyType shall be 'configuration validation property' and the PropertyDefinition.Id shall be 'quality property'.

If an object has several validation properties (for example number of children and some configuration validation property), both PropertyValues shall be associated to the same PropertyValueAssignment (with Class 'validation properties').

Note: the verification of these validation properties might fail if the target system transforms the effectivities like SerialEffectivity '1-3' being mapped to SerialEffectivity '1-1' OR SerialEffectivity '2-2' OR SerialEffectivity '3-3'.
Further transformations like changing parenthesis or the order/use of the operators will not affect the verification. For example:

(SerialEffectivity '1-3' AND Specification 'a') OR (SerialEffectivity '1-3' AND Specification 'b') transformed to
SerialEffectivity '1-3' AND (Specification 'a' OR Specification 'b')

Other example of transformation that will not affect the verification: having two EffectivityAssignments: SerialEffectivity '1-3' as well as Specification 'a' might be transformed to SerialEffectivity '1-3' AND Specification 'a'.

Preprocessor Recommendations:

The use of NumericalValue for the Integer (i.e. not a real with comma or scientific notation) is deprecated and shall not be used anymore.

Postprocessor Recommendations:

For upward compatibility reasons, the use of NumericalValue shall be supported, especially for files former to Edition 4.

7.1.1 Statistics and Validation Properties on ProductClass

7.1.1.1 Number of Specification Categories

For each ProductClass, calculate the number of Specification Categories by counting the SpecificationCategoryAssignments.

For use as a validation property, store the result in a property of kind IntegerValue called 'number of specification categories', attached to the ProductClass.

As all properties that are counts, this is intended as a completeness check.

A value 0 shall be omitted and no validation property mapped.

7.1.1.2 Number of available Specifications

For each ProductClass, calculate the number of Specifications by counting the SpecificationAssignments with Role = 'availability'.

For use as a validation property, store the result in a property of kind IntegerValue called 'number of available specifications', attached to the ProductClass.

A value 0 shall be omitted and no validation property mapped.

7.1.1.3 Number of available Specifications per Specification Category

For each SpecificationCategory, calculate the number of Specifications by counting the instances of Specification.

For use as a validation property, store the result in a property of kind IntegerValue called ‘number of available specifications within category’, attached to the SpecificationCategory.

A value 0 shall not be omitted and shall be mapped as validation property.

7.1.2 Statistics and Validation Properties on ProductConfiguration

7.1.2.1 Number of involved Specifications

For each EffectivityControlledProductConfiguration, calculate the number of Specifications by counting the instances of Specification within EffectivityControlledProductConfiguration.Definition. In case of ConditionalEffectivity involving Conditions and sub-Conditions, all Conditions shall be traversed to compute the number of involved Specifications.

If the same Specification occurs multiple times within a ConditionalEffectivity having sub-Conditions, it shall be counted only once.

Note: since the use of ProductConfiguration.EffectivityAssignment and .DefiningSpecifications is deprecated, it is not recommended to compute this validation property using these two attributes.

For use as a validation property, store the result in a property of kind IntegerValue called ‘number of involved specifications’, attached to the ProductConfiguration.

A value 0 shall not be omitted and shall be mapped as validation property.

7.1.2.2 Number of simple Effectivities for each kind of simple Effectivity

For each EffectivityControlledProductConfiguration, calculate the number of simple Effectivities for each kind of simple Effectivity: Dated/Lot/Serial/TimeInterval/VersionBranch-Effectivity within EffectivityControlledProductConfiguration.Definition. In case of ConditionalEffectivity involving Conditions and sub-Conditions, all Conditions shall be traversed to compute the number of involved Specifications.

The name of the validation property of kind IntegerValue shall be overtaken from the Effectivity kind (dated, lot, serial, time interval, version branch), for example ‘number of dated effectivities’, ‘number of version branch effectivities’, …) and attached to the ProductConfiguration.

If the same simple Effectivity occurs multiple times within a ConditionalEffectivity having sub-Conditions, it shall be counted only once.

ConditionalEffectivities and its subtype ConditionalConfiguration shall not be counted,

Note: since the use of ProductConfiguration.EffectivityAssignment and .DefiningSpecifications is deprecated, it is not recommended to compute this validation property using these two attributes.

A value 0 shall be omitted and no validation property mapped.

7.1.3 Statistics and Validation Properties on the Product Structure

7.1.3.1 Number of involved Specifications

For each PartVersion and NextAssemblyOccurrenceUsage, calculate the number of Specifications by counting the instances of Specification within .EffectivityAssignment. In case of ConditionalEffectivities involving Conditions and sub-Conditions, all Conditions shall be traversed to compute the number of involved Specifications.

If the same Specification occurs multiple times within the ConditionalEffectivities having sub-Conditions, it shall be counted only once.

If multiple EffectivityAssignments apply to an object, all of them shall be considered.

For use as a validation property, store the result in a property of kind IntegerValue called ‘number of involved specifications’, attached to the PartView or NextAssemblyOccurrenceUsage.

A value 0 shall be omitted and no validation property mapped.

7.1.3.2 Number of simple Effectivities for each kind of simple Effectivity

For each PartVersion and NextAssemblyOccurrenceUsage, calculate the number of simple Effectivities for each kind of simple Effectivity: Dated/Lot/Serial/TimeInterval/VersionBranch-Effectivity within .EffectivityAssignment. In case of ConditionalEffectivities involving Conditions and sub-Conditions, all Conditions shall be traversed to compute the number of involved Specifications.

The name of the validation property of kind IntegerValue shall be overtaken from the Effectivity kind (dated, lot, serial, time interval, version branch), for example ‘number of dated effectivities’, ‘number of version branch effectivities’, …) and attached to the PartView or NextAssemblyOccurrenceUsage.

If the same simple Effectivity occurs multiple times within the ConditionalEffectivities having sub-Conditions, it shall be counted only once.

If multiple EffectivityAssignments apply to an object, all of them shall be considered.

ConditionalEffectivities and its subtype ConditionalConfiguration shall not be counted,

A value 0 shall be omitted and shall not be mapped as validation property.

Annex A References

Standard	Name
ISO 10007:2003	Quality Managament Systems – Guidelines for Configuration Management
EIA-649-A	National Consenes Standard for Configuration Management
CMII	Configuration Management II
ISO 10303:4442	Industrial automation systems and integration — Product data representation and exchange Part 4442 Domain model: Managed model based 3d engineering

Document
PDM Schema Usage Guide v4.3.
AP214 CC8 Recommended Practices Version 1.2b
Recommended Practices for STEP AP242 Edition 4 Domain Model XML Product & Assembly Structure [242-PAS]
Recommended Practices for STEP AP242 Edition 4 Domain Model XML Configuration Management [242-CFG]
Recommended Practices for STEP AP242 Edition 4 Domain Model XML Kinematics [242-KIN]
Recommended Practices for STEP AP242 Edition 4 Domain Model XML Product Manufacturing Information (PMI) [242-PMI]
Recommendation for Project Schedule Management Synchronization in Collaboration Networks (PSMS) [242-PSMS] Version 2.0, May 2025
STEP AP 242 Electrical Harness XML Tutorial & Recommended Practices [242-EWH] Version 2.0, November 2020

Annex B Known Issues

This section lists known issues with the AP242 Domain Model, both related to the assembly structure and to other domains within the Domain Model. These issues concern errors in the XSD, mismatches between the EXPRESS and XML schemas, deficiencies in the documentation and other issues that have already been communicated to the AP242 maintenance / development team for resolution. Many issues have already been resolved by the Technical Corrigendum 1 (TC1) of AP242, which was published in 2016, by the Ed.2 (2019), by the Ed.3 (2022) and by the Ed.4 (2025). The issues listed below are on the list to be resolved with the fifth Edition of AP242. Development of that started in 2025.

Since 2022, the Bugzilla list hosted by PDES Inc. has been transferred to ISO JIRA with new issue numbers. The old Bugzilla issue numbers are still mentioned in the short description where applicable.

ISO Jira Issue	Summary	Target Milestone	Status
TCSC410303-2662	ProductDesignAssociation shall support Property assignments	4442 Ed.6	To Do

Annex C Supported Configuration Management for PDM Systems

This chapter contains an overview of the various aspects of configuration management supported by each PLM System for the matching constraints to support the use cases of this document with AP242 standard. Not all capabilities of the PDM systems are mentioned here.

Since Aras currently does not support configuration management (announced for future release), the topics depicted below for Aras consider the productive data model extension of T-Systems.

Concept	STEP AP242 Domain Model	Teamcenter	Windchill	Enovia VPM	3DExperience	Aras (with T-Syst extensions)
Effectivities at PartVersions	PartVersion.EffectivityAssignment	X	X	?	X	-
Effectivities at Occurrences	NextAssemblyOccurrenceUsage.EffectivityAssignment	X	-	X	X	X
Date Range	DatedEffectivity	X	X	X	X	X
Milestone	DatedEffectivity	--	?	?	?	X
Time interval	TimeIntervalEffectivity	--	--	?	--	--
Serial number Range	SerialEffectivity	X (so-called Unit Effectivity)	x	X	X	X
Lot	LotEffectivity	?	X	?	?	X
Multiple intervals (combined implicitly with ‚OR')	Several EffectivityAssignments	X (for Serial)	X (For Date, Lot and Serial)	?	X	X
Specification	Specification	X	?	?	?	X

Specification attributes/versioning	X	--	?	?	?	--
SpecificationCategory	SpecificationCategory	--	?	?	?	X
SpecificationCategory attributes/versioning	X	--	?	?	?	X (but no versioning)
ConditionalEffectivity	ConditionalEffectivity	X (so-called 'classic variants'. The other methods are not detailed here (OpenPDM: OK?)	?	?	X	X
Logical operators	Equals, Or, And, Not, NotEquals, one of	AND, OR, NOT	?	?	?	AND, OR, NOT
Parenthesis	Condition in Condition	X, but NOT can only be used on single options (not for further condition) => similar to NotEquals	?	?	?	X
Logical combination of different effectivities	X	With the same type it's implicitly OR operand With different types it's an AND operand Deep logical combination, i.e. opt1 or (dateEff and serialEff), is not supported	?	?	X?	X

Effectivity context	X (optional)	Required for Serial Optional for Date Required for ConditionalEffectivity (through the context of the Specification(Categories)	Required for Serial and Lot Optional for Date	?	Required for Serial Optional for Date (support of contextual date) Required for ConditionalEffectivity	--
Effectivity Context as EndItem	--	The assembly structure link that has a conditional effectivity has to be related up or down to the EndItem where the Specification(Categories) are defined	The assembly structure link that has a conditional effectivity has to be related up or down to the EndItem where the Specification(Categories) are defined	?	?	--
Type of EffectivityContext	ProductClass or ProductConfiguration	End Item (the ProductClass is managed as an Item If the EndItem is a part, it has to be mapped in AP242 twice (as Part and as ProductClass)	End Item (the ProductClass is managed as an Item	?	Model (equivalent to ProductClass) or Product	--
ProductClass attributes/versioning	ProductClass	EndItem	?	?	?	--
EffectivityContext structure	ProductClassRelationship, ProductCo	EndItem assembly structure link.	?	?	?	--

	nfigura-tionRela-tionship	Not struc-ture of con-figuration filter				
Open intervals	SerialEf-fectivity (right-hand) DateEffec-tivity (right-hand and left-hand)	right-hand; UP (unlim-ited)/SO (stock is out)	left-hand	right-hand	Both sup-ported (right-hand and left-hand)	Both sup-ported (right-hand and left-hand)
Effectivity shar-ing	X	X	Not shared	Not shared	Not shared	Not shared
ProductConfig-uration	ProductCo nfiguration	?	?	?	?	X
ProductConfig-uration defini-tion	0-n ProductCo nfigura-tion.Defin-ingSpecifi-cations 0-n+ .Ef-fectiv-ityAssign-ments to Se-rial/Date/ Mile-stone/Lot/ TimeInter-val Effec-tivities (possibly with inter-vals)	0-n Specifi-cations + 0-1 Effective Date + 0-1 Effective Unit Num-ber with EndItem	?	?	?	0-n Specifica-tions + 0-1 Effective Date(interval) + 0-1 Effective Unit Num-ber(interval) + 0-1 Effective Lot + 0-1 Effective Milestone(in-terval)
Explicit link of a ProductCon-figuration to 1-n assembly top nodes	ProductDe signAssoci-ation	--	?	?	?	--

Explicit link of a ProductConfiguration to the assembly links that are valid for the given configuration	ConfiguredAssemblyEffectivity	--	?	?	?	--

Table 30: Supported Configuration Management approaches for PDM Systems