



# Product Lifecycle Management Services

*version 2.0*

---

OMG Document Number: formal/2009-08-01

Standard document URL: <http://www.omg.org/spec/PLM/2.0>

Associated Files\*: <http://schema.omg.org/specs/PLM/2.0/schema>

- <http://schema.omg.org/specs/PLM/2.0/schema/ConnectionFactory.wsdl>
- <http://schema.omg.org/specs/PLM/2.0/schema/GeneralConnection.wsdl>
- <http://schema.omg.org/specs/PLM/2.0/schema/MessageConnection.wsdl>
- <http://schema.omg.org/specs/PLM/2.0/schema/PLMServicesAll.wsdl>
- <http://schema.omg.org/specs/PLM/2.0/schema/ComputationalCore.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/Exception.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/GenericQueries.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/Information.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/InformationalCore.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/InformationalModel.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/MessageQueries.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/Parameter.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/PdtnetQueries.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/ProxyQueries.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/SchemaInfo.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/SpecificQueries.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/UtilityQueries.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/xml.xsd>
- <http://schema.omg.org/specs/PLM/2.0/schema/XPathQueries.xsd>

---

\* Normative machine-readable files: dtc/08-09-10

Copyright © 2007, 88solutions  
Copyright © 2007, avanion  
Copyright © 2007, Bosch  
Copyright © 2007, Contact  
Copyright © 2007, DaimlerChrysler AG  
Copyright © 2007, IBM  
Copyright © 2007, International Standard Organization  
Copyright © 2007, IN GmbH  
Copyright © 2007, Magna Steyr  
Copyright © 2009, OMG  
Copyright © 2007, proficiency  
Copyright © 2007, PartMaster GmbH  
Copyright © 2007, PDTec GmbH  
Copyright © 2007, PROSTEP AG  
Copyright © 2007, Prostep iViP Association  
Copyright © 2007, SAP AG  
Copyright © 2007, T-Systems GmbH  
Copyright © 2007, UGS  
Copyright © 2007, valtech  
Copyright © 2007, Volkswagen AG

#### USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

#### LICENSES

The companies listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that: (1) both the copyright notice identified above and this permission notice appear on any copies of this specification; (2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

## PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

## GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.

## DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

## RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 140 Kendrick Street, Needham, MA 02494, U.S.A.

## TRADEMARKS

MDA®, Model Driven Architecture®, UML®, UML Cube logo®, OMG Logo®, CORBA® and XMI® are registered trademarks of the Object Management Group, Inc., and Object Management Group™, OMG™, Unified Modeling Language™, Model Driven Architecture Logo™, Model Driven Architecture Diagram™, CORBA logos™, XMI Logo™, CWM™, CWM Logo™, IIOP™, IMM™, MOF™, OMG Interface Definition Language (IDL)™, and OMG Systems Modeling Language (OMG SysML)™ are trademarks of the Object Management Group. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

## COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

## **OMG's Issue Reporting Procedure**

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page <http://www.omg.org>, under Documents, Report a Bug/Issue (<http://www.omg.org/technology/agreement.htm>).

# Table of Contents

Preface .....	vii
1 Scope .....	1
2 Conformance .....	1
3 Normative References .....	1
4 Terms and Definitions .....	2
5 Symbols .....	2
6 Additional Information .....	3
6.1 Changes to Adopted OMG Specifications .....	3
6.2 How to Read this Specification .....	3
6.3 Accompanying Documents .....	4
6.4 Informative References .....	4
6.5 Acknowledgements .....	5
7 Use Cases (informative) .....	7
7.1 Overview .....	7
7.2 Detailed Use cases .....	7
7.2.1 Export of assembly data .....	7
7.2.2 Import of assembly data .....	10
7.2.3 Authentication/Start-Up of session .....	14
7.2.4 Authorization .....	16
7.2.5 Start node identification .....	17
7.2.6 Browsing down product structure data .....	19
7.2.7 Browsing up product structure data .....	22
7.2.8 Download of product data .....	25
7.2.9 Download meta data including structures .....	26
7.2.10 Download a single digital file .....	29
7.2.11 Generic object query .....	31
7.2.12 Search in design space .....	34
7.2.13 Upload of product data .....	37
7.2.14 Upload a single digital file (simple user interaction) .....	38
7.2.15 Upload meta data including structures .....	39
7.2.16 Change notification .....	41
7.2.17 Display content of subscription list and confirm changes.....	43
7.2.18 Change content of subscription list .....	45
7.2.19 Product Class Identification .....	47
7.2.20 Browsing of Abstract Product Structures .....	48
7.2.21 Browsing of Alternative Solutions within an Abstract Product Structure .....	50

7.2.22 Retrieve Configuration Data within an Abstract Product Structure .....	51
7.2.23 Viewing of Change Management Information .....	53
7.2.24 ECM participant proposal for a change .....	54
7.2.25 ECM participant comments .....	55
7.2.26 ECM participant approval .....	58
7.2.27 ECM participant detailing and comments .....	60
<b>8 Informational Viewpoint PIM (normative) .....</b>	<b>63</b>
8.1 Package PLM_base .....	66
8.1.1 Classes .....	66
8.1.2 Datatypes .....	70
8.2 Package Part_identification .....	71
8.2.1 Classes .....	71
8.3 Package Part_structure .....	76
8.3.1 Classes .....	77
8.3.2 Interfaces .....	88
8.4 Package Document_and_file_management .....	89
8.4.1 Classes .....	91
8.4.2 Interfaces .....	106
8.5 Package Shape_definition_and_transformation .....	108
8.5.1 Classes .....	109
8.5.2 Interfaces .....	126
8.6 Package Classification .....	128
8.6.1 Classes .....	129
8.6.2 Interfaces .....	136
8.7 Package Properties .....	138
8.7.1 Classes .....	141
8.7.2 Interfaces .....	158
8.8 Package Alias_identification .....	159
8.8.1 Classes .....	159
8.8.2 Interfaces .....	160
8.9 Package Authorization .....	161
8.9.1 Classes .....	162
8.9.2 Interfaces .....	177
8.10 Package Configuration_management .....	182
8.10.1 Classes .....	186
8.10.2 Interfaces .....	212
8.11 Package Change_and_work_management .....	215
8.11.1 Classes .....	215
8.11.2 Interfaces .....	225
8.12 Package Process_planning .....	227
8.12.1 Classes .....	228
8.12.2 Interfaces .....	237
8.13 Package Multi_language_support .....	238
8.13.1 Classes .....	239
8.13.2 Interfaces .....	240
8.13.3 Datatypes .....	240
<b>9 Computational Viewpoint (normative) .....</b>	<b>241</b>

9.1 Overview .....	241
9.2 Base Model of the Computational Viewpoint .....	241
9.2.1 PLM_service Interface .....	242
9.2.2 PLM_resource_adapter Class .....	243
9.2.3 PLM_object_factory Interface .....	244
9.2.4 PLM_connection_factory Interface .....	244
9.2.5 PLM_connection Interface .....	245
9.2.6 PLM_general_connection .....	246
9.2.7 PLM_message_connection .....	248
9.2.8 PLM_property_descriptor and PLM_properties_descriptor .....	248
9.2.9 PLM_message_query .....	249
9.2.10 PLM_exception classes .....	249
9.2.11 PLM_query Type .....	251
9.3 Utilities Queries Conformance Point .....	252
9.3.1 Concatenatable_query .....	252
9.3.2 Concatenated_query .....	253
9.3.3 Recursive_query .....	253
9.3.4 Batch_query .....	253
9.3.5 Conditional_query .....	253
9.4 Generic Queries Conformance Point .....	254
9.4.1 Alternative_predicate .....	256
9.4.2 Attribute_equals_predicate .....	256
9.4.3 Attribute_greater_than_predicate .....	256
9.4.4 Attribute_less_than_predicate .....	256
9.4.5 Attribute_pattern_predicate .....	256
9.4.6 Identifier_predicate .....	256
9.4.7 Relationship_predicate .....	257
9.4.8 String_select_predicate .....	257
9.4.9 Type_predicate .....	257
9.4.10 Sort_predicate .....	258
9.5 XPath Queries Conformance Point .....	258
9.6 Proxy Queries Conformance Point .....	259
9.6.1 PLM_proxy_object .....	259
9.6.2 PLM_proxy_feature .....	260
9.6.3 PLM_containment_feature .....	260
9.6.4 PLM_reference_feature .....	260
9.6.5 PLM_proxy_container .....	260
9.6.6 Proxy_query .....	261
9.7 Specific Queries Conformance Point .....	261
9.7.1 Common Queries as base types for specific queries .....	261
9.7.2 Activity_query .....	262
9.7.3 Activity_authorization_query .....	264
9.7.4 Activity_element_query .....	265
9.7.5 Activity_relationship_query .....	266
9.7.6 Activity_resolved_request_query .....	267
9.7.7 Alias_identification_query .....	268
9.7.8 Alternative_solution_query .....	269
9.7.9 Application_context_query .....	270
9.7.10 Approval_relationship_query .....	271
9.7.11 Assembly_component_placement_query .....	272



9.7.12	Associated_activity_query	273
9.7.13	Associated_approval_query	274
9.7.14	Associated_classification_query	275
9.7.15	Associated_date_organization_query	276
9.7.16	Associated_date_time_query	277
9.7.17	Associated_document_query	278
9.7.18	Associated_effectivity_query	279
9.7.19	Associated_file_query	280
9.7.20	Associated_general_classification_query	281
9.7.21	Associated_item_property_query	282
9.7.22	Associated_person_organization_query	284
9.7.23	Associated_process_property_query	285
9.7.24	Associated_project_query	287
9.7.25	Associated_property_query	288
9.7.26	Class_structure_query	288
9.7.27	Complex_product_query	289
9.7.28	Configuration_query	290
9.7.29	Date_organization_assignment_query	291
9.7.30	Design_discipline_item_definition_query	292
9.7.31	Document_classification_query	293
9.7.32	Document_classification_hierarchy_query	294
9.7.33	Document_property_query	295
9.7.34	Document_query	296
9.7.35	Document_representation_query	298
9.7.36	Document_structure_query	298
9.7.37	Document_version_query	299
9.7.38	Effectivity_assignment_query	300
9.7.39	Effectivity_query	301
9.7.40	Event_reference_query	302
9.7.41	External_model_query	304
9.7.42	File_property_query	304
9.7.43	Geometric_model_relationship_query	305
9.7.44	Item_classification_query	307
9.7.45	Item_classification_hierarchy_query	307
9.7.46	Item_definition_instance_relationship_query	308
9.7.47	Item_instance_query	310
9.7.48	Item_query	311
9.7.49	Item_use_query	313
9.7.50	Item_version_query	313
9.7.51	Item_version_relationship_query	314
9.7.52	Object_by_uid_query	315
9.7.53	Objects_by_uids_query	316
9.7.54	Organization_query	317
9.7.55	Organization_relationship_query	318
9.7.56	Person_in_organization_query	319
9.7.57	Person_in_organization_relationship_query	321
9.7.58	Person_organization_assignment_query	322
9.7.59	Person_query	322
9.7.60	Product_class_query	323
9.7.61	Product_identification_query	324
9.7.62	Product_structure_query	325

9.7.63	Project_assignment_query	326
9.7.64	Project_query	327
9.7.65	Simple_property_value_query	328
9.7.66	Work_order_query	329
9.7.67	Work_order_is_controlling_query	332
9.7.68	Work_request_activity_query	332
9.7.69	Work_request_query	333
9.7.70	Work_request_relationship_query	335
9.7.71	Work_request_scope_query	336
9.8	PDTnet Queries Conformance Point	336
9.8.1	General_detail_query	337
9.8.2	Document_detail_query	339
9.8.3	Document_selection_query	339
9.8.4	Document_traversal_query	340
9.8.5	Item_detail_query	341
9.8.6	Item_selection_query	342
9.8.7	Item_traversal_query	343
9.8.8	Product_detail_query	344
9.8.9	Product_selection_query	345
9.8.10	Product_traversal_query	346
9.8.11	Work_order_selection_query	347
9.8.12	Work_order_detail_query	348
9.8.13	Work_request_selection_query	349
9.8.14	Work_request_traversal_query	350
9.8.15	Work_request_detail_query	351
9.9	Message Queries Conformance Point	352
9.9.1	Class Message_by_context_query	353
9.9.2	Class Message_by_properties_query	353

10	WebServices PSM	355
10.1	Overview (informative)	355
10.2	UML Profile for XML Schema (informative)	355
10.2.1	UML Model	355
10.2.2	UML Package	356
10.2.3	UML Classes	358
10.2.4	UML Interfaces	360
10.2.5	UML Attributes, Associations and Compositions	360
10.3	Informational viewpoint with applied stereotypes (normative)	362
10.3.1	Informational Core	362
10.4	Computational viewpoint with applied stereotypes (normative)	363
10.4.1	Computational Core	363
10.4.2	Exceptions	364
10.4.3	Generic Queries	365
10.4.4	Informations	366
10.4.5	Parameters	366
10.4.6	PDTnet Queries	367
10.4.7	Proxy Queries	368
10.4.8	Schema Infos	368
10.4.9	Specific Queries	369
10.4.10	Utility Queries	369

10.4.11 XPath Queries .....	369
10.5 PLM Services Web services (normative) .....	370
10.5.1 Connection Factory .....	370
10.5.2 General Connection .....	371
10.5.3 Message Connection .....	371
10.6 Query Examples (informative) .....	372
10.6.1 Generic Queries Conformance Point Example .....	372
10.6.2 XPath Queries Conformance Point Example .....	372
10.6.3 PDTnet Queries Conformance Point Examples .....	373
10.7 Security Features (informative) .....	373
10.7.1 Introduction .....	373
10.7.2 Goals and Requirements .....	373
10.7.3 Non-Goals .....	374
10.7.4 Message Protection Mechanisms .....	374
10.7.5 Authentication and Authorization Security .....	376
10.7.6 Data-Exchange Security .....	379
10.7.7 Attachments Exchange Security .....	382
Annex A - PIM and PSM for Product Lifecycle Management Services support information .....	383
Index.....	385

# Preface

## About the Object Management Group

### OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at <http://www.omg.org/>.

### OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. A Specifications Catalog is available from the OMG website at:

[http://www.omg.org/technology/documents/spec\\_catalog.htm](http://www.omg.org/technology/documents/spec_catalog.htm)

Specifications within the Catalog are organized by the following categories:

#### OMG Modeling Specifications

- UML
- MOF
- XMI
- CWM
- Profile specifications.

#### OMG Middleware Specifications

- CORBA/IIOP
- IDL/Language Mappings
- Specialized CORBA specifications
- CORBA Component Model (CCM).

## Platform Specific Model and Interface Specifications

- CORBA services
- CORBA facilities
- OMG Domain specifications
- OMG Embedded Intelligence specifications
- OMG Security specifications.

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters  
140 Kendrick Street  
Building A, Suite 300  
Needham, MA 02494  
USA  
Tel: +1-781-444-0404  
Fax: +1-781-444-0320  
Email: [pubs@omg.org](mailto:pubs@omg.org)

Certain OMG specifications are also available as ISO standards. Please consult <http://www.iso.org>

## Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

**Helvetica/Arial - 10 pt. Bold:** OMG Interface Definition Language (OMG IDL) and syntax elements.

**Courier - 10 pt. Bold:** Programming language elements.

Helvetica/Arial - 10 pt: Exceptions

**Note** – Terms that appear in *italics* are defined in the glossary. Italic text also represents the name of a document, specification, or other publication.

## Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to <http://www.omg.org/technology/agreement.htm>.

# 1 Scope

This specification defines a Platform Independent Model (PIM) for Product Lifecycle Management Services V2.0. Its informational model is derived from the ISO 10303-214 STEP model by an EXPRESS-X mapping specification and an EXPRESS-to-XMI mapping process. The functional model is derived from the OMG PDM Enablers V1.3 and to fulfill requirements of the RFP.

The specification defines a Platform Specific Model (PSM) applicable to the Web Services implementation defined by a WSDL specification, with a SOAP Binding, and an XML Schema specification.

# 2 Conformance

An implementation compliant to this specification shall support the XML Schema and Web Services PSM described in this specification and shall be capable to deliver and to consume valid XML documents with respect to the XML Schema defined in the PSM of this specification via service interfaces defined in WSDL in the PSM of this specification.

An implementation compliant to the XML Schema and Web Services PSM described in this specification shall support at least one of the Queries Conformance Points listed below.

A Queries Conformance Point consists of a set of specializations of the type `Query`. This specification defines five Queries Conformance Points:

1. the Generic Queries Conformance Point (see Section 9.4 ),
2. the XPath Queries Conformance Point (see Section 9.5 ),
3. the Specific Queries Conformance Point (see Section 9.7 ),
4. the PDTnet Queries Conformance Point (see Section 9.8 ), and
5. the Messages Queries Conformance Point (see Section 9.9 ).

An implementation shall define the Queries Conformance Points it is realizing.

The Conformance Points are independent from each other.

# 3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

- UML Infrastructure Specification, v2.1.1; OMG document formal/07-02-04
- UML Superstructure Specification, v2.1.1; OMG Document formal/07-02-03
- Meta Object Facility (MOF) Core, v2.0; OMG Document formal/06-01-01
- Meta Object Facility (MOF) 2.0 XMI Mapping Specification, v2.1; OMG Document formal/05-09-01
- ISO 10303-11:1994 Description methods: The EXPRESS language reference manual
- ISO 10303-14:2001 Description methods: The EXPRESS-X language reference manual

- ISO CD 10303-25:2003 Implementation methods: EXPRESS to UML mapping
- ISO TS 10303-28:2002 XML representation for EXPRESS-driven data
- ISO 10303-203:2000 Configuration-controlled mechanical design
- ISO 10303-214:2000 Core data for automotive mechanical design process
- ISO 10303-232:2001 Technical Data Package
- ISO/IEC 10746: Reference Model for Object Distributed Computing (RM/ODP)

## 4 Terms and Definitions

The following terms are defined in reference specifications that are used in this specification.

**Conformance Class:** defined in ISO 10303 (STEP) Part 1 as a subset of the informational model defined in Application Protocol (AP) for which conformance may be claimed.

**Data Exchange Tool:** Application to realize EDI based on a choice of appropriate and agreed protocols.

## 5 Symbols

The following abbreviations are from references to documents used in this specification.

**AIM:** Application Interpreted Model, defined in ISO 10303 (STEP) Part 1 as an information model that uses the integrated resources necessary to satisfy the information requirements and constraints of an application reference model (ARM), within an application protocol. (AP)

**ARM:** Application Resource Model, defined in ISO 10303 (STEP) Part 1 as an information model that describes the information requirements and constraints of a specific application context.

**AP:** Application Protocol, defined in ISO 10303 (STEP) Part 1 as a part of this International Standard that specifies an application interpreted model satisfying the scope and information requirements for a specific application.

**CC:** Conformance Class, see terms defined in Section 4.

**CC21:** Conformance Class 21 is defined in ISO 10303 (STEP) AP214, 3<sup>rd</sup> edition, comprising information on product structure and configuration management data.

**ECM:** Engineering Change Management

**ECO:** Engineering Change Order

**ECR:** Engineering Change Request

**EDI:** electronic data interchange, automatic, one-way, asynchronous send of structured data between enterprises and application systems like ERP or PPS.

**ENGDAT:** Engineering Data Message, industry standard for EDI targeted at the exchange of CAD data between multiple enterprises. ENGDAT is defined in the Odette standard ODG11ED9206 and as recommendation 4951 of VDA. It defines a structured meta data format containing sender and receiver information, purpose, status and approval data. ENGDAT is executed on top of the transport protocol OFTP.

**ERP:** Enterprise resource planning application systems integrates enterprise-wide information of an organization and support processes of that organization. Examples of modules in an ERP that formerly would have been stand-alone applications include: Manufacturing, Supply Chain, Financials, Customer Relationship Management, Human Resources, and Warehouse Management.

**Odette:** Organization for Data exchange by Tele Transmission in Europe. UK based non-profit organization of automotive manufacturers and suppliers standardizing on logistics, EDI, and exchange of design data.

**OEM:**

**OFTP:** Odette File Transport Protocol corresponds to recommendation 4914/2 of VDA implemented on top of ISDN, X.25 or TCP/IP transport protocols.

**PDM:** Product data management is a concept to define, represent, and present data and documents related to the product development process. It implements processes and manages status changes of the product information.

**PDTnet:**

**PLM:** Product Lifecycle Management comprises all aspects of data management across the life cycle of a product and integrates them in a unified data base. Ideally all departments and systems related to the product refer to this central data base including the PPS and ERP systems, computer aided design (CAD), computer aided engineering (CAE), computer aided manufacturing (CAM) but also controlling, accounting, after-sales, customer care, and service.

**PPS:** Production planning systems support resource planning and control and the related information management. They can be seen as a subset of ERP systems that usually extend PPS by human resource and finance management facilities.

**STEP:** Standard for the Exchange of product model data ISO 10303 is an international standard for the computer-interpretable representation and exchange of product data. ISO10303 is organized as a series of parts, each published separately.

**STEP PDM file:** A data exchange file formatted according to STEP 10303 Part 21 and storing information related to PDM data. The content of such a file is usually compliant to one of the Conformance Classes of a STEP AP.

**VDA** (in German: Verband der Automobilindustrie) German automotive industry association. Member of the European association Odette.

## 6 Additional Information

### 6.1 Changes to Adopted OMG Specifications

This specification completely replaces the PLM Services 1.0 specification.

### 6.2 How to Read this Specification

The rest of this document contains the technical content of this specification.

Although the chapters are organized in a logical manner and can be read sequentially, this is a reference specification and is intended to be read in a non-sequential manner. Consequently, extensive cross-references are provided to facilitate browsing and search.



### 6.3 Accompanying Documents

The machine readable files accompanying this specification are provided in an extra structured archive document. The nature, the format, and a brief description of the provided files are given in Table 6.1.

**Table 6.1 - Accompanying documents**

Filename	Nature	Format	Description
<b>express/</b>			
./AIM2PIMEquivalence.exx	informative	STEP EXPRESS-X	Mapping of the relevant subset of AIM representation of STEP AP214 CC21
./PIM_Equivalence_Schema.exp	informative	STEP EXPRESS	ARM representation of STEP AP214 CC21 model modified to be mapped to the UML PIM
<b>xmi</b>			
./PIM_Informational Viewpoint.xmi	normative	XMI 2.1	Platform independent infor-mational model of PLM Services 2.0
./PIM_ComputationalViewpoint.xmi	normative	XMI 2.1	Platform independent computational model of PLM Services 2.0
./PSM_XSDSchema.xmi	normative	XMI 2.1	XML Schema Platform specific model of PLM Services 2.0
./PSM_WSDLnamespace.xmi	normative	XMI 2.1	WSDL Platform specific computational model of PLM Services 2.0
<b>webservices/</b>			
./xml	normative	XML Schema	XML Schema representation
./wsdl	normative	WSDL	WSDL representation

### 6.4 Informative References

The PLM V2.0 Specification refers to the following material:

1. [WS-SECURITY] WS-Security Core Specification 1.1 OASIS
2. [TOKEN] Web Services Security Username Token Profile 1.1
3. [SEC] Web Services Security: SOAP Message Security 1.1
4. [ENC]XML-Encryption specification at the W3C organization on <http://www.w3.org/Encryption/2001/>
5. [SIGNATURE] XML- Signature specification at the W3C organization on <http://www.w3.org/TR/xmlsig-core/>
6. [VDA] Recommendation for Engineering Change Management Part 1: Engineering Change request (ECR), VDA recommendation 4965, VDA 2006.
7. [OASIS][http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=security](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security)
8. [PDMSchema]

9. [PDTnet] Project Information, available at <http://www.prostep.org/en/infopool/pdtnet>.
10. [SOAP] SOAP Version 1.2, June 2003, available at [http://www.w3.org/TR/2003/REC-soap12-part\[012\]-20030624](http://www.w3.org/TR/2003/REC-soap12-part[012]-20030624)
11. [WSDL] Web Services Description Language (WSDL) 1.1, March 2001, available at <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>
12. [WS-I] WebServices interoperability Basic Profile version 1.1, <http://www.ws-i.org/Profiles/BasicProfile-1.1.html>
13. [XPath] XML Path Language (XPath) 1.0, November 1999, available at <http://www.w3.org/TR/1999/REC-xpath-19991116>

## 6.5 Acknowledgements

The PLM V2.0 Specification contained in this document is the result of a joint effort from the following organizations:

- 88solutions.
- avanion
- Robert Bosch GmbH
- Contact GmbH
- DaimlerChrysler AG
- IBM
- IN GmbH
- Magna Steyr
- PartMaster GmbH
- PDTec GmbH
- proficiency
- PROSTEP AG
- SAP AG
- T-Systems International GmbH
- UGS
- valtech
- Volkswagen AG



# 7 Use Cases (informative)

This chapter describes the use cases supported by the specification. It utilizes use case, state machine, and sequence diagrams as informative illustrations.

In general, use case diagrams are chosen if the process does not include control structures like alternatives, the relation of the single process steps is informal, and each individual process can be seen as an isolated asset. State machine diagrams are used to express more formally specified control flows including transition states. Usually, not all of the process steps are useful without that context. Sequence diagrams are used if it is necessary or useful to illustrate a more detailed interaction of identified participants and systems of the use case.

## 7.1 Overview

The Information Model of the PLM Service is based on the STEP PDM Schema [PDMSchema] and extended by relevant subsets of STEP ISO 10303-214:2000, especially the Configuration Management modeling parts. The scope is chosen such that the informational model covers the complete Conformance Class CC21.

The selected scope of the Information Model is a superset of the model necessary to fulfill the requirement analysis in the PDTnet project [PDTnet]. The use cases identified in this industrial project of European automotive companies are given in brief in Section 7.2. The information model represents the PLM reference model in the informational viewpoint and is described in Chapter 8.

## 7.2 Detailed Use cases

This section describes the use cases that are subject to the PLM services specification. They are categorized according to the requirement analysis resulting from the PDTnet project [PDTnet]. They are documented in this section, and may be extended continuously.

The scope of the use cases is defined supporting an online PLM integration scenario that is characterized by a data access on remote systems using internet functionality and technology. This integration does not provide a real online integration, but due to the usage of data streaming techniques and due to the possibility of an immediate reply by a system it comes near to it. It is assumed that a neutral PLM client provides access to different PLM data providers (these are usually different PLM systems in different companies).

### 7.2.1 Export of assembly data

Export of product data (meta data and geometry) of assemblies and parts from one partner to another partner via exchange of ENGDAT packages (STEP PDM files, CAD files).

#### 7.2.1.1 Owner of the use case

This use case was defined by Work Group 1 of the PDTnet project.

#### 7.2.1.2 Process purpose

Export of product data that consists of meta data and geometry information of assemblies and its components from one partner to another partner via exchange of ENGDAT formatted packages. The ENGDAT message contains the STEP PDM files and (optionally) the CAD files, in native or neutral format.

### 7.2.1.3 Partner role descriptions

**Table 7.1 - Roles for export of assembly data**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
User	Party that selects and processes product data to be exported.	Person
PLM System	Party that provides the relevant product data and functionality for product data management. This is usually a company's PLM system, which also can be extended by a tool that provides extended STEP processor functionality.	System
Data Exchange (DE) Tool	System that provides communication with a network and functionality to automatically process and pack/unpack file packages (usually ENGDAT-based).	System

### 7.2.1.4 Process definition

The process steps are:

1. User selects parts/documents/CAD models (using the functionality of the PLM system):
  - Selection of root/top level assembly by assembly (version) number
  - Selection of affected sub-assemblies or parts (could be controlled by a context or specific algorithm)
  - Exclusion of elements from selected set is possible
2. PLM system generates STEP PDM file:
  - Passing assembly structure tree and collecting transformation matrices (if appropriate)
  - Generating STEP PDM file
3. User selects addressee of data (using the DE tool or PLM tool)
4. Download of digital files from PLM system
5. DE Tool generates ENGDAT package including message abstract, STEP PDM file(s) and digital files (CAD/CC2 files, etc.)
6. DE Tool initiates sending of ENGDAT message

The order of the process steps could differ depending on specific user requirements and system scenario. Examples for possible alternative process step orders are:

a): 1. - 3. - 4. - 2. - 5. - 6.

b): 3 - 1. - 2. - 4. - 5. - 6.

### 7.2.1.5 Process diagram

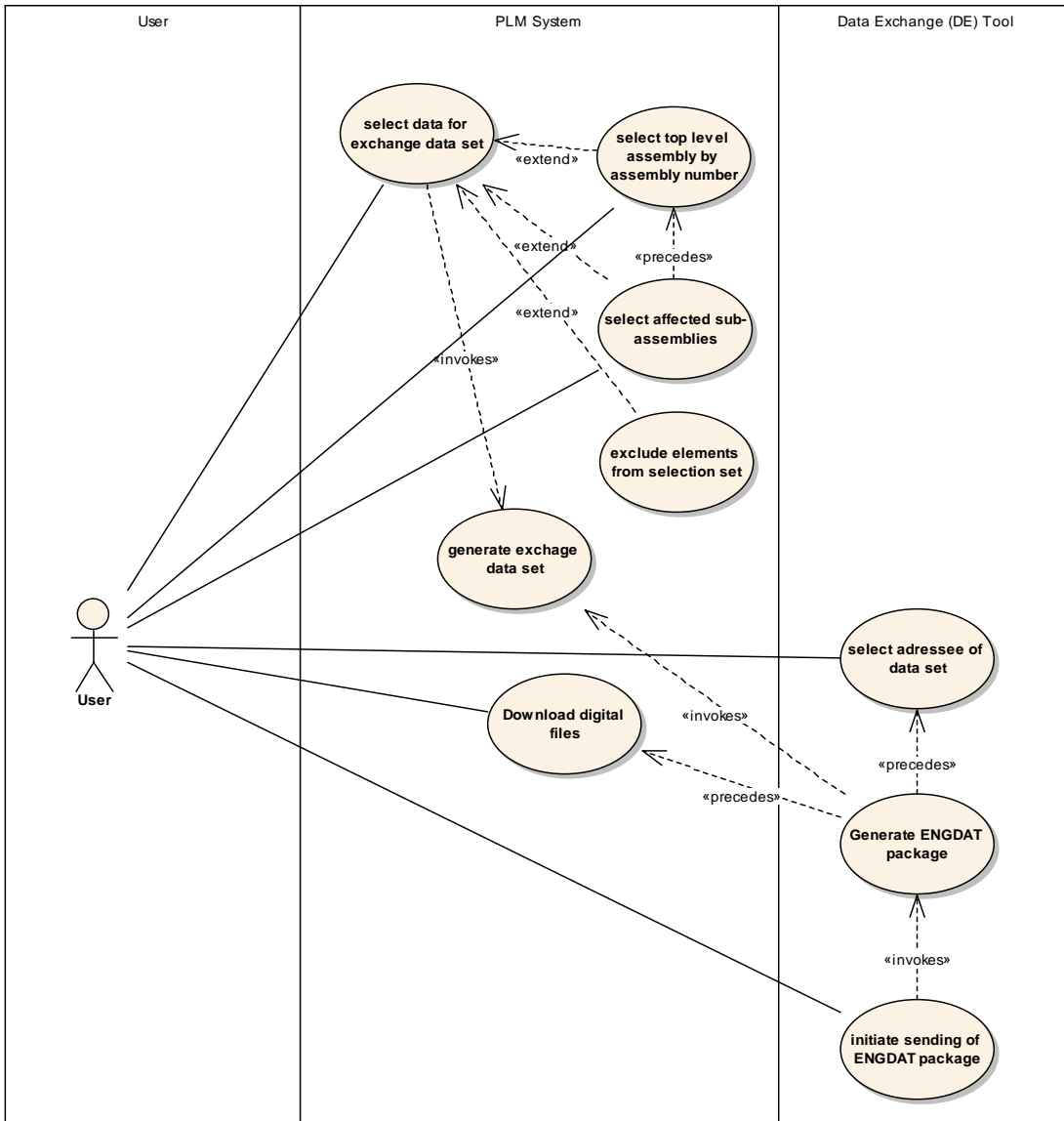


Figure 7.1 - Use case diagram export of assembly data

### 7.2.1.6 Process start and end states

Start state / precondition:

The user knows the assembly/part identifiers and digital file (CAD model) identifiers that are supposed to be exported. At least the identifier of an assembly, which serves as an entry node, is provided. Additionally, a specific “context handle” is known (project, change status, work order, etc.) is known.

Alternative a): Depending on the user environment also a top-level document ID can be the entry node to a structure.

Alternative b): A top-level part and a specific configuration, which controls the way of the expansion of the tree (sub-parts, kind of documents,...), is known.

End state / post condition E1 (Success):

An ENGDAT package including the STEP PDM file and all selected digital files were successfully sent to the addressee.

End state / post condition E2 (Failure):

DE Tool delivers failure notification/report to user. The reasons can be:

- The STEP processor failed.
- The download of files from the PLM system failed.
- The DE Tool failed.

### **7.2.1.7 Constraints and assertions**

Currently the number of STEP files included in one ENGDAT package is recommended to be restricted to one (VDA). Nevertheless, the intention is to allow more than one STEP file per ENGDAT message, see Section 7.2.1.9.

### **7.2.1.8 Relevant data**

- Documents/digital files (CAD files), see Section 8.4.
- Document meta data, see Section 8.4.
- Assembly/part master data, see Section 8.2.
- Assembly structure data (including transformation data), see Section 8.3.

### **7.2.1.9 Topics under discussion / Remarks**

Currently no engineering change information is included in the STEP PDM file.

Should more than one STEP file be allowed in an ENGDAT message?

### **7.2.1.10 Realization with this specification**

This specification supports the functionality to realize selection of start nodes and traversing product structures. Especially the PDTnet queries conformance point provides the necessary methods, see Section 9.8.6 and Section 9.8.7. Retrieving the associated documents is realized by methods as described in Section 9.8.3.

## **7.2.2 Import of assembly data**

Import of product data (meta data and geometry) of assemblies and parts from one partner to another via exchange of ENGDAT packages (STEP PDM files, CAD files).

### **7.2.2.1 Owner of the use case**

This use case was defined by Work Group 1 of the PDTnet project.

### 7.2.2.2 Process purpose

Import of product data that consists of meta data and geometry information of assemblies and its components from one partner to another partner via exchange of ENGDAT formatted packages. The ENGDAT message contains the STEP PDM files and (optionally) the CAD files, in native or neutral format.

### 7.2.2.3 Partner role descriptions

**Table 7.2 - Roles for import of assembly data**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
User	Party that processes product data that has been imported.	Person
PLM System	Party that provides the relevant product data and functionality for product data management. This is usually a company's PLM system, which also can be extended by a tool that provides extended STEP processor functionality.	System
Data Exchange (DE) Tool	System that provides communication with a network and functionality to automatically process and pack/unpack file packages (usually ENGDAT-based).	System

### 7.2.2.4 Process definition

The process steps are:

1. The DE tool receives an ENGDAT package.
2. The DE tool unpacks the ENGDAT package and stores STEP PDM and CAD files in defined directories (routing).
3. The PLM system evaluates the received STEP PDM file and displays the included data (assembly data, part data, CAD file meta data) and, optionally, generates an analysis report (comparison of existing data and data to be imported). This step can be initiated by the user or by the DE tool (if it is appropriately integrated). See 7.2.2.8, Topics under discussion.
4. The user manually processes the data and integrates it into the database of the PLM system or, alternatively, no manual interaction is done. See 7.2.2.8, Topics under discussion.

The DE tool can notify the user of the import process in different ways, e.g., via e-Mail, via PLM system message, a.s.o.



### 7.2.2.5 Process diagram

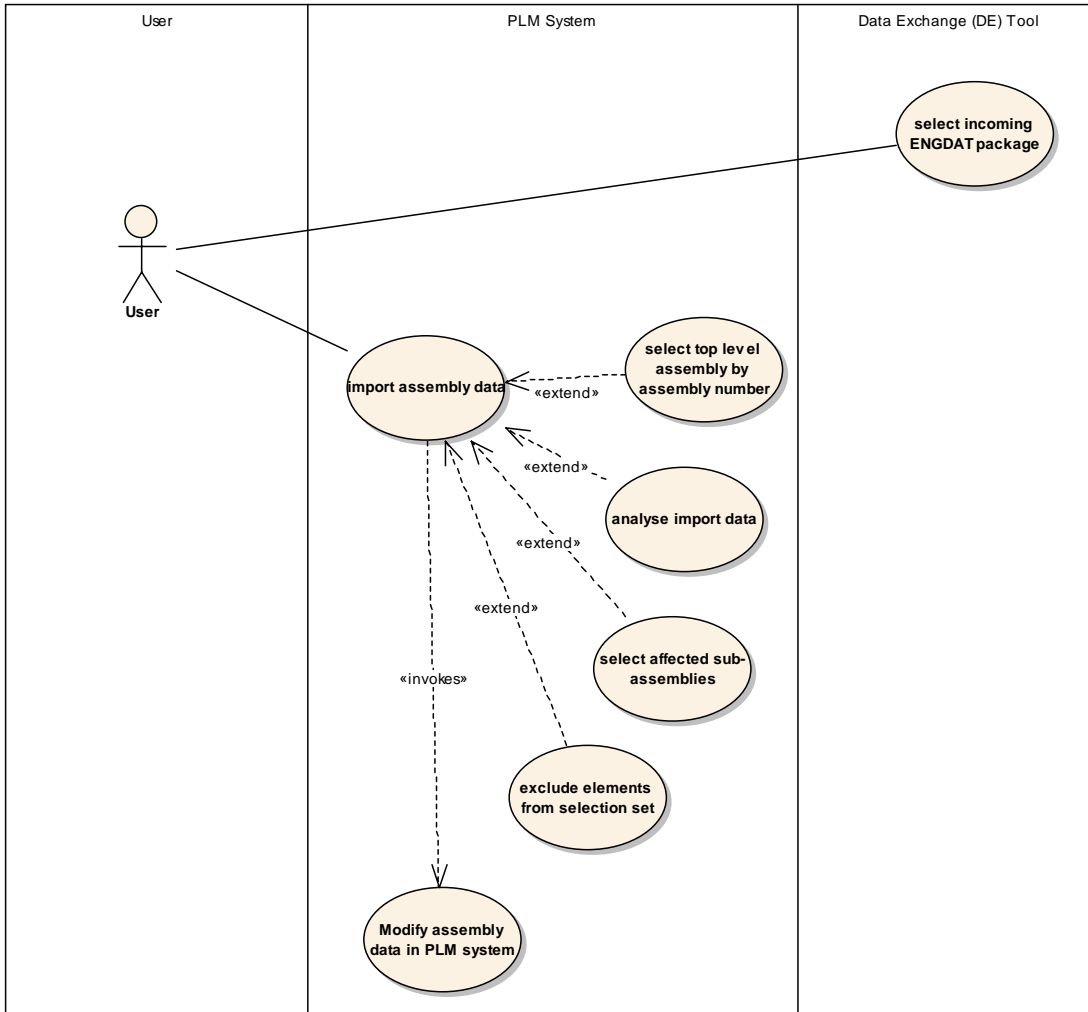


Figure 7.2 - Use case diagram import of assembly data

### 7.2.2.6 Process start and end states

Start state / precondition:

- An ENGDAT package including a STEP PDM file and one or more digital files (CAD files,...) has been received successfully. This means:
  - The ENGDAT message contains the expected correct data.
  - No inconsistencies between STEP file and references to digital files exist. See 7.2.2.8, Topics under discussion.
  - User selected the mode for import (update, create, etc.).

End state / post condition E1 (Success):

The received PDM data has been successfully integrated in the PLM systems' database.

The received CAD files have been successfully stored in the defined storage areas.

Partial incorporation of data in the PLM system, if the user allowed it.

End state / post condition E2 (Failure):

The process results in a failure message. A failure can occur due to the following reasons:

- The ENGDAT message contains errors and cannot be processed correctly.
- The STEP PDM file contains errors and cannot be processed correctly (syntactically, semantically, e.g., STEP PDM Schema, etc.).
- The loading process into the PLM system caused errors.

#### **7.2.2.7 Relevant data**

- Documents/digital files (CAD files), see Section 8.4
- Document meta data, see Section 8.4
- Assembly/part master data, see Section 8.2
- Assembly structure data (including transformation data), see Section 8.3

#### **7.2.2.8 Topics under discussion**

- Who or which system checks whether the STEP file and the references to digital files included in an ENGDAT message are consistent? Definition of a separate use case?
- On supplier's side: How to handle product/document meta data, that is not managed by the own PLM system (or no PLM system exists) but that has to be re-exported to the OEM?
- Export of version/status information for re-exported assemblies/parts could be discussed. At the moment no version/status information is used.
- The CATIA model name must not be changed by a supplier.
- On supplier's side: How to associate product data identified by OEM identifiers to product data in the own PLM system?
- On supplier's side: How to manage different assembly structures?

#### **7.2.2.9 Realization with this specification**

This specification supports the functionality to realize selection of start nodes for insertion and traversing and selecting product structures. Especially, the PDTnet queries conformance point provides the necessary methods, see Section 9.8.6 and Section 9.8.7. This functionality is applied both to the assembly data in the exchange package and to the data already stored in the PLM system. Comparison and consolidating of both data sets is not part of this specification. Updating the selected data is performed by the write methods defined in the services as given in Section 9.2.6 or Section 9.2.7.

### 7.2.3 Authentication/Start-Up of session

This process allows a user to be authenticated via a PLM client by one or more PLM server(s).

#### 7.2.3.1 Owner of the use case

This use case was defined by the Work Group 2 of the PDTnet project.

#### 7.2.3.2 Process purpose

This process allows a user to be authenticated via the PLM client by one or more PLM server(s).

#### 7.2.3.3 Partner role descriptions

**Table 7.3 - Roles for authentication and start-up of session**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
User	Party that wishes to log in a remote PLM server. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PLM data. This is usually a company's PLM system that acts as a server.	System

#### 7.2.3.4 Process definition

This use case includes the initiation of the connection between PLM client and PLM server, the authentication and personalization of the user. This use case usually initiates all following communication and data transfer between a user, using the PLM client, and a PLM server (also called "site").

Two alternative authentication processes are possible, which can also be combined:

1. The first attempt to access a remote PLM server will automatically start the authentication process.
2. The user explicitly starts a login procedure to authenticate in one or more PLM server(s) in the beginning of a session.

The following accesses to specific PDM data will be validated within the use case "Authorization."

### 7.2.3.5 Process diagram

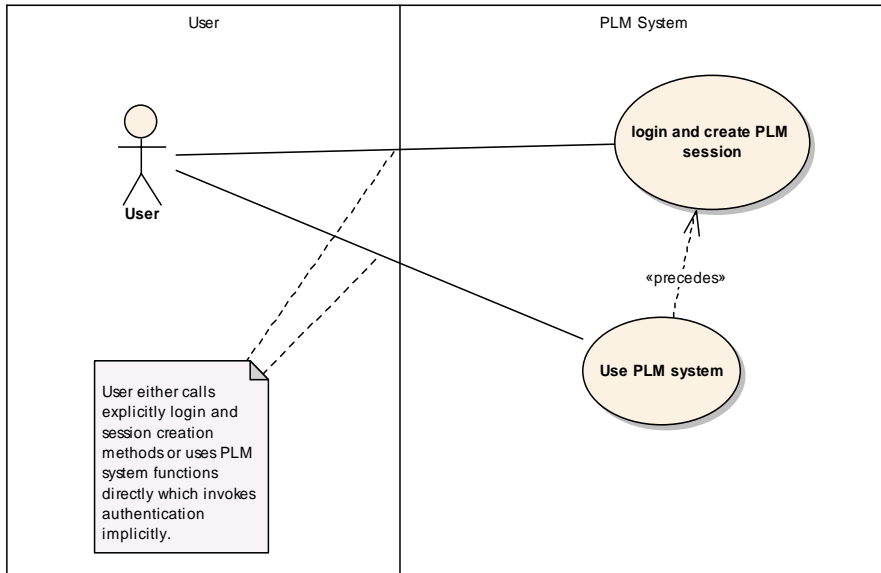


Figure 7.3 - Use case diagram authentication and start-up of session

### 7.2.3.6 Process start and end states

Start state S1:

- The user owns a user name and a password valid for a certain PLM server (site).
- The client provides the necessary site information for the network connection.
- The user knows a valid development project to be authorized to access product data on the PLM server.
- The PLM server provides an authentication service based on user, password, and session.

End state E1 (Success):

- The user is successfully logged in and, optionally, the PLM server returns a session id.

End state E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - The user is not allowed to access the PLM server (return message: "Permission denied").
  - The PLM server itself is not available.

### 7.2.3.7 Constraints and assertions

A development project defines a project in which persons work together on a certain set of product data. A development project can be a car/vehicle project, a module development project, etc.

### 7.2.3.8 Relevant data

User name, password, development project, site information (PLM server system), optional: session id.

### 7.2.3.9 Realization with this specification

This specification recommends two methods to access PLM system functionality:

- using PLM\_resource\_adapter functionality (see Section 9.2.2) to obtain a valid PLM system connection reference via the managed PLM\_connection\_factory (see Section 9.2.4), or
- directly using an appropriate PLM\_connection service instance.

This specification does not define explicit methods to login to a particular system but encourages implementations to use the platform specific security specifications instead - see Section 10.7.4. The handling of additional properties or options to configure access to a PLM system, like development project, site information and others shall be subject to handling properties as defined in the PLM\_service interface, see Section 9.2.1.

## 7.2.4 Authorization

This process validates the access rights of a specific user (designer, group, department, company) to access specific product data on a PLM server.

### 7.2.4.1 Owner of the use case

This use case was defined by the Work Group 1 of the PDTnet project.

### 7.2.4.2 Process purpose

This process validates the access rights of a specific user (designer, group, department, company) to access specific product data on a PLM server.

### 7.2.4.3 Partner role descriptions

**Table 7.4 - Roles for authorization**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
User	Party that wishes to access PDM data on a remote PLM server. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.4.4 Process definition

This use case describes the authorization process of a user who attempts to request specific product data on a PLM server. It is used by all other use cases (e.g., when extracting product structure trees). The actual process description is dependent on the authorization mechanisms provided by the PLM server.

#### 7.2.4.5 Process start and end states

Start state S1:

- A previous authentication process was successful (e.g., by given session id).
- The PLM server provides an authorization service based on user, password, and session related to specific product data elements. Additionally, the association of product data elements to a development project has to be supported.
- Specific product data that is requested by a user.

End state E1 (Success):

- The user is identified to have the appropriate rights to access the requested product data. The calling process is enabled to provide the product data to the user.

End state E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reason:
  - The user is not allowed to access the requested product data. Since it could be intended to keep the existence of the requested data completely secret, the user should not get the information “Access denied.” Instead, he should get a failure message like “Data not found.”

#### 7.2.4.6 Constraints and assertions

The PLM server provides an authorization service based on user, password, and session related to specific product data elements. Additionally, the association of product data elements to a development project has to be supported. The detailed mechanisms of authorizing specific users to access specific product data elements depend on the PLM server’s internal authorization features and company specific customizing.

Specific assertions:

- The PLM server manages the association of user/development project to a specific server internal role concept.
- The general role “owner” is provided having all rights for the owned data objects.
- Defined access rights to all other (not owned) data objects are: View, Download, Write, Create.

#### 7.2.4.7 Relevant data

User name, password, development project, optional: session id

- Requested product data, see Section 8.3

#### 7.2.4.8 Realization with this specification

PLM Services does not define its own authorization mechanisms but encourages PSM to use platform specific solutions - see Section 10.7.

### 7.2.5 Start node identification

Identify the start node of a product structure to enable browsing in the product structure.

### 7.2.5.1 Owner of the use case

This use case was defined by the Work Group 2 of the PDTnet project.

### 7.2.5.2 Process purpose

Identify the start node of a product structure to enable browsing in the product structure.

### 7.2.5.3 Partner role descriptions

**Table 7.5 - Roles for start node identification**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.5.4 Process definition

This use case defines the process of identifying the start node of a product structure in a PLM server. The end state / post condition of the use case is the precondition for the start of the use cases “Browsing down/up product structure data.”

The process steps are:

1. User enters ID (part number and optionally part version number) or Wild Card ("\*" for “all”).
2. PLM client submits search request - Exception: The PLM server does not respond.
3. PLM server receives ID or Wildcard and triggers search in PLM system - Exception: The connection between PLM client and PLM server is down.
4. PLM system executes query in its database - Exceptions: Database is not available, no data found, user is not authorized to access the data, etc.
5. PLM server returns start node and list of views.
6. PLM client displays list of start nodes.

### 7.2.5.5 Process start and end states

Start state / precondition S1:

The user is correctly logged in, connected to the server, positively identified, and authorized.

- The service is available.
- The user enters an ID (“Sachnummer” etc.) or wildcard for the structure start node.

End state / post condition E1 (Success):

- List of product structure nodes including their possible views / configurations.

End state / post condition E2 (Failure):

- In case of missing authorization: Exception, message: “No items found or access denied.”

#### **7.2.5.6 Relevant data**

- Product structure data, see Section 8.3.

#### **7.2.5.7 Topics under discussion**

The user should be able to enter either internal or external part master ids (“Alias-Query”).

#### **7.2.5.8 Realization with this specification**

This functionality is provided by the various query functionality within the Generic Queries Conformance Point, the XPath Conformance Point, the Specific Conformance Point, and the PDTnet Conformance Point. See Section 10.6 for examples utilizing the functions in these conformance points.

### **7.2.6 Browsing down product structure data**

This process allows a user starting with the product structure to get a view on all product structure relevant data including document (structure) data that is relevant for this specific user or a specific project, independently of the provider of the data.

#### **7.2.6.1 Owner of the use case**

This use case was defined by the Work Group 2 of the PDTnet project.

#### **7.2.6.2 Process purpose**

This process allows a user starting with the product structure to get a view on all product structure relevant data including document (structure) data that is relevant for this specific user or a specific project, independently of the provider of the data.



### 7.2.6.3 Partner role descriptions

**Table 7.6 - Roles for browsing down product structure data**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.6.4 Process definition

This use case includes the browsing of product structure data down a product structure, basic part classification data and associated document meta data. For browsing up a product structure (“where used” query), a separate use case is defined.

The following requirements are defined:

Multiple views on the product structure have to be supported, e.g., lead view, supplier's assembly structure, spare part structure, second tier supplier's view, etc.

- The relationship between different base classification data has to be handled (customer's and supplier's data).
- The assignment of structure and classification data to documents has to be consistent and browsing documents must result always in displaying identical information.
- The user defines a set of parameters (filter information), that specifies characteristics of the desired structure nodes in detail. Filtering the data will be defined as a separate use case “PLM filter.”
- Browsing in different PLM server systems has to be supported. This means, the change of a server site has to be possible (“Multi-site support”) when the user selects a structure node, which links to a supplied item provided by another PLM server. This enables the user to browse into a sub-structure of the development partner (e.g., OEM user browses into sub-structure of supplier or vice versa) and to see the information consistently in one single structure tree. The concept for this mechanism is the following:
  - Reference tables connecting the OEM part identifiers to the supplier part identifiers (“alias identifiers”) are managed by the PLM servers, containing for each exchange node:
    - Own part id (item\_version to be supported)
    - Corresponding alias id on PLM server of partner
    - Unique identifier for partner PLM server site: harmonized organization ID (e.g., “bmw.de”).
- An additional reference table for the association of organization id and URL (server site connection) is provided on the PLM client site.

The process steps are:

1. PLM client sends a query for substructure specified by the user to the PLM server:
  - a. In case of the structure node being a “supplied item,” i.e., the selected structure node represents an alias identifier:
    - Client retrieves alias site connection information (URL) from reference table.

- Client asks user for password for alias site (only in case of first request to this site).
- Client performs Login, Start node query on alias server site using current development project.

Steps repeated by PLM server for each product structure node in the scope of the query:

2. Check authorization regarding requested data - Exception: Access denied (PLM server).
3. Collect requested data within PLM server.

End of repeated steps.

4. PLM server sends data to PLM client.
5. Display structure and items in PLM client.

### 7.2.6.5 Process diagram

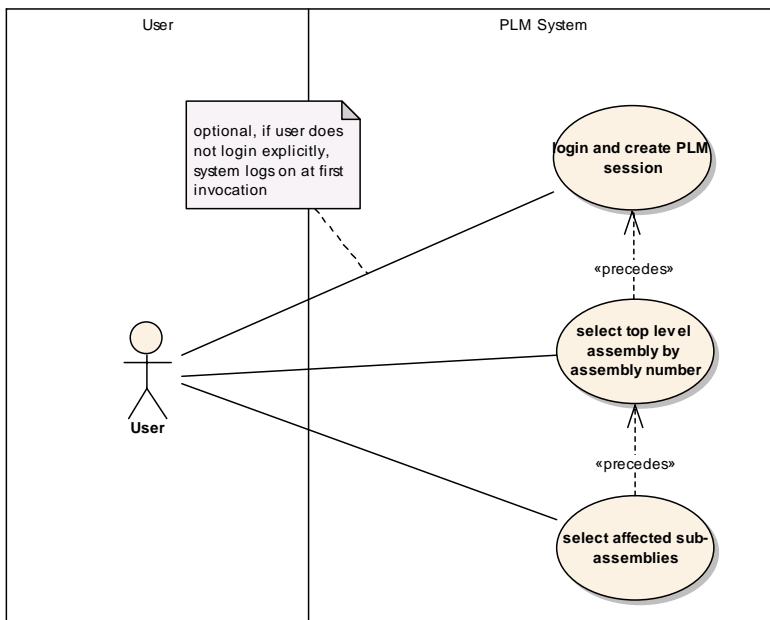


Figure 7.4 - Use case diagram browsing down product structure data

### 7.2.6.6 Process start and end states

Start state / precondition S1:

- A specific development project is defined, which itself defines certain items of product data (e.g., assemblies, parts, documents), that will be subject to change or creation during the project's life time. These items are identified by identifiers.
- The end state / post condition of use case "Start node identification" or one of the children of the start node.
- The user is correctly logged in and authorized to access the requested information.
- The level of depth down the start node / current node is defined (default: 1 level down the current node).

- The necessary filter information is defined, i.e., the result of the use case “PDM filter” is provided.

End state / post condition E1 (Success):

- The process results in a filtered list or a structure tree containing at least the identifiers of product data items, and additional information about the items (e.g., URLs to documents or additional item information to be downloaded).

End state / post condition E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - The user is not authorized to access the data.
  - The requested data is not available on the PLM server.

### **7.2.6.7 Constraints and assertions**

If process step 2 leads to an exception regarding a specific structure node, the whole process must continue. The structure node affected by the exception is not included in the collected data set.

### **7.2.6.8 Relevant data**

Product structure data

- Basic part classification data, see Section 8.2
- Document meta data, see Section 8.4

### **7.2.6.9 Realization with this specification**

This specification supports the query for a particular start node, part, or document. This query may contain filters for specific attribute values expressed by location\_path, attribute, and predicates as described in Section 9.4. The exception handling is described in Section 9.2.10.

## **7.2.7 Browsing up product structure data**

This process allows a user starting with the product structure to get a view on all product structure relevant data including document (structure) data that is relevant for this specific user or a specific project, independently of the provider of the data.

### **7.2.7.1 Owner of the use case**

This use case was defined by the Work Group 1 of the PDTnet project.

### **7.2.7.2 Process purpose**

This process allows a user starting with a specific product structure node to get a view on all relevant product structure nodes in which this specific node is included (“Where used” query). For browsing down a product structure, a separate use case is defined.

### 7.2.7.3 Partner role descriptions

Table 7.7 - Roles for browsing up product structure data

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.7.4 Process definition

This use case includes the browsing of product structure data up a product structure (“where used” query).

The following requirements are defined.

Multiple views on the product structure have to be supported, e.g., lead view, supplier's assembly structure, spare part structure, second tier supplier's view, etc.

- The user defines a set of parameters (filter information), that specifies characteristics of the desired structure nodes in detail.

The process steps are:

1. PLM client sends a query for “where used” nodes specified by the user to the PLM server.

Steps repeated by PLM server for each product structure node in the scope of the query:

2. Check authorization regarding requested data - Exception: Access denied (PLM server)
3. Collect requested data within PLM server

End of repeated steps.

4. PLM server sends data to PLM client
5. Display structure and items in PLM client. The way of presentation and needed interaction have to be defined by the application projects.

### 7.2.7.5 Process diagram

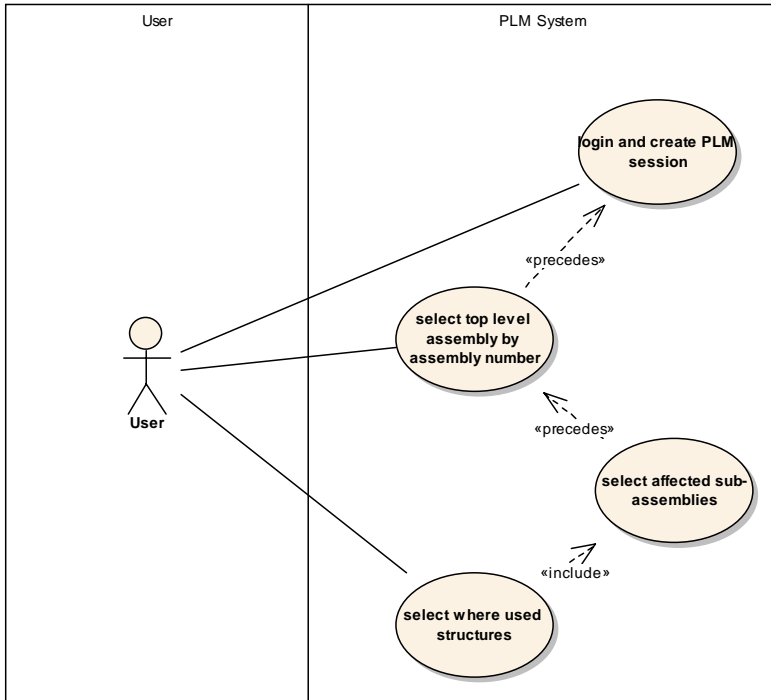


Figure 7.5 - Use case diagram browsing up product structure

### 7.2.7.6 Process start and end states

Start state / precondition S1:

A specific engineering development project is defined, which itself defines certain items of product data (e.g., assemblies, parts, documents) that will be subject to change or creation during the project's life time. These items are identified by identifiers.

- The end state / post condition of use case “Start node identification” or one of the children of the start node, that means item or item\_version. Item\_version is optionally in order to enable the access of versioning information starting from the part number. Additionally, the single\_instance can be identified (maybe by user interaction). This is “nice to have” in general, but required as precondition for a “Search in design space” functionality.
- The user is correctly logged in and authorized to access the requested information.
- The level of depth up the start node / current node is defined and restricted to direct parent or root node (default: direct parent node).

The necessary filter information is defined, i.e., the result of the use case “PLM filter” is provided.

End state / post condition E1 (Success):

- The process results in a filtered list or a structure tree containing only identifiers of product data items (root nodes or direct parent nodes). Only structure nodes that the user is authorized to see are included.

End state / post condition E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - The user is not authorized to access the data.
  - The requested data is not available on the PLM server.

### **7.2.7.7 Constraints and assertions**

Whenever the PLM System is providing a `single_instance` concept, the start node used may be the `single_instance`. If the `single_instance` is used, there is no necessity for repeating process steps 2 and 3.

- The level of depth up the start node / current node is defined and restricted to direct parent or root node (default: direct parent node).
- Exactly one root node exists for one development project.
- Need of unique filter, that displays the root node only once.
- Within the Client GUI the change to one of the resulting development project (in case of a result list containing root nodes) should be possible.

### **7.2.7.8 Relevant data**

- Product structure data, see Section 8.3.

### **7.2.7.9 Realization with this specification**

This specification supports the query for a particular part start node, see Section 9.8.6. This query may contain filters for specific attribute values expressed by `location_path`, `attribute`, and `predicates` as described in Section 9.4. The exception handling is described in Section 9.2.10.

## **7.2.8 Download of product data**

### **7.2.8.1 Owner of the use case**

This use case was defined by the Work Group 1 of the PDTnet project.

### **7.2.8.2 Process purpose**

This use case has to be described under consideration of two main criteria:

What product data is to be downloaded?

- Download of a single digital file: either geometry (CATIA, STEP) or other binary formats (e.g., TIFF).
- Download of a set of digital files.
- Download of structures including optionally digital files.
- Download of product meta data of a (structure) node.

How is the product data to be downloaded?

- Using online download: via HTTP, only for available documents - no conversion functionality provided.
- Using offline download (e.g., via OFTP).

Due to these distinctions the use case “Download of product data” is divided into two use cases, which are described in Section 7.2.9 and Section 7.2.10.

### 7.2.9 Download meta data including structures

This use case allows the user to identify meta data including structures that he wants to store in a local file system, or that he wants to import into an own PLM system. The format of the transferred data differs:

- Online download: The data is transmitted as a data stream (e.g., SOAP message response for web services based implementation). File representations are not supported in this case.
- Offline download: The data is sent as a file within the download package. It can be a STEP AP214 Part21, which is specified in the server configuration and considers requirements at target side.

If the detail level covers digital documents, the download of these files will be initiated. The download of existing Part 21 files is not covered by this use case either. For this, see use case “Download of a single digital file.” If the data is sent offline, the files may be added to the download package, which is specified in the server configuration and considers requirements at target side.

This functionality covers the access of multiple PLM server Interfaces. For this, two possibilities exist:

1. The user has access to the PDM data of his direct (!) partners. This is covered by the use cases.
2. All other alternate possibilities are managed by the PLM server interface (e.g., data in a 2nd-tier supplier’s PLM system).

#### 7.2.9.1 Partner role descriptions

**Table 7.8 - Roles for downloading meta data including structures**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company’s PLM system that acts as a server.	System

#### 7.2.9.2 Non-functional requirements

The following requirements with respect to the design of the PLM client GUI are defined:

- The level of detail (“configuration”) can be defined depending on the application project. The technology for defining this configuration is not defined yet.

- The approval status of the relevant data has to be managed by the PLM server interface via authentication and authorization use cases.
- The user is not able to exclude single objects that belong to the tree defined by the start node.
- An additional use case is needed: “PLM Filter.” This use case enables the user to define some special properties that restrict the following amount of managed data.

### 7.2.9.3 Process definition

The standard process consists of the following steps (the steps directly refer to elements of the user interface of the PLM client):

1. Using the context menu (“right mouse click”) for starting the use case. The user may use this menu only for items and documents in order to be STEP compliant in any cases.
2. By identifying the menu button “download of meta data” a sub-menu appears that provides all available levels of detail (called “configurations”): download of part master data, download of part and document master data, etc.
3. The user identifies the wished level of detail using the sub-menu.
4. If the user defined to download structure information the next sub-menu appears: “Level of structure depth.”
5. In the right frame a list of items appears that were defined for the download process. The user is able to use a scroll bar for browsing through the list.

Optionally: If the download information was not already received by the client, the following steps will be performed:

6. The client is calling the PLM server using a specified query.
7. The server generates the product data and sends the resulting data stream to the client interface.

Mandatory:

8. The User starts the download by choosing the Online or Offline Download entry in the right click menu.

Online Download:

9. The PLM client sends a query to the PLM server.
10. The PLM server sends the requested data as a data stream to the PLM client.
11. The client takes the data stream, and
12. calls the “Upload Query” to the second PLM system, or
13. writes a data file.

Offline Download (see also “Initiation of an Offline Download”):

14. The PLM client sends a query to PLM server interface or to the involved EDI-Tool - Input to use case “Initiation of an Offline Download.”
15. A Client notification is created by the EDI-Tool.



### 7.2.9.4 Process diagram

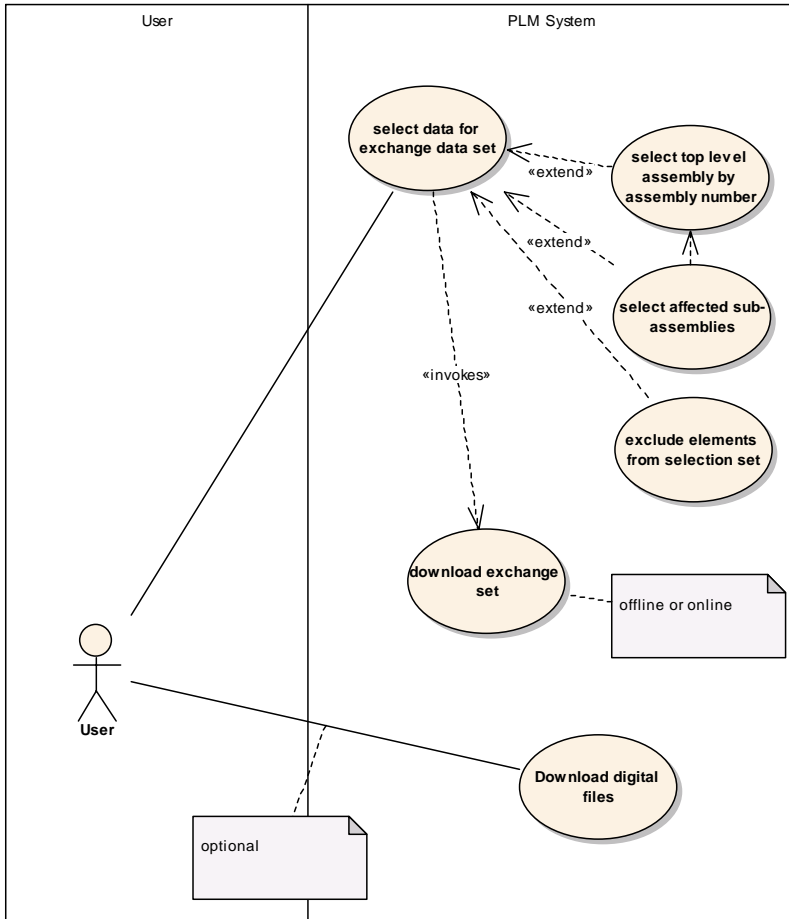


Figure 7.6 - Use case diagram download of product data

### 7.2.9.5 Process start and end states

Start state S1:

- Successful results of Authorization and Browsing use cases.

End state E1 (Success):

- Offline Download: A notification of an additional exchange process is provided (e.g., “Off-line transfer is running”).
- Online Download: A notification for the User, if the download is finished (with success or not).
- The selected meta data including structures is stored in a data file on a local computer (file system), or generated as data stream as input for the Upload use case.

End state E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:

- The user is not authorized to access the PLM server.
- The PLM server interface detected a problem.
- The user is not authorized to download the requested data.
- The PLM server itself is not available.
- Offline Download: Triggering the EDI-Tool failed.
- Online Download: Not sufficient disc space for storing the file.

### 7.2.9.6 Relevant data

- All product data (part master, document master, etc.), see Section 8.2 and Section 8.4.

### 7.2.9.7 Topics under discussion / Remarks

- This download use case ends by creating a data file or a data stream. This data can be re-used by Upload Use Cases.
- Definition of “configurations”: Should they be based on transformation rules?

### 7.2.9.8 Realization with this specification

See Section 7.2.1.

### 7.2.10 Download a single digital file

This process allows a user to download a single specific digital file (geometry file, TIFF, etc.) from a remote PLM server to a local storage. The download also includes the viewing of digital files, as far as a viewing tool is started automatically on the user side after the download process has finished. This process is called “simple viewing.”

#### 7.2.10.1 Partner role descriptions

**Table 7.9 - Roles for downloading a single digital file**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

#### 7.2.10.2 Process definition

This use case includes the identification of a single digital file to be downloaded, the start, the monitoring of the progress, and the check of the success of the data transport from a PLM server to a local storage.

The process steps are as follows:

1. The user identifies the digital file to be downloaded from the PLM server.
2. The User starts the download by choosing the Online or Offline Download entry in the right click menu.

Online Download:

3. The PLM client sends a query to the PLM server.
4. The PLM server sends the requested digital file data to the PLM client.
5. The PLM client receives the digital file and displays it directly, opens an external application to display it or let the user store it in the local file system.
6. A notification is sent to the User (in case of success and in case of failure).

Offline Download (see also “Initiation of an Offline Download,” see Section 7.2.2):

7. The PLM client sends a query to PLM server interface or to the involved EDI-Tool - Input to use case “Initiation of an Offline Download.”
8. For the file export from the PDM Vault a copy of the document should be created, no file locking mechanism (for parallel use by other users) should be implemented. The export could be triggered by the PLM server or by the EDI-Tool.
9. A Client notification is created by the EDI-Tool.

### 7.2.10.3 Process start and end states

Start state S1:

- The user has been successfully authenticated.
- The user is authorized to know that the digital file exists.
- The user has got a list or a structure tree containing at least the identifier of the digital file and an appropriate URL.
- The kind of the access (viewing, changing) is specified. Currently only viewing functionality is considered.
- The final trigger is the selection in the context sensitive menu (“Download selected file online/offline”) that belongs to a selected single digital file.

End state E1 (Success):

- Offline Download: A notification of an additional exchange process is provided (e.g., “Offline transfer is running”).
- Online Download: A notification for the User, if the download is finished (with success or not).
- The digital file, that has been specified by the user for download, is opened and displayed, or stored on the local storage.

End state E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - The user is not authorized to access the PLM server.
  - The user is not authorized to download the digital file.
  - The requested digital file is not available on the PLM server.

- The PLM server itself is not available.
- Offline Download: Triggering the EDI-Tool failed.
- Export (checkout) functionality failed (digital file doesn't exist, the file is already used by another user).
- Online Download: Not sufficient disc space for storing the files.

#### **7.2.10.4 Constraints and assertions**

- The downloaded file is always not compressed if it is sent online. Then the file can be opened directly and may be viewed using a client plug in or an external application. Compression is only allowed if an offline transfer process implies a package mechanism.
- The file name is generated by server/system specific rules.

#### **7.2.10.5 Relevant data**

- Document meta data, see Section 8.4
- Document data (digital file), see Section 8.4

#### **7.2.10.6 Realization with this specification**

Download functionality is assumed to be implemented in the PSM. A basic methodology to access the necessary URL of the desired objects is provided by the methods of the basic interface for the PLM\_connection realizations, see Section 9.2.5.

#### **7.2.11 Generic object query**

This use case allows a user to generically access objects (e.g., items, documents) as result of a specified filter condition. Feasible filter parameters and the functionality for the collection and provision of these objects have to be provided by the PLM server. Therefore, this generic use case can be specialized to further detailed use cases. Examples for detailed use cases are:

- Find all parts contained in a design space by providing bounding box parameters.
- Find heat sensitive parts by providing temperature parameters.

##### **7.2.11.1 Owner of the use case**

This use case was defined by the Work Group 2 of the PDTnet project.

##### **7.2.11.2 Process purpose**

This use case allows a user to generically access objects (items, documents) as result of a specified filter condition. Feasible filter parameters and the functionality for the collection and provision of these objects have to be provided by the PLM server. Therefore, this generic use case can be specialized to further detailed use cases. Examples for detailed use cases are:

- Find all parts contained in a design space by providing bounding box parameters.
- Find heat sensitive parts by providing temperature parameters.

### 7.2.11.3 Partner role descriptions

Table 7.10 - Roles for generic object query

Role name	Role description	Role type
User	Party that wishes to request information. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.11.4 Process definition

The process steps are:

1. User chooses the intended (and provided) functionality (specialized query).
2. User defines a development project or uses the existing one.
3. PLM client displays the parameter names, that have to be provided to filter out the correct data within the PLM server, according to the chosen functionality (see step 1).
4. User provides required parameter values (object properties, bounding box information, etc.) and initiates query to PLM server interface (single PLM Interface).
5. PLM System is processing the query that results in an object list.
6. Object list is displayed within the PLM client.

### 7.2.11.5 Process diagram

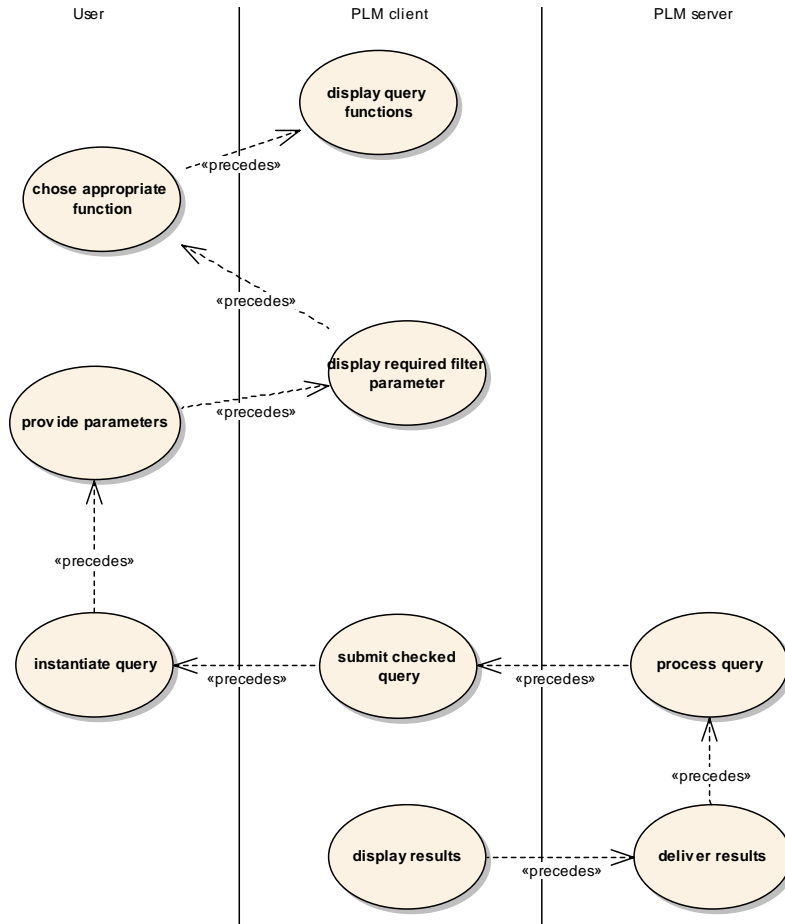


Figure 7.7 - Use case diagram for generic object query

### 7.2.11.6 Process start and end states

Start state S1:

- The authentication and authorization of the user was successful.
- A valid development project is existing.
- The available specialized types of object queries related to specific objects have been previously submitted by the PLM server (see use case “Start-up of session”).

End state E1 (Success):

- List of objects that were requested according to the specialized query and filter parameters. Example for specialized query “Search in design space”: All parts contained in the defined design space as a list of items.

End state E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - No development project defined.
  - The user is not authorized to access the data (see also use case “Authorization”).
  - The requested data is not available on the PLM server.
  - Functionality is not supported for this object type.

#### **7.2.11.7 Constraints and assertions**

Only one single PLM server is accessed. A generic object query that is sent simultaneously to more than one PLM server is not supported.

#### **7.2.11.8 Relevant data**

- Product structure data, see Section 8.3
- Basic part classification data, see Section 8.2
- Document meta data, see Section 8.4
- Document data, see Section 8.4

#### **7.2.11.9 Realization with this specification**

The generic object query functionality is defined in the Generic Queries Conformance Point, see Section 9.4.

### **7.2.12 Search in design space**

This use case is a specialization of the use case “Generic object query,” see Section 7.2.11.

#### **7.2.12.1 Process purpose**

Purpose of the “Search in design space” process is to query all parts that are located in the neighborhood of a given part. This use case allows a designer at the supplier site to search for parts that are positioned in a certain area around a specified part. The calculation of the neighborhood relation of parts will be done by using the “bounding boxes” of the parts. The user should be able to “blow up” the bounding box around a part in order to get all parts in a certain distance of the given part.

### 7.2.12.2 Partner/actor role descriptions

Table 7.11 - Roles for search in design space

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.12.3 Process definition

The process could be seen as a query in which the query parameters do not exist as discrete PDM data in the PLM system. Actually, the criteria for the evaluation of the result set is the geometrical relation between the given part and all other parts in a given assembly. For example, the designer has to modify the design of the oil pump of a car. He needs to know which parts are located near to the pump to be able to check whether the modified pump fits into the space left for this device. With the search described here, he can find those parts easily. This use case would probably only be relevant for the OEM side of the PDTnet project.

The following requirements are defined:

- The parts found during the search are displayed in form of a “virtual container” that contains all parts meeting the design space criteria. The virtual container is an assembly that is only created temporarily and which does not represent any form of a real assembly. It is only meant as a set of objects and therefore can be displayed as an assembly with one and only one level.
- It should be possible to combine different search criteria (search in design space, search by defining PDM data filters). For example, all temperature sensitive parts in a certain distance of a hot part have to be found by the query.
- To ensure the clearness of visualization, the formerly displayed structures should be made available by means of a “Pull down list” or by “Tabs” that allow going directly to the assigned structure display.
- The resulting set of items should allow performing a download (online or offline) on certain items selectable by the user.
- The user should optionally be able to define an assembly (“Start node”) in which the parts to find are contained. For example, all parts in a combustion engine should be found.
- Another option is to enter the depth of search, the levels of deepness in an assembly.



### 7.2.12.4 Process diagram

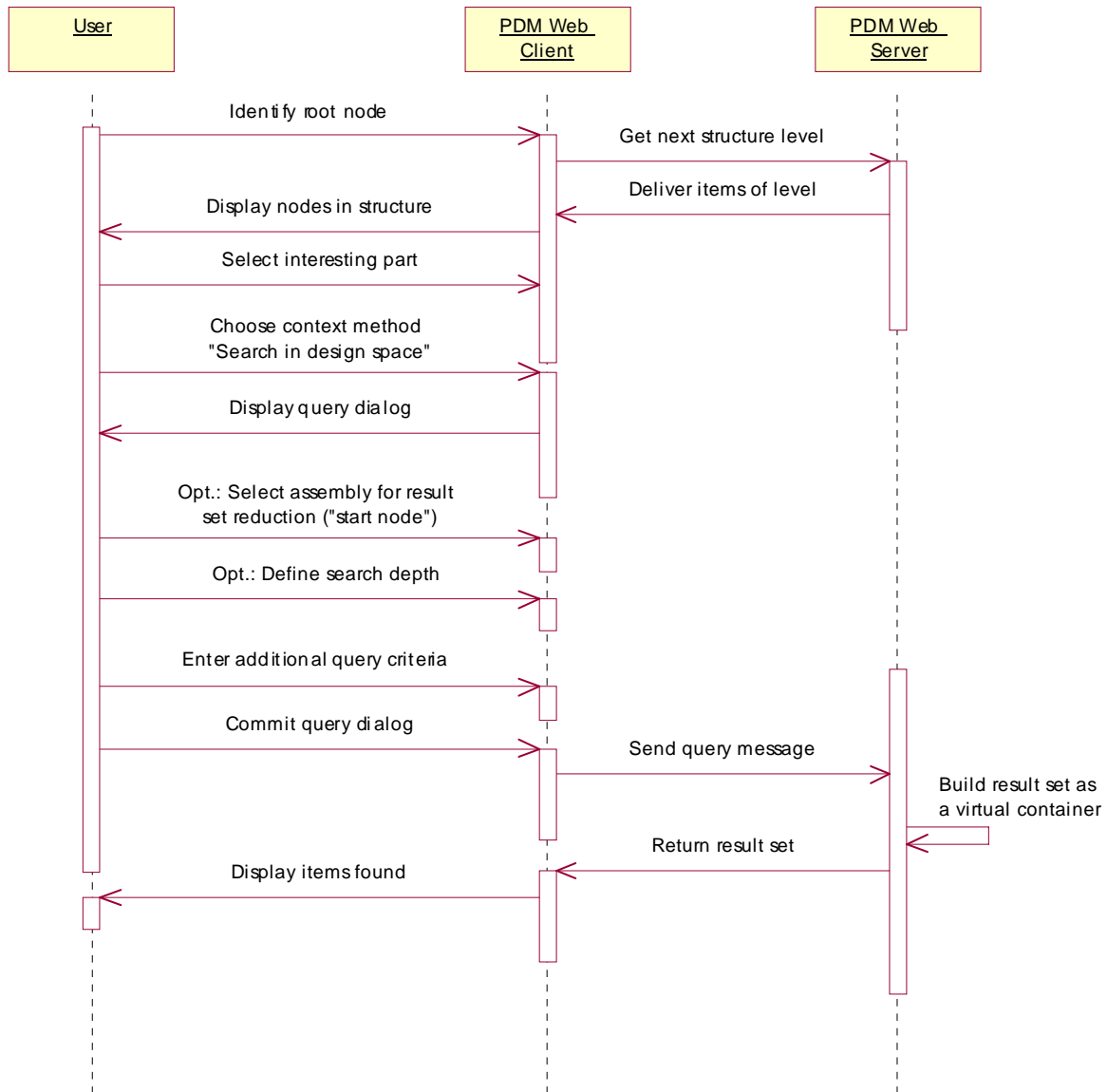


Figure 7.8 - Sequence diagram for search in design space

### 7.2.12.5 Process start and end states

Start state / precondition S1:

A specific engineering project is defined, which itself defines certain items of product data (e.g., assemblies, parts, documents) that will be subject to change or creation during the project's life time. These items are identified by identifiers.

- The end state / post condition of use case “Start node identification” or one of the children of the start node, that means an item.
- The user is correctly logged in and authorized to access the requested information.

The necessary filter information is defined (Use case “Generic object query,” see Section 7.2.11).

End state / post condition E1 (Success):

- The process results in a virtual container (see Section 7.2.12.3) containing all the accessible parts found during the query. The number of parts found is displayed.
- The virtual container contains the transformation matrices of the parts in relation to the car origin.
- If no parts or accessible parts were found, an empty virtual container is presented. The number of parts found is displayed, in this case it is 0.

End state / post condition E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - The selected part contains no geometry. Therefore, there is no possibility to find any parts in the neighborhood of the part. This should be reported by the message “Part contains no geometry.”

#### **7.2.12.6 Constraints and assertions**

- The selected part has to contain any geometry as a base for the query.

#### **7.2.12.7 Relevant data**

- Product structure data, see Section 8.3.

### **7.2.13 Upload of product data**

#### **7.2.13.1 Process purpose**

This use case allows a user to upload specific product data that was created or changed on a local storage to a remote PLM server.

This use case corresponds mainly to use case “Download of product data,” see Section 7.2.8. Additionally, it requires two functions:

- Identification of correct structure nodes for the integration of uploaded data.
- Creation/change of structures and/or structure nodes, if appropriate.

This functionality is closely related to the underlying access authorization concept. Due to the variety of PLM system specific access authorization architectures, this topic is closely depending on the PLM system functionality and/or company specific PLM system usage restrictions.

#### **7.2.13.2 Owner of the use case**

This use case was defined by the Work Group 2 of the PDTnet project.

### 7.2.13.3 Realization with this specification

The requested functionality is provided by the Generic Queries Conformance Point with the write method of the PLM\_general\_connection or PLM\_message\_connection, see Section 9.2.6 and Section 9.2.7. Specialized functions are defined in the Specific Queries and the PDTnet Queries Conformance point, see Section 9.7 and Section 9.8, respectively.

### 7.2.14 Upload a single digital file (simple user interaction)

#### 7.2.14.1 Process purpose

This process allows a user to upload a single file that was created or changed on a local storage to a remote PLM server.

#### 7.2.14.2 Process definition

This use case corresponds mainly to use case “Download of a single digital file” (see Section 7.2.10). Additionally, it requires two functions:

- Identification of the correct structure node for the integration of uploaded data.
- Creation/change of structures and/or structure nodes, if appropriate. This functionality is closely related to the underlying access authorization concept. Due to the variety of PLM system specific access authorization architectures this topic is closely depending on the PLM system functionality.

#### 7.2.14.3 Partner role descriptions

Table 7.12 - Roles for uploading of a single digital file (simple user interaction)

Role name	Role description	Role type
User	Party that wishes to store PDM data on a remote PLM server. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company’s PLM system that acts as a server. The PLM system can be extended by a Web Server to build the complete PLM server.	System

#### 7.2.14.4 Process start and end states

Start state S1:

- The user has a single file stored on his local file system to be uploaded.
- The user knows the correct structure node in the database of the PLM server for the integration of the data.

End state E1 (Success):

- Offline Upload: A notification of an additional exchange process is provided (e.g., “Offline transfer is running”).

- Online Upload: A notification for the User, if the upload is finished (with success or not). The displayed target structure is refreshed on the screen.
- The file, that had been specified by the user for upload, is stored on the remote PLM server and attached to the target structure. Maybe some new structure nodes were created to attach the file to.

End state E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - The user is not authorized to access the PLM server.
  - The user is not authorized to upload the digital file.
  - The user is not authorized to create needed structure nodes.
  - The server can't create needed structure nodes with default values.
  - The specified data could not be integrated in the database of the PLM server (e.g., the correct structure node for data integration could not be identified).
  - The PLM server itself is not available.
  - Offline Upload: Triggering the EDI-Tool failed.

#### **7.2.14.5 Constraints and assertions**

The uploaded file is always not compressed. Compression is only allowed if an offline transfer process implies a packaging mechanism.

The target element to assign an uploaded file to can be of type “Item\_version” or “Document\_version.” In case of a “Document\_version” the file can be assigned directly. If an “Item\_version” is selected, the server has to create a document with default values to assign the file to. If any creation is not possible, the action fails and the user is notified.

Any directives/parameters for the upload process are stored at server side.

#### **7.2.14.6 Relevant data**

Product structure data:

- Document meta data
- Document data (digital file)

### **7.2.15 Upload meta data including structures**

#### **7.2.15.1 Process purpose**

This process allows a user to upload meta data including structures to a remote PLM server. This data was created or changed on a local storage or is the result of a download process.

#### **7.2.15.2 Process definition**

This use case corresponds mainly to use case “Download of meta data including structures” (see Section 7.2.9). Additionally, it requires two functions:

- Identification of correct structure nodes for the integration of uploaded data.

- Creation/change of structures and/or structure nodes, if appropriate. This functionality is closely related to the underlying access authorization concept. Due to the variety of PLM system specific access authorization architectures, this topic is closely depending on the PLM system functionality.

### 7.2.15.3 Partner role descriptions

**Table 7.13 - Roles for uploading meta data including structures**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
User	Party that wishes to store PDM data on a remote PLM server. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.15.4 Process start and end states

Start state S1:

- The user has data stored on his local file system or stored temporarily as a result of a download process.
- The user knows the correct structure nodes in the database of the PLM server for the integration of the data.

End state E1 (Success):

- Offline Upload: A notification of an additional exchange process is provided (e.g., “Offline transfer is running”).
- Online Upload: A notification for the User, if the upload is finished (with success or not). The displayed target structure is refreshed on the screen.
- The data, that had been specified by the user for upload, is stored on the remote PLM server and integrated into the target structure.

End state E2 (Failure):

- The process results in a failure message. A failure can occur due to the following reasons:
  - The user is not authorized to access the PLM server.
  - The user is not authorized to upload the data.
  - The specified data could not be integrated in the database of the PLM server (e.g., the correct structure nodes for data integration could not be identified).
  - The PLM server itself is not available.
  - Offline Upload: Triggering the EDI-Tool failed.

### 7.2.15.5 Constraints and assertions

The new structure is sent as message set to the server. The data can be assigned to one or more target elements. If the whole uploaded structure should be assigned to one single element, this will be selected within a message parameter. If there are more complex relations between the new and target elements, the message set also contains the target elements

and the relationships to them. In case of an offline transfer, the message set can be replaced by a STEP Part 21 file, which is specified in the server configuration and considers requirements at target side. In case of an online transfer, STEP Part 21 is not supported.

Referenced files has to be uploaded separately using the use cases “Upload a single digital file” or “Upload a set of digital files.” If the data is sent offline, the files may be added to the upload package, which is specified in the server configuration and considers requirements at target side.

Any directives/parameters for the upload process are stored at server side.

### 7.2.15.6 Relevant data

Product structure data

- Basic part classification data
- Document meta data
- Document data

## 7.2.16 Change notification

### 7.2.16.1 Process purpose

The designer of a part needs notification when a change to a part happens that affects one of the parts he is responsible for. This could take place when a part in the neighborhood of a given part is changed in its dimensions or properties or when a part in an assembly is moved to another place than before. The user specifies the parts on which he wants to be notified by using the functionality of subscribing specified in use case “Change content of subscription list.”

### 7.2.16.2 Partner/actor role descriptions

**Table 7.14 - Roles for change notification**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
E-Mail Client	System that is able to maintain the user’s e-mail.	System
PLM server	System that provides the relevant PDM data. This is usually a company’s PLM system that acts as a server.	System

### 7.2.16.3 Process definition

Target of the process is the evaluation of objects being changed since the last visit of the user to this object. When a modification of those objects is being detected, an appropriate message has to be delivered to the user. Objects could be parts (and part versions), documents (and document versions), or models.

Changes to report could be:

- Creation of a new version of an object.

- Change of the release status of an object.
- Objects are deleted.
- Geometry has changed.
- Properties have changed.

The following requirements are defined:

- Two possibilities of detecting changes on the server side are conceivable. Which of them is used depends on the PLM server implementation:
- Whenever an object linked to anybody's subscription list is changed, an e-mail is sent to the user(s).
- In certain periods of time, the subscription lists of all users are checked against the objects they include. When a modification of a certain object is detected, an e-mail is sent to the user.
- The frequency and content of e-mail notifications (confidential data must not be included!) are defined server-specifically.

#### 7.2.16.4 Process diagram

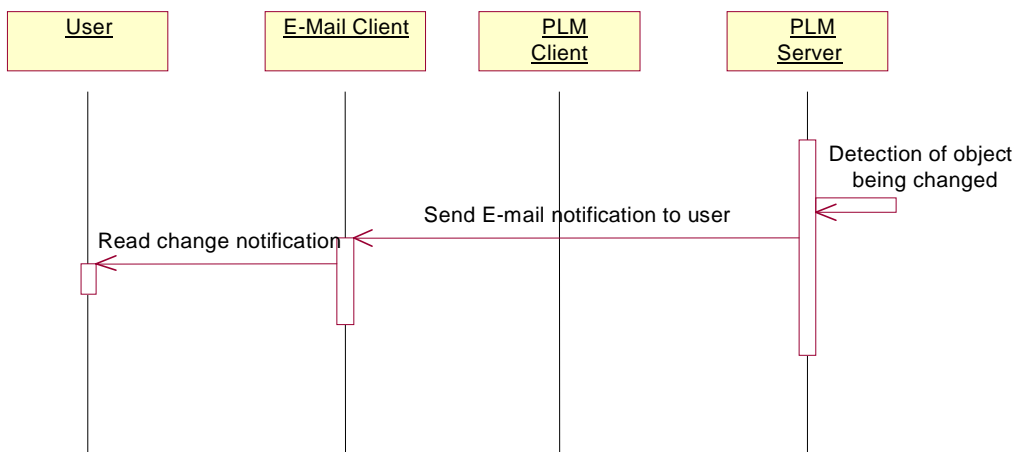


Figure 7.9 - Process diagram for change notification

#### 7.2.16.5 Process start and end states

Start states / preconditions S1 and S2:

- User has access to his e-mail client.

End state / post condition E1 and E2 (Success):

- An e-mail notification about changes to one of his objects collected in the clipboard is sent to the user.

#### 7.2.16.6 Relevant data

- Product meta data

## 7.2.17 Display content of subscription list and confirm changes

### 7.2.17.1 Process purpose

To get an overview about objects being changed on the PLM server, the user should be able to display the contents of his subscription list in which he collects all the objects to track. The changed objects should be displayed in an emphasized style to show the status of being changed.

The current content of the subscription list including notifications of changes can be requested by the PLM client:

- when logging in at the server,
- when interactively initiated by the PLM client user.

### 7.2.17.2 Partner/actor role descriptions

**Table 7.15 - Roles for displaying content of subscription list and confirm changes**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.17.3 Process definition

Target of the process is the evaluation of objects being changed on the PLM server since the last visit of the user and the notification of the user by displaying the content of the subscription list. When a modification of those objects is being detected, the objects are marked as changed in the subscription list and the reasons of the changes are displayed.

The following requirements are defined:

- The user controls the start of the evaluation process via the client. The results of the evaluation process are displayed directly in the client.
- The change notification data is transferred by the PLM server using the data constructs provided by AP214 (work management information). An additional transfer of change management/notification documents (like PDF files) is currently not needed.
- The user must be able to define and to modify the content of his subscription list (see use case “Change content of subscription list”).
- The subscription list should be represented as a separate folder within the PLM client GUI.



#### 7.2.17.4 Process diagram

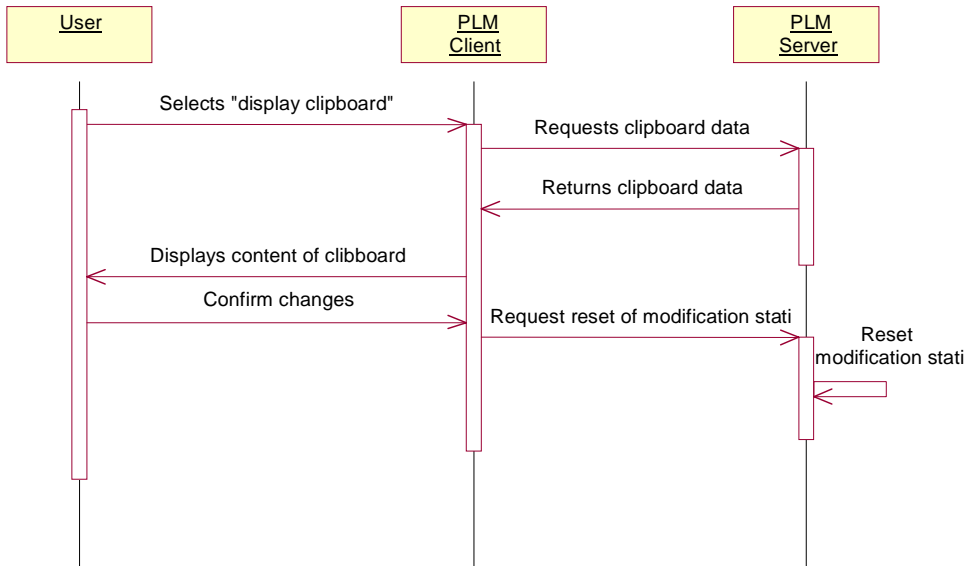


Figure 7.10 - Display Content of Subscription list and confirm changes Process Flow

#### 7.2.17.5 Process start and end states

Start state / precondition S1:

A specific engineering project is defined, which itself defines certain items of product data (e.g., assemblies, parts, documents) that will be subject to change or creation during the project life time. These items are identified by identifiers.

- The user is correctly logged in and authorized to access the requested information.

End state / post condition E1 (Success):

- The process results in a virtual container (see use case “Search in design space”) containing all the objects in the subscription list.
- Objects modified since the last look on the subscription list are displayed emphasized. Deleted objects are displayed in a different style.
- After confirmation, the modification status of the objects is reset and in the case of deleted objects in the PLM system, they are also deleted from the subscription list.

#### 7.2.17.6 Relevant data

- Product meta data
- Work management data

## 7.2.18 Change content of subscription list

### 7.2.18.1 Process purpose

The idea of the subscription list is that the user needs a sort of folder in which he can collect objects. The purpose of the Subscription lists is to collect objects for which the change notification should be provided. The modification of the objects in this subscription lists is tracked and the user will be notified if such a modification takes place. The user should be able to change the content of his subscription list. The subscription list contains all objects the user wants to be notified when changes are applied to them.

### 7.2.18.2 Partner/actor role descriptions

Table 7.16 - Roles for changing content of subscription list

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.18.3 Process definition

1. The user selects objects in the subscription list and wants the PLM system to delete the objects from the subscription list.
2. The user selects objects in the PLM client and wants the PLM system to link those objects into the subscription list.

The following requirements are defined:

- The user has a subscription list in the PLM system.
- For use case a), the content of the subscription list with the objects to delete have to be displayed.
- For use case b), the objects to add have to be displayed in the client.

### 7.2.18.4 Process diagram

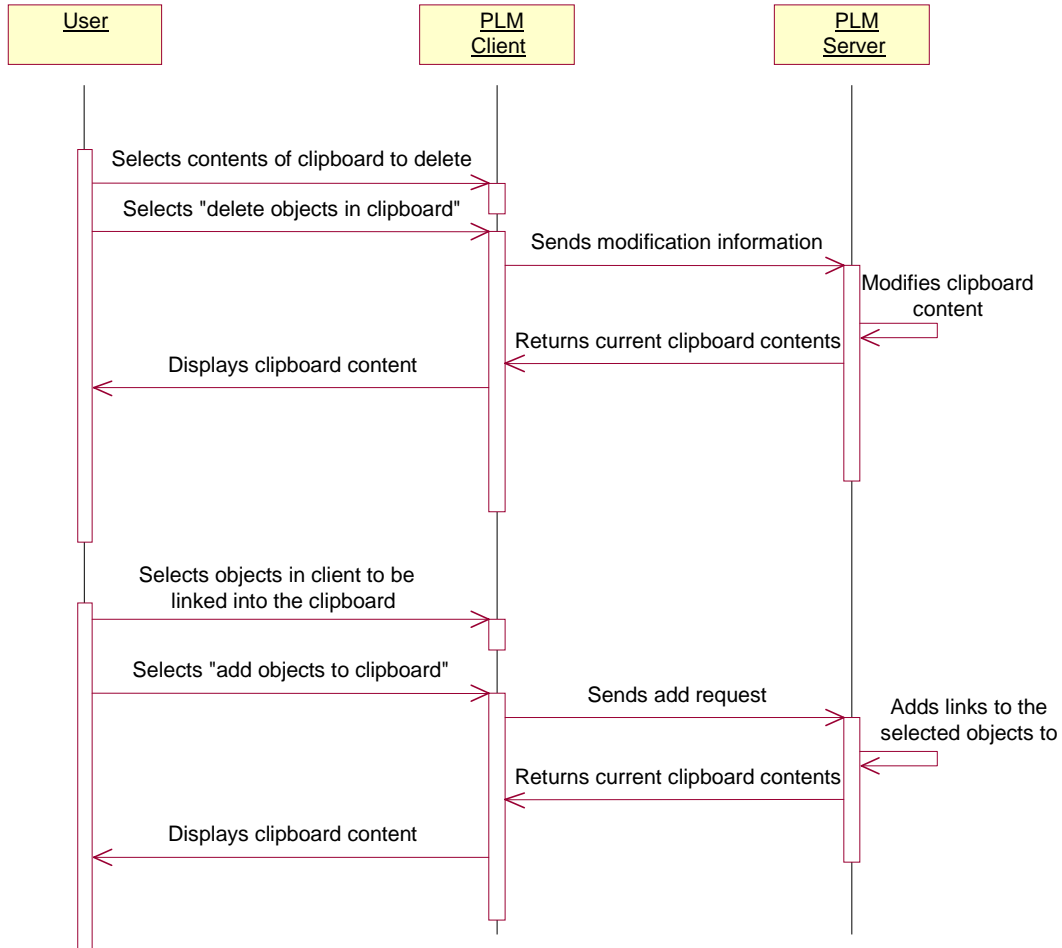


Figure 7.11 - Change content of subscription list process flow

### 7.2.18.5 Process start and end states

Start state / precondition S1 (use case a):

A specific engineering project is defined, which itself defines certain items of product data (e.g., assemblies, parts, documents) that will be subject to change or creation during the project life time. These items are identified by identifiers.

- The user is correctly logged in and authorized to access the requested information.
- The content of the subscription list is being displayed in the client.

Start state / precondition E2 (use case b):

A specific engineering project is defined, which itself defines certain items of product data (e.g., assemblies, parts, documents) that will be subject to change or creation during the project life time. These items are identified by identifiers.

- The user is correctly logged in and authorized to access the requested information.
- Product data is displayed.

End state / post condition E1 and E2 (Success):

- The process results in an updated view to the subscription list.

### 7.2.18.6 Relevant data

Product meta data

## 7.2.19 Product Class Identification

### 7.2.19.1 Process purpose

Identification of a top level product\_class to enable browsing of an abstract product structure.

### 7.2.19.2 Partner/actor role descriptions

**Table 7.17 - Roles for product class identification**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.19.3 Process definition

This use case defines the process of identifying the start node of an abstract product structure in a PLM server. The end state / post condition of the use case is the precondition of the use case “Browsing of an abstract product structure.”

The process steps are:

- The user enters an ID or Wild Card.
- PLM server receives ID or Wild Card and triggers search in PLM System.
- Exception: The PLM server does not respond.
- PLM System executes query in its database.

-> Exception: Database is not available, no data found, user is not authorized to access the data, etc.

- PLM server returns a list of product\_class and product\_component nodes.
- PLM client displays the resulting product\_class nodes. If the list has only one member, it shall be displayed as the root node of a tree. If the list contains more than one node, then the result should be displayed as a list from which the user may select one node that is then displayed as the root node of a tree.

Remark: according to the AP214 CC8 Recommended Practices, each product\_class is associated to one instance of product\_component (with relation\_type='realization') having the same attribute values. From this instance of product\_component (not displayed within the client), the abstract product structure may be traversed (ProductStructureQuery).

#### **7.2.19.4 Process start and end states**

Start state / precondition S1:

- The user is correctly logged in and authorized to access the requested information.
- The service is available.
- The user enters an Id or Wild Card.

End state / post condition E1 (Success):

- The list of resulting nodes is displayed as described above.

End state / post condition E2 (Failure):

- The process results in a failure message.

#### **7.2.19.5 Constraints and assertions**

None.

#### **7.2.19.6 Relevant data**

- Product\_class information

### **7.2.20 Browsing of Abstract Product Structures**

#### **7.2.20.1 Process purpose**

This process allows a user starting with an identified product\_class, product\_component, or alternative\_solution to get information on the subcomponents of an abstract product structure (product\_component or item\_instance).

### 7.2.20.2 Partner/actor role descriptions

**Table 7.18 - Roles for browsing of abstract product structures**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.20.3 Process definition

The process steps are:

- The PLM client evaluates if the product structure information is already obtained, then it is directly displayed in a table.
- The PLM client sends a query for a substructure of product\_class, product\_component, or alternative\_solution specified by the user to the PLM server.
- For each product structure node in the scope of the query the PLM server.
- Checks the authorization regarding the requested data.

-> Exception: Access denied

- Collects requested data within the PLM server.
- PLM server sends data to the PLM client.
- PLM client displays the resulting nodes within the structure. The kind of relationship (e.g., product\_structure\_relationship of kind “decomposition” or “realization”) and child node (product\_component or item\_instance) should be displayed within the PLM client.

Remark: only one level of the product structure is retrieved at a time.

Remark: only product\_structure\_relationships from product\_component to product\_component from alternative\_solution to item\_instance and from alternative\_solution to product\_component are supported.

Remark: all the subtypes of item\_instance are supported (single, quantified, and selected). selected\_instance is used in the case of a quantity ‘as needed’: selected\_instance.selection\_quantity refers to an instance of value\_limit with limit=0 and limit\_qualifier=‘minimum.’

Remark: this functionality is also available on item\_version nodes if they are handled both as part (for their usage) as well as product\_component (having an own abstract product structure). In this case, the function handles the item\_version just as if it was a product\_component.

### 7.2.20.4 Process start and end states

Start state / precondition S1:

- The user is correctly logged in and authorized to access the requested information.
- The service is available.
- The user enters an Id.

End state / post condition E1 (Success):

- The list of resulting of the resulting nodes is displayed as described above.

End state / post condition E2 (Failure):

- The process results in a failure message.

### 7.2.20.5 Relevant data

- Product\_structure\_relationships, Product\_components, Alternative\_solutions, Item\_instances.

## 7.2.21 Browsing of Alternative Solutions within an Abstract Product Structure

### 7.2.21.1 Process purpose

This process allows a user starting with an identified product\_component (or alternative\_solution) to get information on the (sub-)alternative solutions of an abstract product structure.

### 7.2.21.2 Partner/actor role descriptions

**Table 7.19 - Roles for browsing of alternate solutions within an abstract product structure**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.21.3 Process definition

The process steps are:

- The PLM client evaluates if the alternative\_solutions are already obtained, then it is directly displayed in a table.
- The PLM client sends a query for the alternative solutions of a product\_component (or alternative\_solution) specified by the user to the PLM server.
- For each alternative solution node in the scope of the query the PLM server.
- Checks the authorization regarding the requested data.

-> Exception: Access denied

- Collects requested data within the PLM server.
- PLM server sends data to the PLM client.
- PLM client displays the resulting nodes within the structure. The kind of child node (alternative\_solution, technical\_solution, final\_solution, supplier\_solution) should be displayed within the PLM client.

#### 7.2.21.4 Process start and end states

Start state / precondition S1:

- The user is correctly logged in and authorized to access the requested information.
- The service is available.
- The user enters an Id.

End state / post condition E1 (Success):

- The list of resulting of the resulting nodes is displayed as described above.

End state / post condition E2 (Failure):

- The process results in a failure message.

#### 7.2.21.5 Relevant data

- Product\_structure\_relationships, Product\_components

### 7.2.22 Retrieve Configuration Data within an Abstract Product Structure

#### 7.2.22.1 Process purpose

This process allows a user starting with an identified alternative\_solution or item\_instance to get information on the configuration of an abstract product structure.

#### 7.2.22.2 Partner/actor role descriptions

**Table 7.20 - Roles for retrieving configuration data within an abstract product structure**

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

#### 7.2.22.3 Process definition

The process steps are:



- The PLM client evaluates if configuration information is already obtained, then it is directly displayed in a table.
- The PLM client sends a query for the configuration[s] of an alternative\_solution or item\_instance specified by the user to the PLM server.
- For [each] configuration node in the scope of the query the PLM server.
- Checks the authorization regarding the requested data.

-> Exception: Access denied

- Collects requested data within the PLM server.
- PLM server sends data to the PLM client.
- PLM client displays the resulting nodes within the structure. The associated Specification referenced through Configuration and Class\_specification\_association should be displayed within the PLM client as a property of the configuration.

Remark: currently, configuration may be only displayed on alternative\_solution and item\_instance, but not on product\_component and product\_function.

Remark: for complexity reason the specification\_expression corresponding to the logical rule stored within the legacy system is mapped to a single string and mapped to a pseudo-Specification.id. This specification is directly referenced by the Class\_specification\_association. The category of this specification has id=/DUMMY.

Remark: the product\_class referenced by the class\_specification\_association will not be displayed to the PLM client, since it is either derived from the root node of the abstract product structure, or is project independent (for example, in the case on configured assembly structures) and would have to be instantiated with a product\_class of kind 'enterprise.'

Remark: if the usage of a part or product\_component is not configured (i.e., the associated logical rule is empty), this function will give no results.

#### **7.2.22.4 Process start and end states**

Start state / precondition S1:

- The user is correctly logged in and authorized to access the requested information.
- The service is available.
- The user enters an Id.

End state / post condition E1 (Success):

- The list of resulting of the resulting nodes is displayed as described above.

End state / post condition E2 (Failure):

- The process results in a failure message.

#### **7.2.22.5 Relevant data**

- Alternative\_solution, Item\_instance, Configuration, Product\_class, Class\_specification\_association, Specification, Specification\_category, see Section 8.3 and Section 8.10.

## 7.2.23 Viewing of Change Management Information

### 7.2.23.1 Process purpose

Browsing through a product structure the user is able to see the assigned change management information.

### 7.2.23.2 Partner/actor role descriptions

Table 7.21 - Roles for viewing of change management information

Role name	Role description	Role type
User	Party that requests PDM data. This could be a person who interacts with the PLM client, or a system that triggers the PLM client.	Person / System
PLM client	System that provides the communication between user and PLM server.	System
PLM server	System that provides the relevant PDM data. This is usually a company's PLM system that acts as a server.	System

### 7.2.23.3 Process definition

The process steps are:

- The user selects a node (product\_class, product\_component, item\_version) within the PLM client.
- The PLM client evaluates if work management information is already obtained, then it is directly displayed in a table.
- If work management information is not obtained, the PLM client sends a query for this node to the PLM server.
- PLM System executes query in its database.

-> Exception: Database is not available, no data found, user is not authorized to access the data, etc.

- PLM server sends obtained work management data to the PLM client.
- PLM client displays the resulting data in a table.

Remark: according to the CC8 Recommended Practices, the effectivity references an event\_reference, which references again an activity. Effectivity\_assignment.effective\_element and Activity\_Element.element both reference the product\_class, product\_component, or item\_version node.

Remark: other object nodes are not supported at this time.

### 7.2.23.4 Process start and end states

Start state / precondition S1:

- The user is correctly logged in and authorized to access the requested information.
- The service is available.
- The user selects a node of kind product\_class, product\_component, or item\_version in the tree view.

End state / post condition E1 (Success):

- The resulting information is displayed as described above.

End state / post condition E2 (Failure):

- The process results in a failure message.

### 7.2.23.5 Relevant data

- Activity, Activity\_element, Effectivity, Effectivity\_assignment, Event\_reference, see Section 8.10 and Section 8.11.

## 7.2.24 ECM participant proposal for a change

### 7.2.24.1 Owner of the use case

This use case was defined by the joint ECM Project Group of the ProSTEP iViP association and the VDA.

### 7.2.24.2 Process purpose

The ECM participant requests a new engineering change request and is not involved in subsequent processing of this request. This use case is used if the participant identifies the potential need for an engineering change but is not directly affected by the engineering change itself (for example, a change to a part that is the responsibility of a partner/supplier). In this use case, an automotive OEM is typically the ECM coordinator and a supplier is typically the ECM participant.

### 7.2.24.3 Partner role descriptions

Table 7.22 - Roles for the use case ECM participant proposal for a change

Role name	Role description	Role type
ECM coordinator	Owner of the engineering change request (e.g., an automotive OEM).	organization
ECM participant	Party involved in the elaboration of the engineering change request (e.g., a part supplier).	organization
ECM server	ECM system, that provides the relevant ECM data. It is usually the ECM system of the ECM coordinator that acts as a server.	system
ECM client	System of the ECM participant, that provides the communication between ECM participant and ECM server of the ECM coordinator.	system

### 7.2.24.4 Process definition

The ECM participant uses the ECM client to send a message (Request\_initial\_ECR) to the ECM coordinator to request him to create a new engineering change request. The ECM server of the ECM coordinator first sends a response message (Respond\_initial\_ECR) to the request. This message contains the coordinator's and the participant's request number (counterparts). The engineering change request is handled subsequently under the responsibility of the ECM coordinator. The ECM client of the ECM participant then receives the message notifying him whether the engineering change request

has been created (Notify\_initial\_ECR\_accepted) or rejected (Notify\_initial\_ECR\_rejected). If the engineering change request has been created, only one further message (Notify\_ECR\_decided) is received by the ECM client, notifying the ECM participant whether the ECM coordinator has approved, rejected, or canceled the request.

### 7.2.24.5 Process diagram

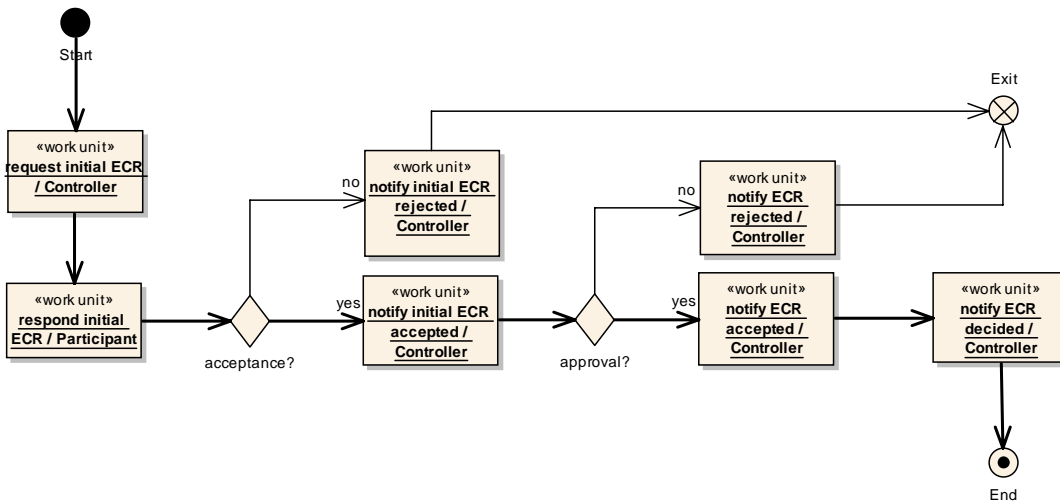


Figure 7.12 - Illustration of the process definition of the use case “ECM Participant proposal for a change”

### 7.2.24.6 Process start and end states

Start state / precondition:

- The ECM participant has elaborated an engineering change request.

End state / post condition:

- The ECM coordinator has rejected the engineering change request of the ECM participant (failure).
- The ECM coordinator has further elaborated/detailed and approved the engineering change request of the ECM participant (success).
- The ECM coordinator has further elaborated/detailed and disapproved the engineering change request of the ECM participant (failure).

### 7.2.24.7 Relevant data

- Work management data, see Section 8.11.

## 7.2.25 ECM participant comments

### 7.2.25.1 Owner of the use case

This use case was defined by the joint ECM Project Group of the ProSTEP iViP association and the VDA.

### 7.2.25.2 Process purpose

In this use case, the ECM participant is requested to provide a technical and/or commercial evaluation and comment of an ECR, but the ECM participant was not involved in the preceding technical analysis and detailing phase.

In this use case, an OEM is typically the coordinator of the engineering change request. The engineering change request is primarily within his responsibility. For sets of parts developed or manufactured externally, the supplier is incorporated in the role of an ECM participant and provides comments for the parts that affect him or for which he is responsible.

### 7.2.25.3 Partner role descriptions

**Table 7.23 - Roles for the use case “ECM participant comments”**

Role name	Role description	Role type
ECM coordinator	Owner of the engineering change request (e.g., an automotive OEM).	organization
ECM participant	Party involved in the elaboration of the engineering change request (e.g., a module supplier).	organization
ECM server	ECM system, that provides the relevant ECM data. It is usually the ECM system of the ECM coordinator that acts as a server.	system
ECM client	System of the ECM participant, that provides the communication between ECM participant and ECM server of the ECM coordinator.	system

### 7.2.25.4 Process definition

Simple case: The ECM coordinator requests that the ECM participant provide a comment of the engineering change request (Request\_ECR\_comments). The ECM participant provides the requested information (Respond\_ECR\_comments) and receives a message notifying him whether the request was approved or rejected by the coordinator (Notify\_ECR\_decided) on completion of the entire engineering change request.

Options: If the need for an engineering change is first identified by an ECM participant, he can send the request to create an engineering change request to the coordinator (Request\_initial\_ECR). The ECM participant first receives a confirmation that his request has been received under the initial engineering change request number of the coordinator (Respond\_initial\_ECR). Later on, the ECM coordinator sends the message Notify\_initial\_ECR\_rejected if the request was rejected, i.e., if no engineering change request was created, or the message Notify\_initial\_ECR\_accepted if a regular engineering change request was created. The subsequent procedure is the same as for the simple case.

In addition to the simple case or to the aforementioned option, the ECM coordinator can also request that the ECM participant provide a confirmation comment following receipt of the comments (Request\_ECR\_confirmation\_comments). Depending on the agreement made, the ECM participant provides his feedback and complete comments on the entire engineering change request or on missing or incorrect information and provides any additional information (Respond\_ECR\_confirmation\_comments).

### 7.2.25.5 Process diagram

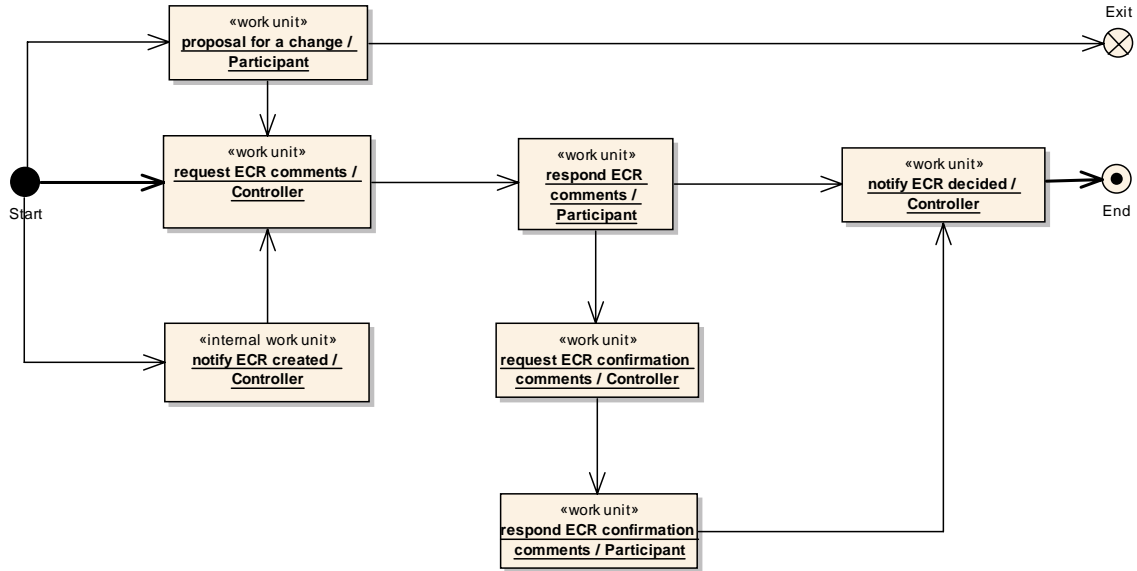


Figure 7.13 - Illustration of the process definition of the use case “ECM participant comments”

### 7.2.25.6 Process start and end states

Start state / precondition:

- The ECM coordinator requests comments about an ECR from an ECR participant (simple case).
- The ECM participant has elaborated an engineering change request.

End state / post condition:

- The ECM coordinator has rejected the engineering change request of the ECM participant (failure).
- The ECM coordinator has further elaborated/detailed and approved the engineering change request (success).
- The ECM coordinator has further elaborated/detailed and disapproved the engineering change (failure).

### 7.2.25.7 Constraints and assertions

For constraints and assertions of this use case, see [VDA].

### 7.2.25.8 Relevant data

- Work management data, see Section 8.11.

## 7.2.26 ECM participant approval

### 7.2.26.1 Owner of the use case

This use case was defined by the joint ECM Project Group of the ProSTEP iViP association and the VDA.

### 7.2.26.2 Process purpose

This use case comes into play when the supplier has full responsibility for development of certain sets of parts of the OEM and these are to be changed on the initiative of the partner, or when the supplier acts as the prime contractor in the vehicle project.

In this case, the supplier takes on the role of the ECM coordinator and the OEM takes on the role of the ECM participant. In this use case, the need to make an engineering change is identified only by the ECM coordinator and communication between the ECM coordinator and the ECM participant takes place solely in the approval phase of the ECR process.

### 7.2.26.3 Partner role descriptions

**Table 7.24 - Roles for the use case “ECM participant approval”**

<b>Role name</b>	<b>Role description</b>	<b>Role type</b>
ECM coordinator	Owner of the engineering change request (e.g., a prime contractor).	organization
ECM participant	Party involved in the elaboration of the engineering change request (e.g., an automotive OEM).	organization
ECM server	ECM system, that provides the relevant ECM data. It is usually the ECM system of the ECM coordinator that acts as a server.	system
ECM client	System of the ECM participant, that provides the communication between ECM participant and ECM server of the ECM coordinator.	system

### 7.2.26.4 Process definition

In this use case, the ECM coordinator issues an engineering change request internally, elaborates the engineering change request in detail, provide comments on the proposed changes from his point of view and sends this detailed engineering change request, which is complete from his point of view, to the ECM participant, requesting him to check the technical completeness of the request (Request\_ECR\_completeness\_check message).

The ECM participant first sends back to the ECM coordinator the engineering change request number he is using internally to track the engineering change request in the Notify\_ECR\_id message, together with the ECM coordinator's engineering change request number. The ECM participant can reject the request at this early stage. This is done using the “reject” flag. The ECM coordinator must respond by sending a Notify\_ECR\_decided message.

If the ECM participant did not reject the engineering change request early, the ECM participant sends his response to the ECM coordinator's request for a completeness check in the Respond\_ECR\_completeness\_check message.

- If the ECM coordinator’s engineering change request is incomplete from the point of view of the ECM participant, the ECM participant informs the coordinator using the flag “is\_not\_complete” and includes the set of parts that are missing in his opinion and/or his comments. The ECM coordinator must then send a further Request\_ECR\_completeness\_check message.
- If, on the other hand, the engineering change request is complete, the ECM participant informs the ECM coordinator using the flag “is\_complete.”

As soon as the ECM coordinator has received a positive response from the ECM participant with respect to the completeness of the engineering change request, he can send the final engineering change request to the participant using the Request\_ECR\_acceptance message, requesting the ECM participant to approve the engineering change request.

The ECM participant sends his response to the ECM coordinator using the Respond\_ECR\_acceptance message. This response takes the form of approval or rejection from the point of view of the ECM participant.

Finally, the ECM coordinator informs the ECM participant of the final decision regarding the engineering change request in the Notify\_ECR\_decided message. This message is also sent if the ECM participant has rejected the engineering change request early using the “reject” flag in the Request\_ECR\_completeness\_check message.

### 7.2.26.5 Process diagram

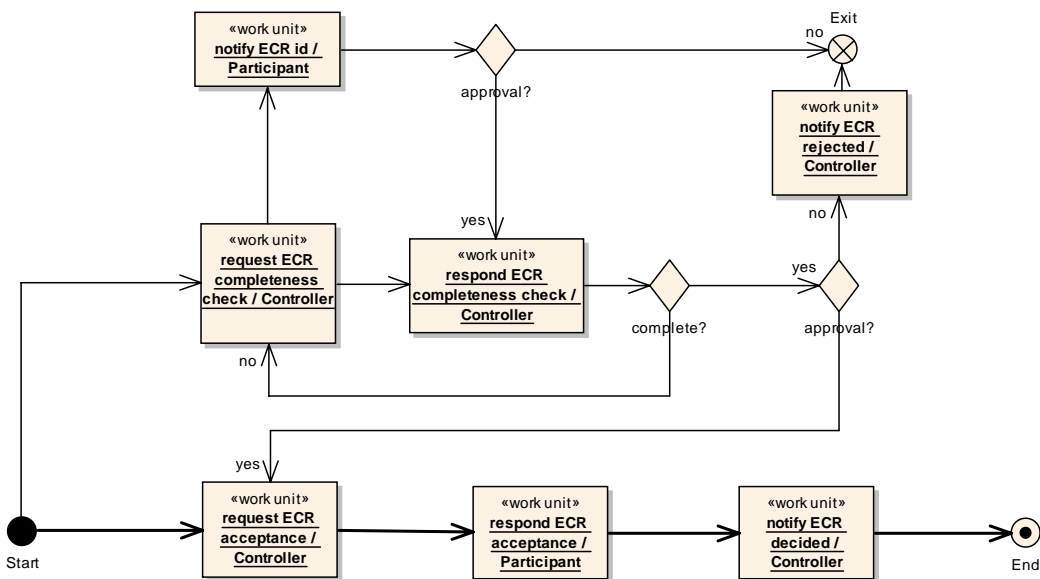


Figure 7.14 - Illustration of the process definition of the use case “ECM participant approval”

### 7.2.26.6 Process start and end states

Start state / precondition:

- The ECM coordinator sends the Request\_ECR\_completeness\_check message to the ECM participant

End state / post condition:

- The ECM coordinator has finally approved the engineering change request (success).



- The ECM coordinator has finally disapproved the engineering change (failure).

### 7.2.26.7 Constraints and assertions

For constraints and assertions of this use case, see [VDA].

### 7.2.26.8 Relevant data

- Work management data

## 7.2.27 ECM participant detailing and comments

### 7.2.27.1 Owner of the use case

This use case was defined by the joint ECM Project Group of the ProSTEP iViP association and the VDA.

### 7.2.27.2 Process purpose

This use case describes a joint engineering change process where the ECM coordinator and ECM participant work closely together throughout all phases of the process. A typical application is the development and production of complete systems or entire vehicles in a cooperative venture. Typically, the OEM takes on the role of the ECM coordinator and its cooperation partner is in the role of the ECM participant.

### 7.2.27.3 Partner role descriptions

**Table 7.25 - Roles for the use case “ECM participant detailing and comments”**

Role name	Role description	Role type
ECM coordinator	Owner of the engineering change request (e.g., an automotive OEM).	organization
ECM participant	Party involved in the elaboration of the engineering change request (e.g., a system supplier or a prime contractor).	organization
ECM server	ECM system, that provides the relevant ECM data. It is usually the ECM system of the ECM coordinator that acts as a server.	system
ECM client	System of the ECM participant, that provides the communication between ECM participant and ECM server of the ECM coordinator.	system

### 7.2.27.4 Process definition

Either the ECM coordinator (Notify\_ECR\_creation) or the ECM participant (Request\_initial\_ECR) requests the creation of an engineering change request. In the technical analysis phase, the ECM coordinator obtains information about particular objects (parts and documents), which affect the ECM participant (Request\_ECR\_details). The ECM participant can also add further objects to the engineering change request (Respond\_ECR\_details). If required, the ECM coordinator can include the ECM participant in the commenting phase. He can request a comment on individual affected objects (e.g.,

the costs for a part) or on the entire engineering change request including any comments already received from another ECM participant (Request\_ECR\_comments). Finally, the ECM coordinator informs the ECM participant of the decision made with respect to the engineering change request.

### 7.2.27.5 Process diagram

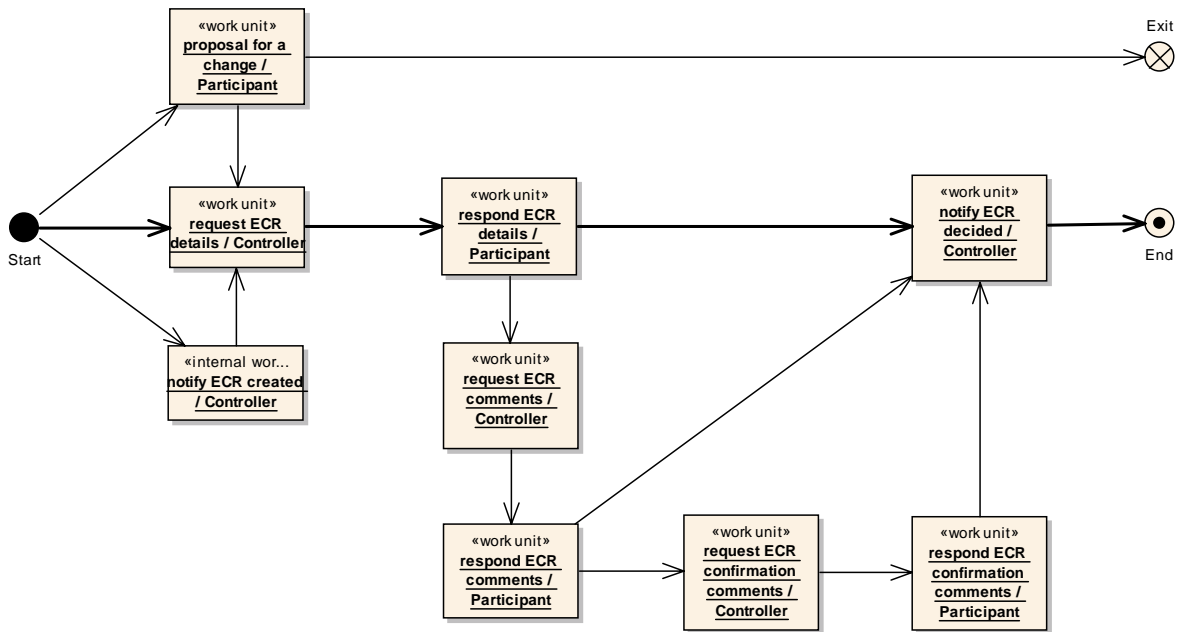


Figure 7.15 - Illustration of the process definition of the use case “ECM participant detailing and comments”

### 7.2.27.6 Process start and end states

Start state / precondition:

- The ECM coordinator creates an ECR
- The ECM participant has elaborated an engineering change request.

End state / post condition:

- The ECM coordinator has rejected the engineering change request of the ECM participant (failure).
- The ECM coordinator has further elaborated/detailed and approved the engineering change request (success).
- The ECM coordinator has further elaborated/detailed and disapproved the engineering change (failure).

### 7.2.27.7 Constraints and assertions

For constraints and assertions of this use case, see [VDA].

### 7.2.27.8 Relevant data

- Work management data, see Section 8.11.



## 8 Informational Viewpoint PIM (normative)

In this chapter the PIM in the informational viewpoint of the PLM Services 2.0 is defined. It is derived from the STEP AP214 CC21 ARM model.

Additionally, some new classes were created and put into a package called “PLM\_Base.” This package realizes two modeling concepts of the PIM. Firstly, it introduces the concept of identifying instances by a unique identifier. This identifier must be unique throughout a session as defined by the computational model in Chapter 9. Secondly, it defines a container concept to establish a correct handling of the data passed to and from the computational model.

All classes and interfaces are listed with their packages, base classes, attributes, compositions, and associations. Additionally the classes and their members are described textually. The text of all descriptions (except for the “PLM Base”) are reproduced from ISO 10303-214 with permission of ISO. The copyright remains with ISO.

The PIM Informational Model has the following package hierarchy:

Package PLM\_services

- Package PLM\_base
- Package Part\_identification  
This package includes the primary objects used for product data management. This subset provides the capability to represent product management information. It includes information about items that are either raw materials, parts, or tools about versions and views of items. A part may represent one of a variety of physical entities used in discrete manufacturing; including raw material, semi-finished parts, assemblies, instruction manuals, kits, manufacturing by-products, and products. The manufacturing industry is defined by the design, production, and sales of parts, and almost every business activity in some way works with data that describes parts.
- Package Part\_structure  
Base of this package is the group of objects that define the bill of material relationships between items for discrete manufacturing.  
  
A part is not defined by a single object with a set of attributes, but a collection of objects and relationships, each describing different aspects of the part. For example, a part definition may consist of several engineering attributes, links to suppliers of the part, references to CAD drawings describing the parts geometry, and a list of components used to assemble the part. These different pieces of the part definition will be referred to as part data objects. This subset supports explicit hierarchical product structures representing assemblies and the constituents of those assemblies. This explicit part structure corresponds to the traditional engineering and manufacturing bill of material indented parts list.
- Package Document\_and\_file\_management  
The scope of this package is the handling of electronic documents comprising one or more files and track documents that are not actively managed by the PLM system.

External files represent a simple external reference to a named file. An external file is not managed independently by the system - there is usually no revision control or any representation definitions of external files. Version identification may optionally be associated with an external file, but this is for information only and is not used for managed revision control.

- If a file is under configuration control, it should be represented as a constituent of a document definition view/representation. In this case it is actually the managed document that is under direct configuration control, the file is in this way indirectly under configuration control. A change to the file results in a change to the managed document (i.e.,

a new version) - the changed file would be mapped as a constituent of a view/representation definition of the new document version. A simple external reference alone is not configuration controlled; it is just an external file reference to product data. Documents may be associated with product data in a specified role, to represent some relationship between a document and other elements of product data. Constraints may also be specified on this association, in order to distinguish an applicable portion of an entire document or file in the association.

- Package Shape\_definition and\_transformation

The scope of this subset provides the capability to associate items with shape or to identify aspects of the shape. It allows also to distinguish between geometric elements used as auxiliary elements and geometric elements that describe product data. Additionally, it contains the capability of an empty geometric model with only a geometric element for placement purposes and an unconstrained three-dimensional geometric model that may contain any geometric data elements.

This subset allows linking geometric structures that result from relating different shape representations with associated product structure when applicable, i.e., when the geometric structure directly corresponds to the assembly structure.

Two alternatives for the implementation of geometric structures related to assembly structures are recommended:

- The assembly is described with the components built in. With this approach the shape of the component is mapped into the shape of the assembly via mapped\_item. The basic idea of the mapped\_item is: an item will become part of another item. The assembly component geometry is used as a template in the assembly geometry.
  - The components of an assembly are described together with the construction history. This approach uses the representation\_relationship\_with\_transformation. The transformation describes the relation between different workspaces.
- The usage of both alternatives is considered reasonable, because both mechanisms make sense even in mixed combinations. With regard to the transformations in the context of assembly, a part is in principle incorporated in the assembly only by rigid motion (i.e., translation and/or rotation) excluding mirroring and scaling.
- Package Classification  
A simple basic type of classification of products in STEP works by assigning categories to product data items. These categories are identified by name labels that define the related classification. This type of classification is referred to as specific classification. A specific\_item\_classification\_hierarchy is used to build up hierarchical structures of specific\_item\_classification.
  - Package Properties  
The scope of this subset allows specifying properties associated with parts. A property is the definition of a special quality and may reflect physics, or arbitrary user defined measurements. A general pattern for instantiating property information is in this subset. A number of pre-defined property type names are also proposed for use when appropriate.

A special case of part properties is that of the part shape property - a representation of the geometrical shape model of the part, which are described in section 2.3.4.

- Package Alias\_identification

An alias identification is a mechanism to associate an object with an additional identifier used to identify the object of interest in a different context, either in another organization, or in some other context. The alias identification mechanism shall not be used to alias supplied parts.

The scope of the alias identification shall be specified either by the description of the associated identification\_role or - if the scope is defined by an organization - with help of an applied\_organization\_assignment. The scope of an alias defines the context in which the id specified via applied\_identification\_assignment.assigned\_id overrides the original id. A scenario might be that an object has an id in the context of the organization assigned in the role 'id owner' as a primary id and other ids defined via aliases that are valid in the context of some other organizations.

- Package Authorization

The scope of this subset represents organizations and people in organizations as they perform functions related to other product data and data relationships. A person in this scope must exist in the context of some organization. An organization or a person in an organization is then associated with the data or data relationship in some role indicating the function being performed. Both people and organizations may have addresses associated with them.

Approving in this scope is accomplished by establishing an approval entity and relating it to some construct through an applied\_approval\_assignment. The applied\_approval\_assignment entity may have a role associated with it through the entity role\_association and its related object\_role entity to indicate the reason/role of this approval related to the particular element of product data.

Approval may be represented as a simple basic approval, or it may represent a more complex approval cycle involving multiple provers, on different dates/times, and possibly with different status values.

- Package Configuration\_management

The purpose of this package from the STEP AP214 is meeting the requirements of enterprises that offer many possible configurations of their products for sale. In most cases, the different configurations of a product differ from each other in only minor ways. Configuration identification 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. Because this model is based on the configuration management model defined in STEP AP214, additional information and description of how to use the model can be found in the ARM model and other documentation on AP214.

- Package Change\_and\_work\_management

This package describes the process by which companies request, implement, and effect change to products, documents, components, assemblies, manufactured or purchased parts, processes, or even suppliers. This subset provides the capability to represent activity, project, and contract related information. Activities may be initiated by work requests and may be authorized by work orders. Activities may result in changes of models or of properties; such changes can also be represented.

- Package Process\_planning

This package provides the capability to represent process related data. This includes process plans, versions of process plans with version tracking, process operations, and properties of processes. A process plan is decomposed into one or more occurrences of process operations. Process plans and process operations establish relationships among raw materials, in-process items, and final items, as well as the relationship between the items and the tools used to manufacture them. Additionally, the representation of the connection of parts in various kinds of mating is part of this package.

- Package Multi\_language\_support

This package provides the capability to represent descriptive information about objects in different languages.

## 8.1 Package PLM\_base

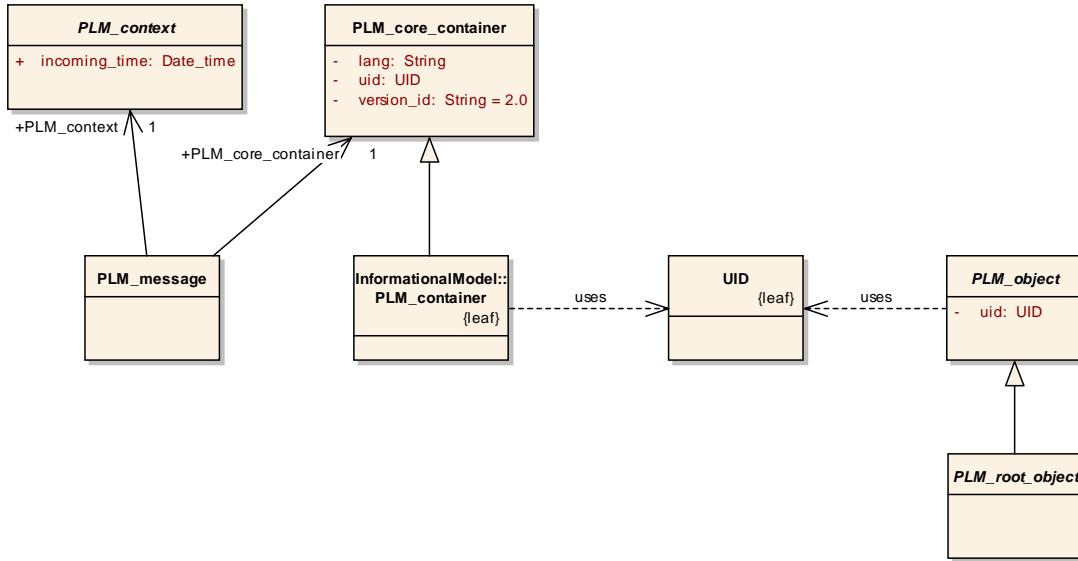


Figure 8.1 - PLM base

### 8.1.1 Classes

#### 8.1.1.1 Class PLM\_core\_container

All operations in this specification use the type PLM\_core\_container as input parameter type or return type when PLM data has to be transferred. So, the PLM\_core\_container serves as a container to transfer arbitrary PLM data.

##### Base Class

- none

##### Attributes

- lang:String[1]:  
Language identifying string
- uid:UID  
Unique identifier
- version\_id:String[1]=2.0:  
Indicates the version number of the PLM Services specification this container is compliant to.

#### 8.1.1.2 Class PLM\_container

The PLM\_container class is introduced to ensure that data is only handled by the computational model in a valid way.

##### Base Class

- PLM\_core\_container

## **Attributes**

- none

## **Compositions**

- accuracy : Accuracy [0..\*]
- activity : Activity [0..\*]
- activity\_method : Activity\_method [0..\*]
- address : Address [0..\*]
- application\_context : Application\_context [0..\*]
- approval\_status : Approval\_status [0..\*]
- axis2\_placement\_3d : Axis2\_placement\_3d [0..\*]
- axis\_placement : Axis\_placement [0..\*]
- cartesian\_coordinate\_space : Cartesian\_coordinate\_space (ABS) [0..\*]
- cartesian\_point : Cartesian\_point [0..\*]
- certification : Certification [0..\*]
- classification\_attribute : Classification\_attribute [0..\*]
- classification\_system : Classification\_system [0..\*]
- complex\_product : Complex\_product (ABS) [0..\*]
- contract : Contract [0..\*]
- data\_environment : Data\_environment [0..\*]
- date\_time : Date\_time [0..\*]
- descriptive\_specification : Descriptive\_specification [0..\*]
- design\_constraint : Design\_constraint [0..\*]
- direction : Direction [0..\*]
- document : Document [0..\*]
- document\_content\_property : Document\_content\_property [0..\*]
- document\_creation\_property : Document\_creation\_property [0..\*]
- document\_file : Document\_file (ABS) [0..\*]
- document\_format\_property : Document\_format\_property [0..\*]
- document\_location\_property : Document\_location\_property [0..\*]
- document\_size\_property : Document\_size\_property [0..\*]
- document\_type\_property : Document\_type\_property [0..\*]
- duration : Duration [0..\*]
- effectivity : Effectivity [0..\*]
- event\_reference : Event\_reference [0..\*]
- external\_library\_reference : External\_library\_reference [0..\*]
- general\_classification : General\_classification [0..\*]
- geometric\_model : Geometric\_model [0..\*]



- interval\_of\_time : Interval\_of\_time [0..\*]
- item : Item [0..\*]
- item\_shape : Item\_shape [0..\*]
- language : Language [0..\*]
- material : Material [0..\*]
- model\_property\_value
- organization : Organization [0..\*]
- person : Person [0..\*]
- physical\_instance : Physical\_instance [0..\*]
- plib\_class\_reference
- plib\_property\_reference
- process\_operation\_definition : Process\_operation\_definition [0..\*]
- process\_plan : Process\_plan [0..\*]
- product\_class : Product\_class [0..\*]
- project : Project [0..\*]
- property : Property (ABS) [0..\*]
- property\_value : Property\_value (ABS) [0..\*]
- rectangular\_size : Rectangular\_size [0..\*]
- retention\_period : Retention\_period [0..\*]
- security\_level : Security\_level [0..\*]
- shape\_dependent\_property : Shape\_dependent\_property [0..\*]
- specific\_document\_classification : Specific\_document\_classification [0..\*]
- specific\_item\_classification : Specific\_item\_classification [0..\*]
- specification : Specification [0..\*]
- specification\_category : Specification\_category [0..\*]
- specification\_expression : Specification\_expression [0..\*]
- transformation : Transformation (ABS) [0..\*]
- unit : Unit [0..\*]
- work\_order : Work\_order [0..\*]
- work\_request : Work\_request [0..\*]

### **Associations**

- none

#### **8.1.1.3 Class PLM\_context (ABS)**

The PLM\_context is an abstract base type for the context of a PLM\_message, e.g., if the message is part of a workflow, the context describes the position of the message in the workflow.

### **Base Class**

- none

### **Attributes**

- incoming\_time : Date\_time [1]  
The attribute incoming\_time is set by the PLM\_message\_connection implementation with the time when the containing PLM\_message object was received as parameter of the operation write\_messages.

### **Compositions**

- none

### **Associations**

- none

#### **8.1.1.4 Class PLM\_message**

The PLM\_message is a base type for a message exchange based client service communication and is used by the PLM\_message\_connection service in the Computational Viewpoint (see Chapter 9 ). The PLM\_context describes the position of the message in a workflow and the PLM\_core\_container contains the user data of the message.

### **Base Class**

- none

### **Attributes**

- none

### **Compositions**

- PLM\_context : PLM\_context [1]  
The PLM\_context describes relevant data to identify the message, to associate it to a particular workflow and to determine the position of the message in a workflow.
- PLM\_core\_container : PLM\_core\_container [1]  
The PLM\_core\_container contains the pay load of the message.

### **Associations**

- none

#### **8.1.1.5 Class PLM\_object (ABS)**

The abstract PLM\_object class is introduced to provide a mechanism of binding a unique identifier to each PLM class instance. These identifiers must be valid and unique throughout a complete session defined by the computational model. After each session the identifiers may be invalid.

### **Base Class**

- none

### **Attributes**

- uid : UID [1]  
The uid uniquely identifies an object throughout a complete session defined by the computational model. After each session the uid is invalid.

### **Compositions**

- none

### **Associations**

- none

#### **8.1.1.6 Class PLM\_root\_object (ABS)**

The abstract class PLM\_root\_object is defined to distinguish between types that can be directly inserted into PLM\_container instances and types that are contained in the container through PLM\_root\_object instances.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

## **8.1.2 Datatypes**

Datatype Boolean

Datatype Double

Datatype Integer

Datatype String

Datatype UID

## 8.2 Package Part\_identification

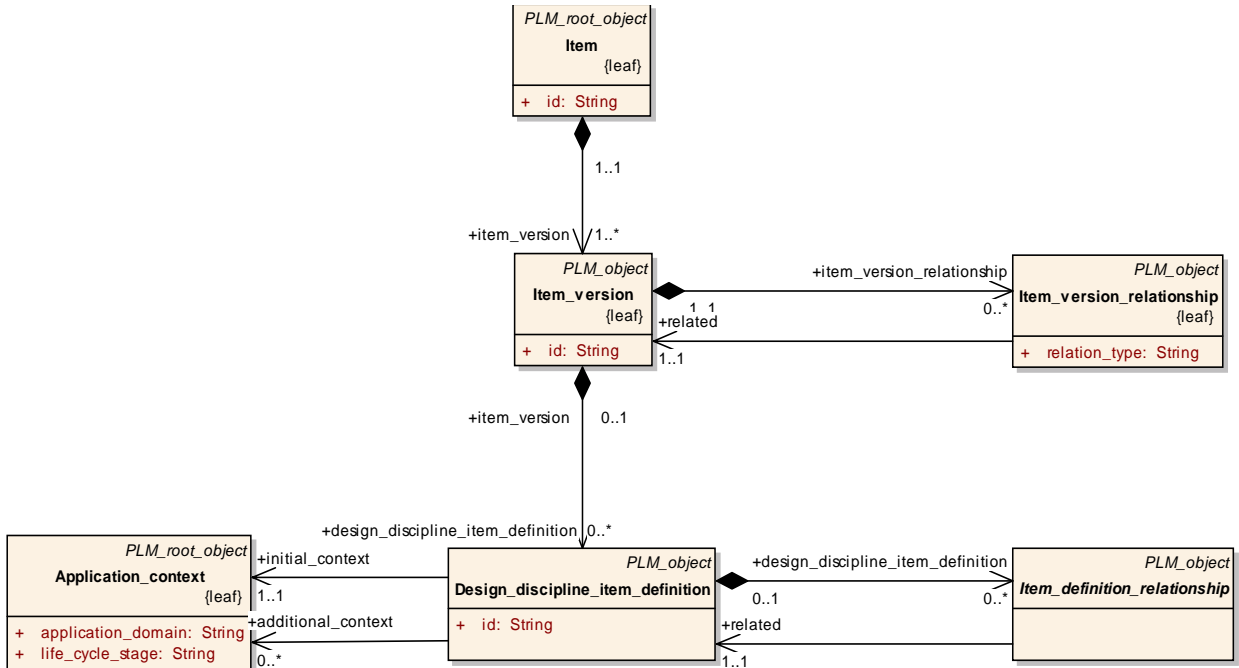


Figure 8.2 - Part identification

### 8.2.1 Classes

#### 8.2.1.1 Class Application\_context

An Application\_context is a shared universe of discourse.

##### Base Class

- PLM\_root\_object (ABS)

##### Attributes

- application\_domain : String [1]

The application\_domain is the identification of the applications for which an object may be relevant. Where applicable the following values shall be used:

- ‘assembly study’ The object may be relevant for an assembly study.
- ‘digital mock-up’ The object may be relevant for digital mock-up.
- ‘electrical design’ The object may be relevant for the electrical design.

'mechanical design'	The object may be relevant for the mechanical design.
'preliminary design'	The object may be relevant for the preliminary design.
'process planning'	The object may be relevant for the process planning.

- `life_cycle_stage` : String [1]  
The `life_cycle_stage` is the specification of the general stage in the product life cycle to which the concerned items belong. Where applicable the following values shall be used:

'design'	The concerned item belongs to the design phase of the life cycle.
'manufacturing'	The concerned item belongs to the manufacturing phase of the life cycle
'recycling'	The concerned item belongs to the recycling phase of the life cycle.

### **Compositions**

- `description` : String\_select [0..1]  
The description specifies additional information about the `Application_context`.

### **Associations**

- none

#### **8.2.1.2 Class Design\_discipline\_item\_definition**

A `Design_discipline_item_definition` is a view of an `Item_version`. This view is relevant for the requirements of one or more life cycle stages and application domains and collects product data of the `Item_version`.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `id` : String [1]  
The `id` specifies the identifier of the `Design_discipline_item_definition`.

### **Compositions**

- `item_instance` : `Item_instance` (ABS) [0..\*]  
The `item_instance` specifies the `item_instance` that is defined by this `Design_discipline_item_definition`.
- `item_definition_relationship` : `Item_definition_relationship` (ABS) [0..\*]  
The `item_definition_relationship` specifies the `Item_definition_relationship` that relates the first of the two `Design_discipline_item_definition` objects.
- `name` : String\_select [0..1]  
The `name` specifies the word or group of words used to refer to the `Design_discipline_item_definition`.
- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Design_discipline_item_definition`.

- **item\_function\_association** : Item\_function\_association [0..\*]  
The item\_function\_association specifies the Item\_function\_association that this Design\_discipline\_item\_definition is associated with.
- **alias\_identification** : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Design\_discipline\_item\_definition.
- **simple\_property\_value** : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.
- **Item\_definition\_instance\_relationship** : Item\_definition\_instance\_relationship (ABS) [0..\*]  
The Item\_definition\_instance\_relationship specifies the Item\_definition\_instance\_relationship that this Design\_discipline\_item\_definition is part of. If the Design\_discipline\_item\_definition is an Assembly\_definition, the relationship shall be an Assembly\_component\_relationship. If the Design\_discipline\_item\_definition is a Collection\_definition, the relationship shall be a Collected\_item\_association.

### **Associations**

- **initial\_context** : Application\_context [1]  
The initial\_context specifies the Application\_context in which this view of the Item\_version has been designed primarily.
- **additional\_context** : Application\_context [0..\*]  
The additional\_context specifies the set of Application\_context objects in which this view of the Item\_version is also relevant. The additional\_context shall not contain the Application\_context that is referenced as the 'initial\_context.'

### **8.2.1.3 Class Item**

An Item is either a single object or a unit in a group of objects. It collects the information that is common to all versions of the object. An Item shall always be classified as 'part,' 'tool,' or 'raw material' using a Specific\_item\_classification. Additionally, if an Assembly\_definition exists for at least one version of the Item, the Item shall be classified as being an 'assembly' using Specific\_item\_classification.

### **Base Class**

- **PLM\_root\_object** (ABS)

### **Attributes**

- **id** : String [1]  
The id specifies the identifier of the Item. For the id, an owner shall be specified by a Person\_organization\_assignment with role 'id owner.' The id shall be unique within the scope of the organization that is specified by the Person\_organization\_assignment with the role 'id owner.'

### **Compositions**

- **item\_version** : Item\_version [1..\*]  
The item\_version specifies the Item\_version that is associated with this Item.
- **description** : String\_select [0..1]  
The description specifies additional information about the Item.
- **name** : String\_select [1]  
The name specifies the word or group of words used to refer to the Item.

- `alias_identification` : `Alias_identification` [0..\*]  
The `Alias_identification` specifies the `Alias_identification` that is applied to this Item.
- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this Item.

### **Associations**

- none

#### **8.2.1.4 Class `Item_definition_relationship` (ABS)**

An `Item_definition_relationship` is a relationship between two `Design_discipline_item_definition` objects.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- none

### **Compositions**

- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Item_definition_relationship`.
- `simple_property_value` : `Simple_property_value` (ABS) [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- `related` : `Design_discipline_item_definition` [1]  
The `related` specifies the second of the `Design_discipline_item_definition` objects that are part of the relationship.

#### **8.2.1.5 Class `Item_version`**

An `Item_version` is a version of an Item and serves as the collector of the data characterizing a physically realizable object in various application contexts.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `id` : `String` [1]  
The `id` specifies the identifier of the `Item_version`. The `id` shall be unique within the scope of the associated Item.

### **Compositions**

- `item_version_relationship` : `Item_version_relationship` [0..\*]  
The `item_version_relationship` specifies the `item_version_relationship` that relates the first of the two `Item_version` objects.

- **description** : String\_select [0..1]  
The description specifies additional information about the Item\_version.
- **product\_design** : Product\_design [0..1]  
The product\_design specifies the Product\_design for which the Item\_version meets the requirements.
- **design\_discipline\_item\_definition** : Design\_discipline\_item\_definition [0..\*]  
The design\_discipline\_item\_definition specifies the Design\_discipline\_item\_definition that is a view for this Item\_version.
- **alias\_identification** : Alias\_identification [0..\*]  
The alias\_identification specifies the Alias\_identification that is applied to this Item\_version.
- **document\_assignment** : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Item\_version.

### **Associations**

- none

### **8.2.1.6 Class Item\_version\_relationship**

An Item\_version\_relationship is a relationship between two Item\_version objects.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- **relation\_type** : String [1]  
The relation\_type specifies the meaning of the relationship. Where applicable, the following values shall be used:

‘derivation’	The application object defines a deriving relationship where the related Item_version is based on the relating Item_version that is an earlier version of the same or of a different Item.
‘hierarchy’	The application object defines a hierarchical relationship where the related Item_version is a subordinate version of the relating Item_version.
‘sequence’	The application object defines a version sequence where the relating Item_version is the preceding version of the related Item_version that is the following version. For a given Item_version there shall be at most one Item_version_relationship of this relation_type referring to this Item_version as ‘relating’ and at most one Item_version_relationship of this relation_type referring as ‘related.’
‘supplied item’	The application object defines a relationship between two Item_version objects representing the same object in different organizational contexts.

#### **Compositions**

- **description** : String\_select [0..1]  
The “description” specifies additional information about the Item\_version\_relationship.



- change : Change [0..\*]  
The “change” specifies the change for which this object references a modified object and the corresponding original object.

### Associations

- related : Item\_version [1]  
The “related” specifies the second of the two Item\_version objects related by the Item\_version\_relationship.

## 8.3 Package Part\_structure

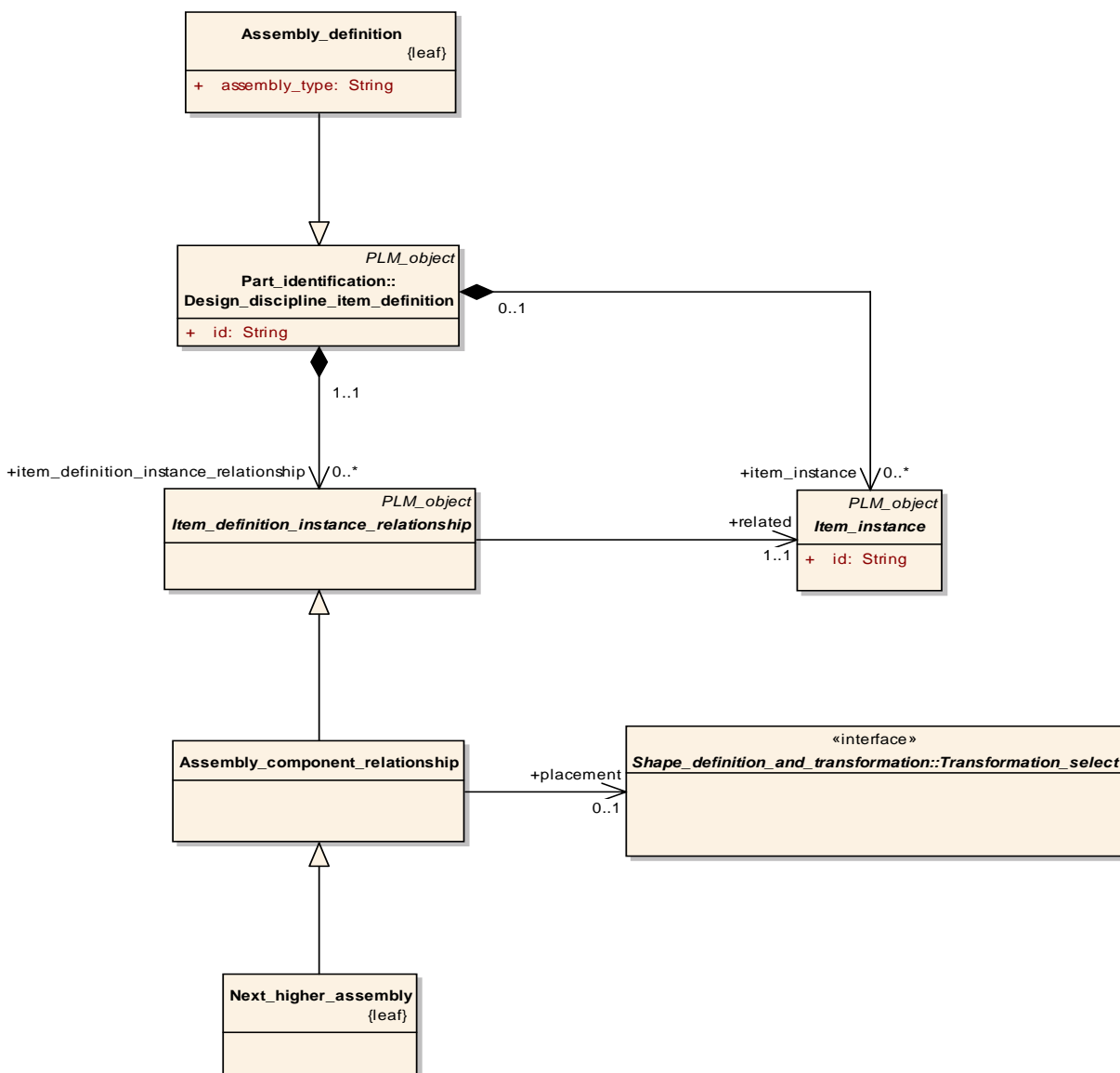


Figure 8.3 - Part structure - Assembly

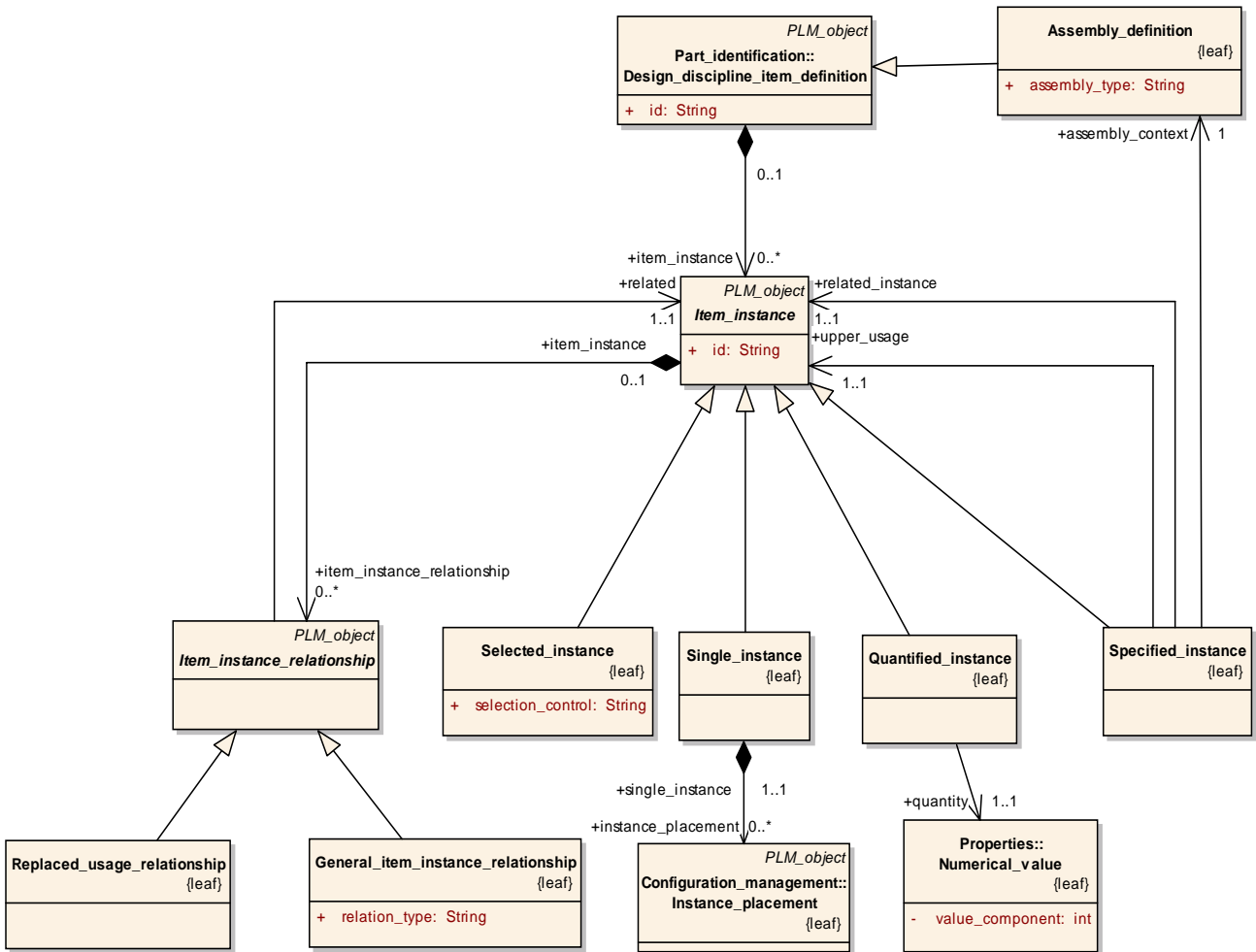


Figure 8.4 - Part structure - Item instance

### 8.3.1 Classes

#### 8.3.1.1 Class Alternate\_item\_relationship

An Alternate\_item\_relationship is the relationship between two Item objects specifying that all versions of the alternate Item may be used in place of the base Item, regardless of its usages and of its versions.

NOTE: This concept usually refers to form, fit, function, and quality. Additional properties such as performance, noise, endurance, or reliability may also be considered as a prerequisite for the replacement.

#### Base Class

- PLM\_object (ABS)

### **Attributes**

- fulfilled\_requirements: string[1]:  
The fulfilled\_requirements specify the word or group of words describing the requirements that are covered by both the base and the alternate and therefore are the basis for the statement regarding the capability of replacement.

### **Compositions**

#### **Associations**

- alternate: Item[1]:  
The alternate specifies the Item that may be used in place of the base Item.

### **8.3.1.2 Class Assembly\_component\_relationship**

An Assembly\_component\_relationship is the relation between an Assembly\_definition and an Item\_instance representing a constituent of the assembly. The Assembly\_definition and the Design\_discipline\_item\_definition that serves as 'definition' of the Item\_instance shall share at least one Application\_context.

#### **Base Class**

- Item\_definition\_instance\_relationship (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- placement : Transformation\_select [0..1]  
The placement specifies the Geometric\_model\_relationship\_with\_transformation or the Template\_instance that specifies the transformation information used to locate and orient the constituent in the coordinate space of the Assembly\_definition. In the case of a Template\_instance, the scale factor shall be omitted or set to 1.0.

### **8.3.1.3 Class Assembly\_definition**

An Assembly\_definition is a definition of an Item\_version that contains other subordinate objects.

#### **Base Class**

- Design\_discipline\_item\_definition

#### **Attributes**

- assembly\_type : String [0..1]  
The assembly\_type specifies the kind of the Assembly\_definition.

#### **Compositions**

- none

### **Associations**

- none

#### **8.3.1.4 Class Assembly\_substitute\_relationship**

An Assembly\_substitute\_relationship is a relationship that indicates that one Assembly\_component\_relationship may be substituted for another Assembly\_component\_relationship. The subjects of the substitution are the related Item\_instance objects of both Assembly\_component\_relationship objects. The relating Assembly\_definition shall be the same in both Assembly\_component\_relationship objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

### **Compositions**

- description: String\_selectString\_select[0..1]:  
The description specifies additional information about the Assembly\_substitute\_relationship.

### **Associations**

- substitute: Assembly\_component\_relationship[1]:  
The substitute specifies the Assembly\_component\_relationship that may be used instead of the base Assembly\_component\_relationship.

#### **8.3.1.5 Class Collected\_item\_association**

A Collected\_item\_association is a mechanism to associate Item\_instance objects with a Collection\_definition.

### **Base Class**

- Item\_definition\_instance\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

#### **8.3.1.6 Class Collection\_definition**

A Collection\_definition is the definition of an Item\_version that serves as a collector for Item\_instance objects that are mounted in the same vehicle but may not be assembled together.

### **Base Class**

- Design\_discipline\_item\_definition

### **Attributes**

- none

### **Compositions**

- purpose : String\_select [0..1]  
The purpose specifies the rationale behind the Collection\_definition.

### **Associations**

- none

### **8.3.1.7 Class General\_item\_definition\_instance\_relationship**

A General\_item\_definition\_instance\_relationship is a relationship between a Design\_discipline\_item\_definition and an Item\_instance whose meaning is defined by the attribute 'relation\_type.'

### **Base Class**

- Item\_definition\_instance\_relationship (ABS)

### **Attributes**

- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the General\_item\_definition\_instance\_relationship.

### **Associations**

- none

### **8.3.1.8 Class General\_item\_definition\_relationship**

A General\_item\_definition\_relationship is a relationship between two Design\_discipline\_item\_definition objects whose meaning is defined by the attribute 'relation\_type.'

### **Base Class**

- Item\_definition\_relationship (ABS)

### **Attributes**

- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the General\_item\_definition\_relationship.

### **Associations**

- none

#### **8.3.1.9 Class General\_item\_instance\_relationship**

A General\_item\_instance\_relationship is a relationship between two Item\_instance objects whose meaning is defined by the attribute 'relation\_type.'

### **Base Class**

- Item\_instance\_relationship (ABS)

### **Attributes**

- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the General\_item\_instance\_relationship.

### **Associations**

- none

#### **8.3.1.10 Class Item\_definition\_instance\_relationship (ABS)**

An Item\_definition\_instance\_relationship is a relationship between a Design\_discipline\_item\_definition and an Item\_instance.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Item\_definition\_instance\_relationship.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- related : Item\_instance (ABS) [1]  
The related specifies the Item\_instance that is part of the Item\_definition\_instance\_relationship.

#### **8.3.1.11 Class Item\_instance (ABS)**

An Item\_instance is the occurrence of an object in a product structure that is defined either by a Design\_discipline\_item\_definition or by a Product\_identification.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- id : String [1]  
The id specifies the identifier of the Item\_instance.

### **Compositions**

- item\_instance\_relationship : Item\_instance\_relationship (ABS) [0..\*]  
The item\_instance\_relationship specifies the item\_instance\_relationship that relates the first of the two Item\_instance objects.
- description : String\_select [0..1]  
The description specifies additional information about the Item\_instance.
- manufacturing\_configuration : Manufacturing\_configuration (ABS) [0..\*]  
The Manufacturing\_configuration specifies the Manufacturing\_configuration that controls this Item\_instance.
- configuration : Configuration [0..\*]  
The configuration specifies the configuration that controls this Item\_instance for its valid usage.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Item\_instance.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Item\_instance.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- none

#### **8.3.1.12 Class Item\_instance\_relationship (ABS)**

An Item\_instance\_relationship is a relationship between two Item\_instance objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Item\_instance\_relationship.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- related : Item\_instance (ABS) [1]  
The related specifies the second of the two objects related by the Item\_instance\_relationship.

### **8.3.1.13 Class Make\_from\_relationship**

A Make\_from\_relationship is a relationship between a Design\_discipline\_item\_definition that provides the definition of a raw material, or of a semi-finished item and a Design\_discipline\_item\_definition that provides the definition of an object manufactured out of that material, or semi-finished item.

### **Base Class**

- Item\_definition\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Make\_from\_relationship.

### **Associations**

- none

### **8.3.1.14 Class Next\_higher\_assembly**

A Next\_higher\_assembly is a relationship where the attribute 'related' specifies a constituent of an assembly and the attribute 'relating' specifies the immediate parent assembly of the constituent.

### **Base Class**

- Assembly\_component\_relationship

### **Attributes**

- none

### **Compositions**

- none



### **Associations**

- none

#### **8.3.1.15 Class Physical\_assembly\_relationship**

A Physical\_assembly\_relationship is a mechanism to relate one Physical\_instance as a component to another Physical\_instance that plays the role of an assembly.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Physical\_assembly\_relationship.

### **Associations**

- physical\_component : Physical\_instance [1]  
The physical\_component specifies the Physical\_instance that serves as a component in the physical structure.
- is\_realization\_of : Item\_instance (ABS) [1]  
The is\_realization\_of specifies the Item\_instance the physical component is the realization of.

#### **8.3.1.16 Class Promissory\_usage**

A Promissory\_usage is a relationship between a constituent of an assembly and the assembly itself. The relationship describes the intention to use the constituent in an assembly.

NOTE: There may not be any geometric information provided for the constituent or the assembly that are associated by a Promissory\_usage.

EXAMPLE: In early design stages, the Promissory\_usage may be used to create preliminary BOMs for prototyping.

A Promissory\_usage is a type of Assembly\_component\_relationship.

### **Base Class**

- Assembly\_component\_relationship

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

#### **8.3.1.17 Class Quantified\_instance**

- A Quantified\_instance is the identification of the quantified occurrence of an object that is defined either as a Design\_discipline\_item\_definition or as a Product\_identification.

### **Base Class**

- Item\_instance (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- quantity : Numerical\_value [1]  
The quantity specifies a Numerical\_value specifying the quantity of occurrences.

#### **8.3.1.18 Class Replaced\_definition\_relationship**

A Replaced\_definition\_relationship is a relationship between two Design\_discipline\_item\_definition objects where the relating Design\_discipline\_item\_definition is replaced by the related Design\_discipline\_item\_definition.

### **Base Class**

- Item\_definition\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- change : Change [0..\*]  
The change specifies the change for which this object references a modified object and the corresponding original object.
- description : String\_select [0..1]  
The description specifies additional information about the Replaced\_definition\_relationship.

### **Associations**

- none

#### **8.3.1.19 Class Replaced\_usage\_relationship**

A Replaced\_usage\_relationship is a relationship between two Item\_instance objects where the relating Item\_instance is replaced by the related Item\_instance.

### **Base Class**

- Item\_instance\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Replaced\_usage\_relationship.

### **Associations**

- usage\_context : Instance\_usage\_context\_select [1]  
The usage\_context specifies the object that identifies the context in which the replacement is applicable. In the case where the usage\_context refers to a Process\_operation\_input\_or\_output, the 'relating' Item\_instance shall be referred to as 'element' by the Process\_operation\_input\_or\_output. In the case where the usage\_context refers to an Item\_definition\_instance\_relationship, the 'relating' Item\_instance shall be referred to as 'related' by the Item\_definition\_instance\_relationship. In the case where the usage\_context refers to a Product\_structure\_relationship, the 'relating' Item\_instance shall be referred to as 'related' by the Product\_structure\_relationship.

### **8.3.1.20 Class Selected\_instance**

A Selected\_instance is the identification of the occurrence of an object that is either defined as a Design\_discipline\_item\_definition or as a Product\_identification and whose quantity depends on certain constraints.

### **Base Class**

- Item\_instance (ABS)

### **Attributes**

- selection\_control : String [1]  
The selection\_control specifies the constraint that has to be evaluated for the Selected\_instance.

### **Compositions**

- none

### **Associations**

- selected\_quantity : Value\_with\_unit (ABS) [1]  
The selected\_quantity specifies the quantity of the part, tool, or raw material foreseen as Selected\_instance. The selected\_quantity shall be of type Value\_limit or Value\_range.

### **8.3.1.21 Class Single\_instance**

A Single\_instance is one particular occurrence of an object that is defined either as a Design\_discipline\_item\_definition or as a Product\_identification.

### **Base Class**

- Item\_instance (ABS)

### **Attributes**

- none

### **Compositions**

- instance\_placement : Instance\_placement [0..\*]  
The instance\_placement specifies the instance\_placement that this Single\_instance is placed with.

### **Associations**

- none

### **8.3.1.22 Class Specified\_instance**

A Specified\_instance is a mechanism to identify a certain Item\_instance in a multi level assembly structure that reuses partial decompositions.

### **Base Class**

- Item\_instance (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- assembly\_context : Assembly\_definition [1]  
The assembly\_context specifies an Assembly\_definition object in which the instance identified by this mechanism is used.
- related\_instance : Item\_instance (ABS) [1]  
The related\_instance specifies the Item\_instance that is to be identified.
- upper\_usage : Item\_instance (ABS) [1]  
The upper\_usage specifies the Item\_instance in which the related\_instance is used. This Item\_instance shall be the immediate upper level instance or another Specified\_instance.

### **8.3.1.23 Class Tool\_part\_relationship**

A Tool\_part\_relationship is a relationship between two Design\_discipline\_item\_definition objects. It establishes a relationship between an item (related) and a tool (relating) used to produce the item.

### **Base Class**

- Item\_definition\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- used\_technology\_description : String\_select [0..1]  
The used\_technology\_description specifies the technology used to manufacture the part using this tool and, possibly, the reasons for the use of a particular technology.

### **Associations**

- placement : Transformation\_target\_select [0..1]  
The placement specifies the relative position of the Item representing the part with respect to the local coordinate system of the Item representing the tool.

## **8.3.2 Interfaces**

### **8.3.2.1 Interface Instance\_usage\_context\_select**

This empty interface is realized by the following classes:

- Product\_structure\_relationship
- Item\_definition\_instance\_relationship (ABS)
- Process\_operation\_input\_or\_output

### **8.3.2.2 Interface Item\_information\_select**

This empty interface is realized by the following classes:

- Product\_component
- Physical\_instance
- Design\_discipline\_item\_definition
- Item\_instance (ABS)

### **8.3.2.3 Interface Product\_constituent\_select**

This empty interface is realized by the following classes:

- Product\_function
- Product\_component
- Item\_instance (ABS)

## 8.4 Package Document\_and\_file\_management

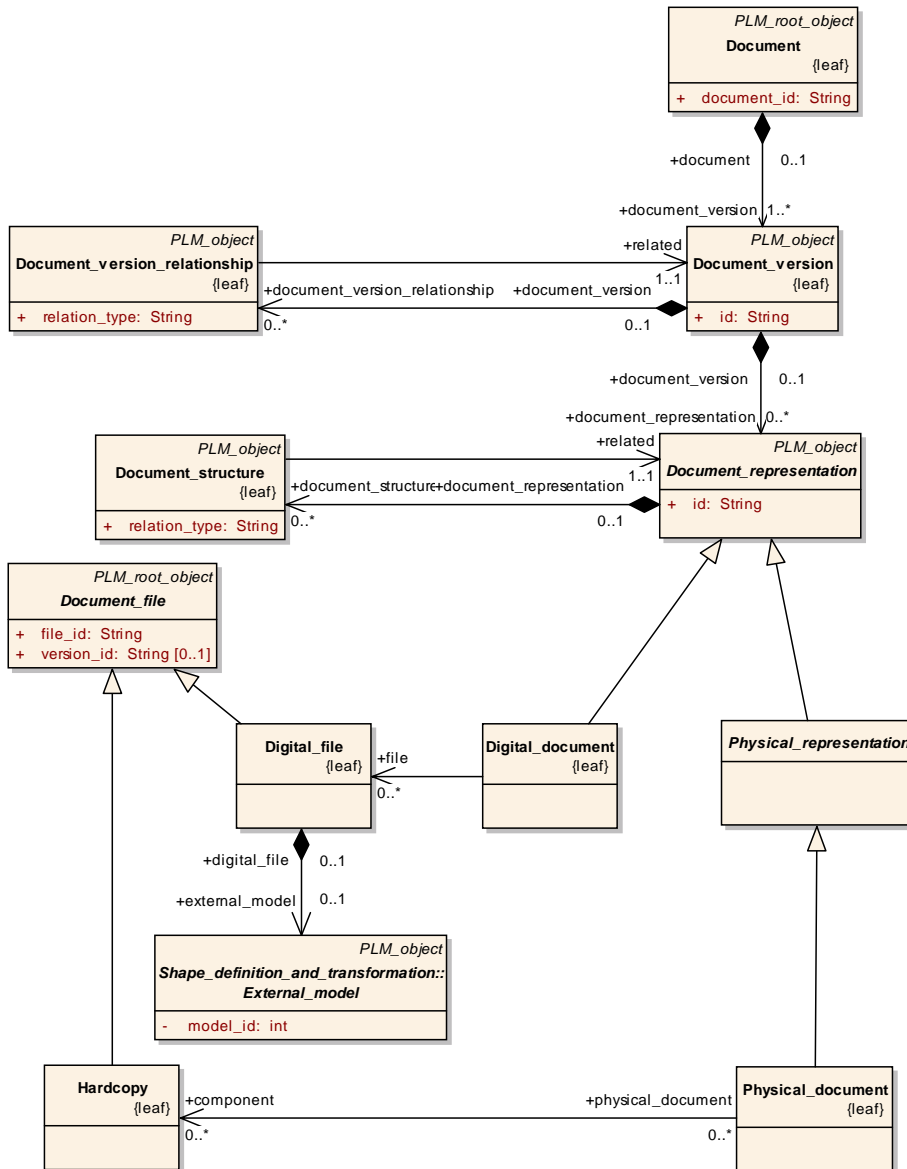


Figure 8.5 - Document and file management

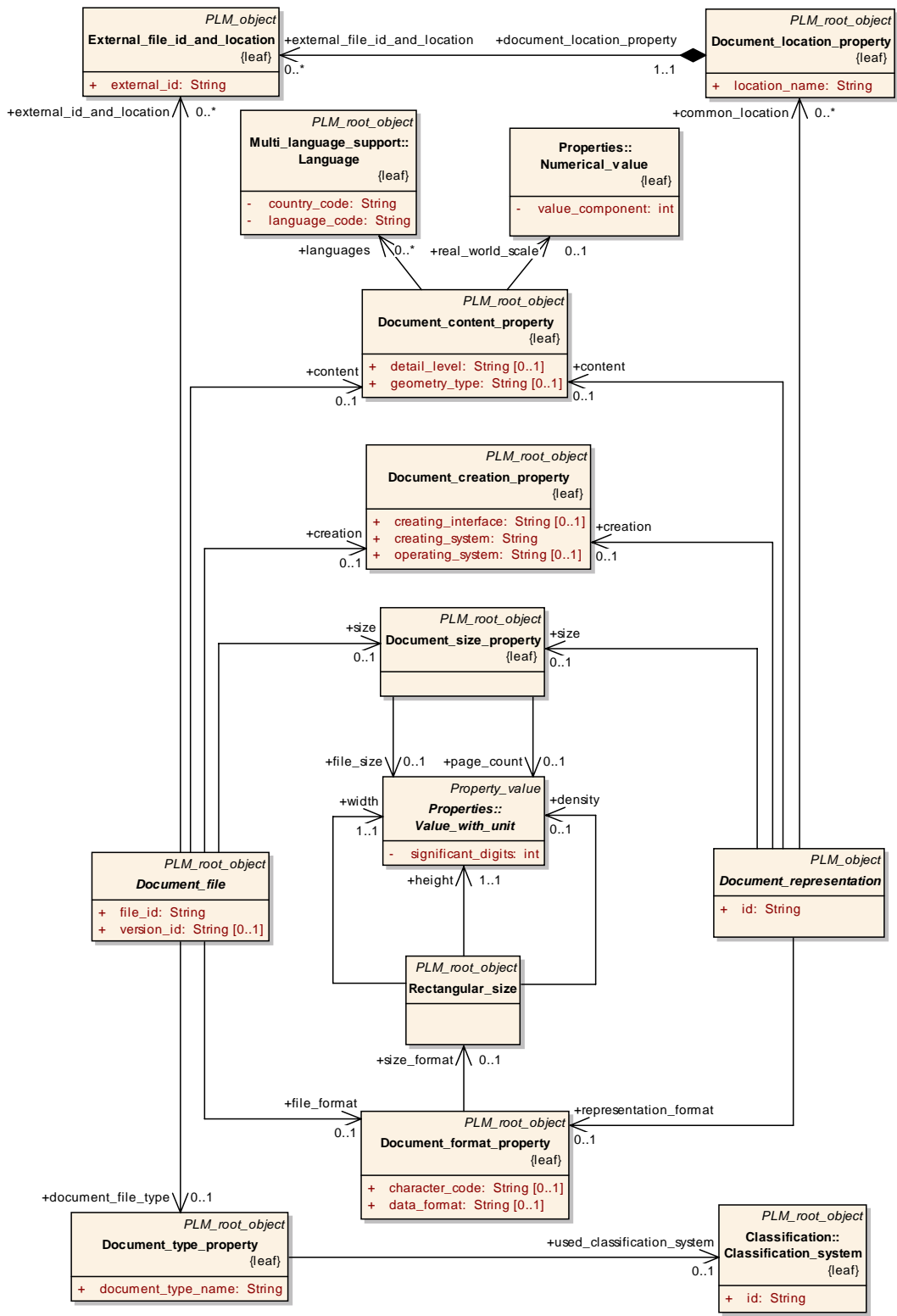


Figure 8.6 - Document properties

## 8.4.1 Classes

### 8.4.1.1 Class Digital\_document

A Digital\_document is a piece of product data that is archived in a digital format.

#### **Base Class**

- Document\_representation (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- file : Digital\_file [0..\*]  
The file specifies a computer interpretable realization of the Digital\_document.

### 8.4.1.2 Class Digital\_file

A Digital\_file contains computer interpretable data.

#### **Base Class**

- Document\_file (ABS)

#### **Attributes**

- none

#### **Compositions**

- external\_model : External\_model (ABS) [0..1]  
The external\_model specifies the externally defined geometry information contained in this Digital\_file.

#### **Associations**

- none

### 8.4.1.3 Class Document

A Document is a logical document that serves as the identifier for a container for some product data.

#### **Base Class**

- PLM\_root\_object (ABS)



### **Attributes**

- document\_id : String [1]  
The document\_id specifies the identifier of the Document.

### **Compositions**

- document\_version : Document\_version [1..\*]  
The document\_version specifies the document\_version of this logical document.
- name : String\_select [1]  
The name specifies the word or group of words by which the Document is referred to.
- description : String\_select [0..1]  
The description specifies additional information about the Document.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Document.

### **Associations**

- none

#### **8.4.1.4 Class Document\_assignment**

A Document\_assignment is a mechanism to associate a document with an object, where the assigned document provides information about the object it is associated to.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- role : String [1]  
The role specifies the meaning of the Document\_assignment:

'additional information'	The assigned document provides information that is relevant for the associated object, but is not a description of the associated object itself.
'behavior'	The assigned document specifies information about the behavior of the associated object.
'description'	The assigned document provides textual information for the associated object itself.
'informative'	The assigned document may or may not be considered.
'mandatory'	The associated object shall conform to the content of the assigned document.
'mathematical description'	The assigned document specifies the associated object by providing the algorithmic specification of its behavior.

## Compositions

- none

## Associations

- assigned\_document : Assigned\_document\_select [1]  
The assigned\_document specifies the Document, a Document\_version, a Document\_representation, or a Document\_file used to provide information.
- is\_assigned\_to : Documented\_element\_select [1..\*]  
The documented elements.

### 8.4.1.5 Class Document\_content\_property

A Document\_content\_property specifies characteristics precisising the content of a Document\_file or of a Document\_representation. At least one of the optional attributes shall be specified for each instance of this object.

#### Base Class

- PLM\_root\_object (ABS)

#### Attributes

- detail\_level : String [0..1]  
The detail\_level specifies the level of detail that the Document\_file or the Document\_representation provides. Where applicable the following values shall be used:

'rough 3d shape'	3D shape model without edge rounds and fillets
'rounded edges'	3D shape model with edge rounds and fillets

- geometry\_type : String [0..1]  
The geometry\_type specifies the kind or kinds of geometry that an object contains. Where applicable the following values shall be used:

'3D wireframe model'	The document contains a 3D shape model in wireframe representation.
'2D shape'	The document contains a 2D shape model or contours only.
'surface model'	The document contains a 3D shape model in surface representation.
'closed volume'	The document contains a 3D shape model in closed body topological surface representation.
'solid model'	The document contains a 3D shape model in advanced boundary representation.
'solid and surface model'	The document contains a 3D shape model in surface and advanced boundary representation.
'assembly'	The document contains an assembly structure with reference to the assembled components and their transformation matrices.

'assembly with mating elements'	The document contains an assembly structure including the mating components only, such as screws or rivets, with exact positioning information. This assembly representation is intended to be overlaid with the assembly structure for the main components.
'2D drawing'	The document contains a technical drawing without 3D shape representation.
'drawing derived from 3D data'	The document contains a technical drawing that has been derived from a 3D shape model.
'drawing related to 3D data'	The document contains a technical drawing that visualizes a 3D shape model and possibly establishes associative links to the 3D shape model.

### **Compositions**

- none

### **Associations**

- languages : Language [0..\*]  
The languages specify which language or languages are used in the characterized objects.
- real\_world\_scale : Numerical\_value [0..1]  
The real\_world\_scale specifies the scale used in the Document\_file or in the Document\_representation the Document\_content\_property is referred by.

#### **8.4.1.6 Class Document\_creation\_property**

A Document\_creation\_property specifies characteristics of Document\_file or of Document\_representation objects. It specifies the context of the creation of the object. At least one of the optional attributes shall be specified for each instance of this object.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- creating\_system : String [1]  
The creating\_system specifies the computer application or the machine used to create the object that is characterized.
- operating\_system : String [0..1]  
The operating\_system specifies the operating system used to execute the computer application that created the characterized object.
- creating\_interface : String [0..1]  
The creating\_interface specifies the computer application used to create the Document\_file or Document\_representation object.

### **Compositions**

- none

### **Associations**

- none

#### **8.4.1.7 Class Document\_file (ABS)**

A Document\_file is one of potentially more files on a computer system or in actual stacks of paper that make up a Document\_representation.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- file\_id : String [1]  
The file\_id specifies the identifier used to locate the file either on a computer system or in a repository of paper documents.
- version\_id : String [0..1]  
The version\_id specifies the identification of the version that distinguishes one Document\_file object from other versions of Document\_file objects with the same file\_id.
- description : String [0..1]  
The description specifies additional information about the Document\_file. The description need not be specified for a particular Document\_file. If present, there shall be exactly one object that defines the description for a Document\_file.

### **Compositions**

- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- creation : Document\_creation\_property [0..1]  
The creation specifies further details of the context of the creation of the Document\_file.
- content : Document\_content\_property [0..1]  
The content characterizes the content of the Document\_file.
- file\_format : Document\_format\_property [0..1]  
The file\_format specifies the characteristics of the Document\_file that specifies the format of the object.
- size : Document\_size\_property [0..1]  
The size specifies characteristics for the size of the Document\_file.
- external\_id\_and\_location : External\_file\_id\_and\_location [0..\*]  
The external\_id\_and\_location specifies alternatives of the identifier and location of the Document\_file.
- document\_file\_type : Document\_type\_property [0..1]  
The document\_file\_type specifies the format of the Document\_file. It shall only be specified, if the Document\_file does not participate in a Document.

#### 8.4.1.8 Class Document\_file\_relationship

A Document\_file\_relationship is a relationship between two Document\_file objects. It specifies that the related Document\_file is referenced from the relating Document\_file.

EXAMPLE: A service manual may contain graphics for explanatory reasons. In this case the Document\_file objects that contain the graphics are referenced as related from the Document\_file object that contains the body of the service manual with relation\_type 'reference.'

##### **Base Class**

- PLM\_object (ABS)

##### **Attributes**

- relation\_type: string[1]:  
The relation\_type specifies the meaning of the relationship. The values 'addition,' 'decomposition,' and 'peer' are only allowed if neither the relating nor the related Document\_file are included in a Document\_representation.

##### **Compositions**

- description: String\_select[0..1]:  
The description specifies additional information about the Document\_file\_relationship.

##### **Associations**

- related : Document\_file[1]  
The related specifies the second of the two objects related by the Document\_file\_relationship.

#### 8.4.1.9 Class Document\_format\_property

A Document\_format\_property specifies characteristics of a Document\_file or of a Document\_representation that specifies the format of the object. At least one of the optional attributes shall be specified for each instance of this object.

##### **Base Class**

- PLM\_root\_object (ABS)

##### **Attributes**

- data\_format : String [0..1]  
The data\_format specifies the convention that was used to structure the information in the characterized object. Where applicable the following values shall be used:

'DXF'	The document contains data in Drawing Exchange File format
'IGES'	The document contains data in Initial Graphics Exchange Specification format
'ISO 10303-203'	The document contains data in ISO 10303-203 format.
'ISO 10303-214'	The document contains data in ISO 10303-214 format.

'TIFF CCITT GR4'	The document contains data in TIFF CCITT GR4 format
'VDAFS'	The document contains data in VDAFS format
'VOXEL'	The document contains data in VOXEL format.

- `character_code : String [0..1]`  
The `character_code` specifies the character code used in the characterized object. Where applicable the following values shall be used:

'binary'	The document contains data in binary format
'IEC 61286'	The coded character set used to encode the document data according to IEC 61286.
'ISO 646'	The coded character set used to encode the document data according to ISO 646.
'ISO 3098-1'	The coded character set used to encode the document data is according to ISO 3098-1.
'ISO 6937'	The coded character set used to encode the document data is according to ISO/IEC 6937.
'ISO 8859-1'	The coded character set used to encode the document data according to ISO 8859-1.
'ISO 10646'	The coded character set used to encode the document data according to ISO/IEC 10646.

### **Compositions**

- none

### **Associations**

- `size_format : Rectangular_size [0..1]`  
The `size_format` specifies the dimensions of a physical presentation of the object the `size_format` is provided for.

#### **8.4.1.10 Class Document\_location\_property**

A `Document_location_property` specifies where a `Document_file` or a `Document_representation` can be found in a digital or physical data storage system.

### **Base Class**

- `PLM_root_object (ABS)`

### **Attributes**

- `location_name : String [1]`  
The `location_name` specifies the location, where the object that refers to the `Document_location_property`, can be found. 'C:\mpbs\{ }programs' and '/usr/local/bin' are examples for a `location_name`.

### **Compositions**

- `external_file_id_and_location : External_file_id_and_location [0..*]`  
The `external_file_id_and_location` specifies the `Document_file` that is stored in this `Document_location_property`.

### **Associations**

- none

#### **8.4.1.11 Class Document\_representation (ABS)**

A Document\_representation is one of potentially more alternative representations of a Document\_version.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- id : String [1]  
The id specifies the identifier of the Document\_representation.

### **Compositions**

- document\_structure : Document\_structure [0..\*]  
The document\_structure specifies the document\_structure that relates the first of the two Document\_representation objects.
- description : String\_select [0..1]  
The description specifies additional information about the Document\_representation.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Document\_representation.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- content : Document\_content\_property [0..1]  
The content specifies characteristics of the content of the Document\_representation.
- size : Document\_size\_property [0..1]  
The size specifies the size of the represented document.
- representation\_format : Document\_format\_property [0..1]  
The representation\_format specifies the format of the document represented by Document\_representation.
- common\_location : Document\_location\_property [0..\*]  
The common\_location specifies the location of a Document\_representation, where all its constituents can be found.
- creation : Document\_creation\_property [0..1]  
The creation specifies further details of the creation of the Document\_representation.

#### **8.4.1.12 Class Document\_size\_property**

A Document\_size\_property specifies the size of a Document\_file or of a Document\_representation object. At least one of the optional attributes shall be specified for each instance of this object.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- page\_count : Value\_with\_unit (ABS) [0..1]  
The page\_count specifies the number of pages of the application object the Document\_size\_property is referred by. The page\_count shall only be used in cases where the Document\_size\_property is referred by a Hardcopy or a Physical\_representation.
- file\_size : Value\_with\_unit (ABS) [0..1]  
The file\_size specifies the Value\_with\_unit that represents the size of a digitally stored document. The file\_size shall only be applied in cases where the Document\_size\_property is referred by a Digital\_document or a Document\_file.

### **8.4.1.13 Class Document\_structure**

A Document\_structure is a relationship between two Document\_representation objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship. Where applicable the following values shall be used:

'addition'	The application object specifies that the related document provides supplementary or collateral information with regard to the information provided by the relating document.
'copy':	The application object defines a relationship where the related Document_representation is a copy of the relating Document_representation.
'decomposition'	The application object defines a relationship where the related Document_representation is one of potentially more sub documents of the relating Document_representation.
'derivation'	The application object defines a relationship where the related Document_representation is derived from the relating Document_representation.
'peer'	The application object specifies that the related document provides required information with regard to that provided by the relating document. The peer document is essential for a complete understanding.
'reference'	The application object defines a relationship where the related document is referenced from the relating.



'sequence'	The application object defines a logical sequence where the related Document_representation comes after the relating Document_representation.
'substitution'	The application object defines a relationship where the related Document_representation replaces the relating Document_representation.
'translation'	The Document_structure specifies that the related document is generated through a translation process from the relating document.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Document\_structure.

### **Associations**

- related : Document\_representation (ABS) [1]  
The related specifies the second of the two objects related by the Document\_structure.

#### **8.4.1.14 Class Document\_type\_property**

A Document\_type\_property specifies the kind of a Document\_file.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- document\_type\_name : String [1]  
The document\_type\_name specifies the word or the group of words that describe the kind of object the characteristics are provided for. Where applicable the following values shall be used:

'geometry'	The document represents a shape model.
'NC data'	The document represents numerical control data.
'FE data'	The document represents finite element data.
'sample data'	The document represents measured data.
'process plan'	The document represents process planning data.
'check plan'	The document represents quality control planning data.
'drawing'	The document represents a technical drawing.

### **Compositions**

- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Document\_type\_property.

### **Associations**

- used\_classification\_system : Classification\_system [0..1]  
The used\_classification\_system specifies the Classification\_system the document\_type\_name is defined in.

#### **8.4.1.15 Class Document\_version**

A Document\_version is a release of a Document.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- id : String [1]  
The id specifies the identifier of the Document\_version. The id shall be unique within the scope of the associated Document.

### **Compositions**

- document\_version\_relationship : Document\_version\_relationship [0..\*]  
The document\_version\_relationship specifies the document\_version\_relationship that relates the first of the two Document\_version objects.
- description : String\_select [0..1]  
The description specifies additional information about the Document\_version.
- document\_representation : Document\_representation (ABS) [0..\*]  
The document\_representation specifies the document\_representation that represents this version of the logical document.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Document\_version.

### **Associations**

- none

#### **8.4.1.16 Class Document\_version\_relationship**

A Document\_version\_relationship is a relationship between two Document\_version objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- `relation_type` : String [1]  
The `relation_type` specifies the meaning of the relationship. Where applicable the following values shall be used:

'derivation'	The application object defines a deriving relationship where the related <code>Document_version</code> is based on the relating <code>Document_version</code> , which is an earlier version of the same or of a different <code>Document</code> .
'hierarchy'	The application object defines a hierarchical relationship where the related <code>Document_version</code> is a sub version of the relating <code>Document_version</code> .
'sequence'	The application object defines a version sequence where the relating <code>Document_version</code> is the preceding version and the related <code>Document_version</code> is the following version.
'supplied document'	The application object defines a relationship between two <code>Document_version</code> objects representing the same object in different organizational contexts.

### **Compositions**

- `description` : String\_select [0..1]  
The `description` specifies additional information about the `Document_version_relationship`.

### **Associations**

- `related` : `Document_version` [1]  
The `related` specifies the second of the two objects related by the `Document_version_relationship`.

#### **8.4.1.17 Class External\_file\_id\_and\_location**

An `External_file_id_and_location` specifies the location of a file in an external storage system.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `external_id` : String [0..1]  
The `external_id` specifies the identifier of a document in an external storage system.

### **Compositions**

- none

### **Associations**

- none

#### **8.4.1.18 Class Hardcopy**

A Hardcopy is the actual stack of paper consisting of one or more sheets on which some product data is written, printed, or plotted.

##### **Base Class**

- Document\_file (ABS)

##### **Attributes**

- none

##### **Compositions**

- none

##### **Associations**

- none

#### **8.4.1.19 Class Named\_size**

A Named\_size is the definition of the size of a Document\_file or of a Document\_representation where the size is specified by a standardized identifier.

##### **Base Class**

- Rectangular\_size

##### **Attributes**

- size : String [1]  
The size specifies the size of the object. If the size differs from the dimensions specified by the inherited 'width' and 'height' attributes the size is overridden.

##### **Compositions**

- none

##### **Associations**

- referenced\_standard : Classification\_system [0..1]  
The referenced\_standard specifies a standard according to which the size is specified.

#### **8.4.1.20 Class Partial\_document\_assignment**

A Partial\_document\_assignment identifies a specific portion of the contents of a document. It assigns this portion to the product data for which it is relevant.

A Partial\_document\_assignment is a type of Document\_assignment.

##### **Base Class**

- Document\_assignment

### **Attributes**

- document\_portion: string[1]:  
The document\_portion specifies the word or group of words that convey the subject or sub contents of the Document.  
EXAMPLE: If the assigned document is a service manual, the document\_portion may specify that, in the given situation, only the information related to electric fuses is relevant among the overall contents of the document.

### **Compositions**

- none

### **Associations**

- none

#### **8.4.1.21 Class Physical\_document**

A Physical\_document is a piece of product data that is archived in a non-digital form.

### **Base Class**

- Physical\_representation (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- component : Hardcopy [0..\*]  
The component specifies the physical realization of the Physical\_document.

#### **8.4.1.22 Class Physical\_model**

A Physical\_model is a model of the shape of a part made from a certain material (e.g., clay or wood) at a certain scale. For a Physical\_model, neither a Document\_size\_property, a Document\_format\_property, nor a Document\_creation\_property shall be specified.

EXAMPLE: A Physical\_model may be used to get an impression of the future appearance of, e.g., a car or to use it, e.g., in a wind-tunnel.

A Physical\_model is a type of Physical\_representation.

### **Base Class:**

- Physical\_representation (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

#### **8.4.1.23 Class Physical\_representation (ABS)**

A Physical\_representation is a physically realizable representation of a Document\_version.

### **Base Class**

- Document\_representation (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

#### **8.4.1.24 Class Rectangular\_size**

A Rectangular\_size is the definition of the planar size of an object.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- density : Value\_with\_unit (ABS) [0..1]  
The density specifies the resolution of the object if it is a raster picture.
- height : Value\_with\_unit (ABS) [1]  
The height specifies the size of the object in vertical direction.
- width : Value\_with\_unit (ABS) [1]  
The width specifies the size of the object in horizontal direction.

## 8.4.2 Interfaces

### 8.4.2.1 Interface Assigned\_document\_select

This empty interface is realized by the following classes:

- Document\_version
- Document\_representation (ABS)
- Document\_file (ABS)
- Document

### 8.4.2.2 Interface Documented\_element\_select

#### **Compositions**

- document\_assignment : Document\_assignment [0..\*]

#### **Implemented By**

- Shape\_element\_relationship
- Process\_operation\_occurrence
- Work\_order
- Product\_identification
- Organization
- Contract
- Physical\_instance\_test\_result
- Item\_definition\_instance\_relationship
- Complex\_product
- Classification\_attribute
- Item
- Product\_class
- Class\_structure\_relationship
- Item\_definition\_relationship
- Specification\_category
- Change
- Specific\_item\_classification
- Material
- Specification
- Item\_version
- Activity\_element

- Project
- Classification\_system
- Security\_classification
- Process\_plan
- Activity\_method
- Approval
- Item\_instance
- Descriptive\_specification
- Property
- Product\_structure\_relationship
- Shape\_element
- Shape\_element\_relationship
- General\_classification
- Design\_discipline\_item\_definition
- Item\_instance\_relationship
- Physical\_instance
- Work\_request
- Certification
- Item\_shape
- Design\_constraint
- Physical\_assembly\_relationship
- Activity
- Retention\_period
- Class\_structure\_relationship
- Person



## 8.5 Package Shape\_definition\_and\_transformation

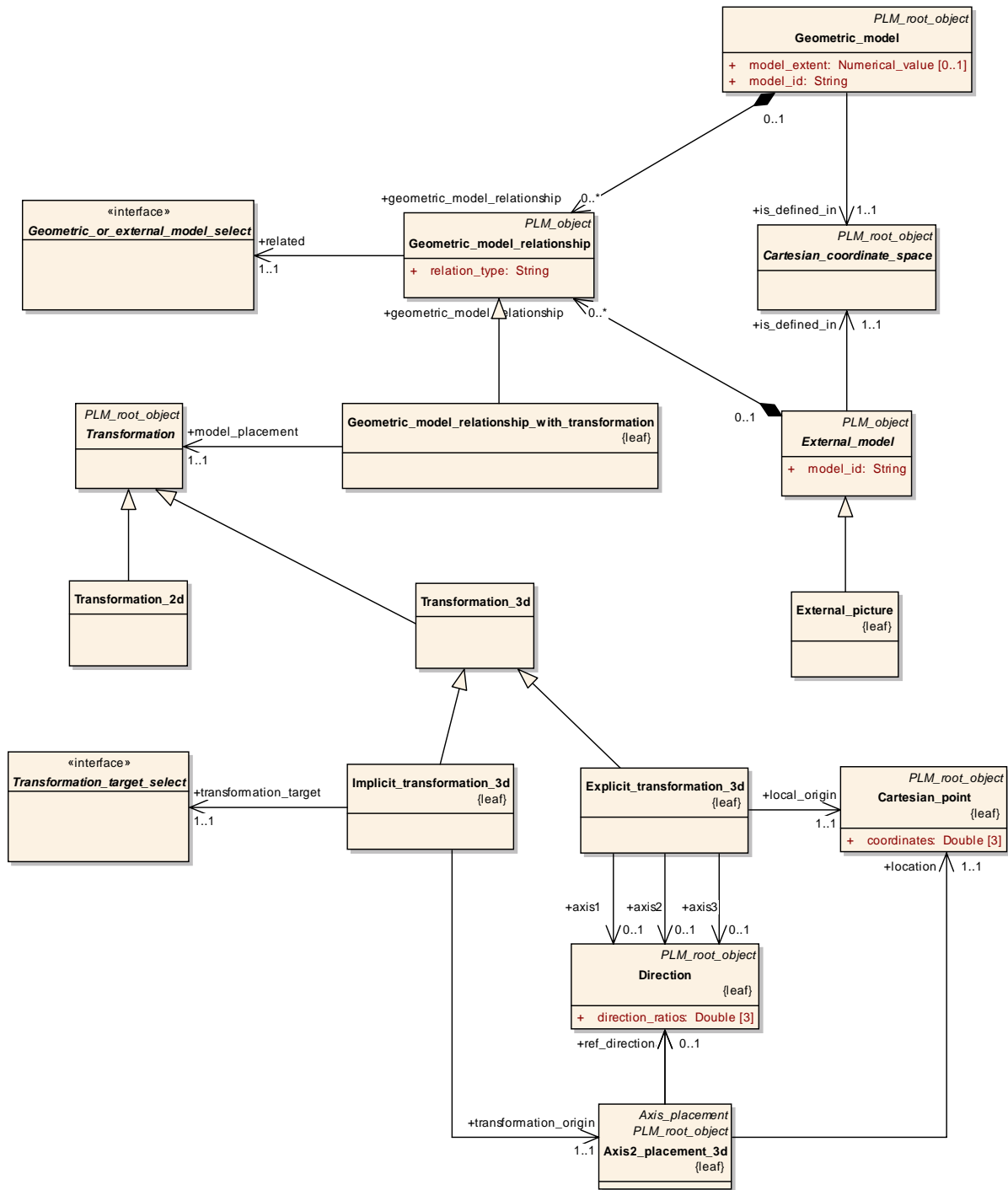


Figure 8.7 - Shape definition and transformation

## 8.5.1 Classes

### 8.5.1.1 Class Accuracy

An Accuracy is the information about the geometrical accuracy of the product data contained in a model.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- accuracy\_value : Double [1]  
The accuracy\_value specifies a numerical value defining the Accuracy.
- accuracy\_type : String [1]  
The accuracy\_type specifies the kind of accuracy that is applied. Where applicable the following values shall be used:

'angular accuracy'	A kind of accuracy that specifies the maximum value for the absolute angle between two curve tangents or two surface normals for which the creating system assumes curve tangents or surface normals being identical.
'curvature accuracy'	A kind of accuracy that specifies the value for the term under which a system can assume that the two radii of curvature R1 and R2 are identical. The curvature accuracy value is used to determine the accuracy range for curvature continuous curve or surface connections.
'distance accuracy'	A kind of accuracy that specifies the distance under which two points can be considered as having the same location. The distance accuracy value defined for a Geometric_model is valid for all geometric elements of the Geometric_model.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Accuracy.

#### **Associations**

- is\_defined\_for : Accuracy\_select [1..\*]  
The is\_defined\_for specifies the geometry to which the Accuracy is assigned.

### 8.5.1.2 Class Axis\_placement

An Axis\_placement is the representation of three orthogonal axes defined in a three-dimensional reference coordinate system, which intersect at a unique point. This point defines the origin of the Axis\_placement.

#### **Base Class:**

- PLM\_root\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- location: Cartesian\_point[0..1]  
The location specifies the geometric position of a reference point, such as the centre of a circle, of the item to be located.

#### **8.5.1.3 Class Axis1\_placement**

An Axis1\_placement represents the direction and location in three-dimensional space of a single axis. An axis1\_placement is defined in terms of a locating point (inherited from the Axis\_placement supertype) and an axis direction; this is either the direction of axis or defaults to (0.0,0.0,1.0). The actual direction for the axis placement is given by the derived attribute z; the normalized direction of the local Z axis.

### **Base Class**

- Axis\_placement (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- axis: Direction[0..1]:  
The axis specifies the direction of the local Z axis.

#### **8.5.1.4 Class Axis2\_placement\_2d**

An Axis2\_placement\_2D is the location and orientation in two-dimensional space of two mutually perpendicular axes. An axis2\_placement\_2d is defined in terms of a point, (inherited from the placement supertype), and an axis. It can be used to locate and orientate an object in two-dimensional space and to define a placement coordinate system. The entity includes a point that forms the origin of the placement coordinate system. A direction vector is required to complete the definition of the placement coordinate system. The ref\_direction defines the placement X axis direction; the placement Y axis direction is derived from this.

### **Base Class**

- Axis\_placement (ABS)

### **Attributes**

- none

## **Compositions**

- none

## **Associations:**

- ref\_direction: Direction[0..1]:  
The ref\_direction specifies the direction used to determine the direction of the local X axis. If ref\_direction is omitted, this direction is taken from the geometric coordinate system.

### **8.5.1.5 Class Axis2\_placement\_3d**

An Axis2\_placement\_3d is the location and orientation in three-dimensional space of two mutually perpendicular axes. An axis2\_placement\_3d is defined in terms of a point (inherited from the placement supertype) and two (ideally orthogonal) axes. It can be used to locate and orientate a non axis-symmetric object in space and to define a placement coordinate system. The entity includes a point that forms the origin of the placement coordinate system. Two direction vectors are required to complete the definition of the placement coordinate system. The axis is the placement Z axis direction and the ref\_direction is an approximation to the placement X axis direction.

NOTE: Let z be the placement Z axis direction and a be the approximate placement X axis direction. There are two methods, mathematically identical but numerically different, for calculating the placement X and Y axis directions.

The vector a is projected onto the plane defined by the origin point P and the vector z to give the placement X axis direction as  $x = (a - (a.z)z)$ . The placement Y axis direction is then given by  $y = (z \times x)$ .

The placement Y axis direction is calculated as  $y = (z \times a)$  and then the placement X axis direction is given by  $x = (y \times z)$ .

The first method is likely to be the more numerically stable of the two, and is used here.

A placement coordinate system referenced by the parametric equations is derived from the axis2\_placement\_3d data for conic curves and elementary surfaces.

## **Base Class**

- Axis\_placement (ABS)

## **Attributes**

- none

## **Compositions**

- none

## **Associations**

- ref\_direction : Direction [0..1]  
The ref\_direction can be used to determine the direction of the local X axis.
- axis : Direction [0..1]  
The axis defines the exact direction of the local Z axis.

### 8.5.1.6 Class Cartesian\_coordinate\_space (ABS)

Cartesian\_coordinate\_space is a coordinate space in which geometric and annotation elements may be defined. It is either two-dimensional or three-dimensional. An origin for coordinate values is implicitly defined. The units applicable to the coordinate values of elements defined in the Cartesian\_coordinate\_space are specified.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- unit\_of\_values : Unit [0..\*]  
The unit\_of\_values specifies the various units in which any values are expressed. The same length unit is applied to each coordinate direction. Only one unit of a kind shall be specified. In the case where geometric elements are defined in the Cartesian\_coordinate\_space, there shall be at least two units specified, the length unit, and the plane angle unit.

### 8.5.1.7 Class Cartesian\_coordinate\_space\_2d

A Cartesian\_coordinate\_space\_2d is a two-dimensional coordinate space. Any two-dimensional geometric and annotation element shall be defined in a Cartesian\_coordinate\_space\_2d.

#### **Base Class**

- Cartesian\_coordinate\_space (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- none

### 8.5.1.8 Class Cartesian\_coordinate\_space\_3d

A Cartesian\_coordinate\_space\_3d is a three-dimensional coordinate space. Any three-dimensional geometric data shall be defined in a Cartesian\_coordinate\_space\_3d.

#### **Base Class**

- Cartesian\_coordinate\_space (ABS)

**Attributes**

- none

**Compositions**

- none

**Associations**

- none

**8.5.1.9 Class Cartesian\_point**

A Cartesian\_point is a point that is defined by its coordinates in a rectangular Cartesian coordinate system.

**Base Class**

- Point

**Attributes**

- coordinates : Double [3]  
The coordinates specify the 3 coordinates of the point.

**Compositions**

- none

**Associations**

- none

**8.5.1.10 Class Detailed\_element**

A Detailed\_element is a single element of a Geometric\_model or of a Styled\_model.

Each Detailed\_element is a Detailed\_model\_element, a Styled\_element, or a Draughting\_callout.

**Base Class:**

- PLM\_object (ABS)

**Attributes**

- none

**Compositions:**

- name: String\_select[0..1]  
The name specifies the word or group of words by which the Detailed\_element is referred to.

**Associations**

- none

#### **8.5.1.11 Class Detailed\_geometric\_model\_element**

A Detailed\_geometric\_model\_element is the identification of a geometric element.

NOTE: The Detailed\_geometric\_model\_element may be used to define the elements of a Geometric\_model.

A Detailed\_geometric\_model\_element is a type of Detailed\_model\_element.

##### **Base Class**

- Detailed\_model\_element (ABS)

##### **Attributes**

- none

##### **Compositions**

- none

##### **Associations**

- none

#### **8.5.1.12 Class Detailed\_model\_element**

A Detailed\_model\_element is a single element of a model. A Detailed\_model\_element is a type of Detailed\_element. Each Detailed\_model\_element is a Template\_instance or a Detailed\_geometric\_model\_element.

##### **Base Class**

- Detailed\_element (ABS)

##### **Attributes**

- none

##### **Compositions**

- none

##### **Associations**

- none

#### **8.5.1.13 Class Direction**

A Direction in a 3-dimensional space is expressed as a vector.

##### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- direction\_ratios : Double [3]  
The direction\_ratios specify the 3 ratios of the direction vector components.

### **Compositions**

- none

### **Associations**

- none

#### **8.5.1.14 Class Explicit\_transformation\_2d**

An Explicit\_transformation\_2d is a geometric transformation in two-dimensional space where the translation and rotation is explicitly defined.

An Explicit\_transformation\_2d is a type of Transformation\_2d.

### **Base Class**

- Transformation\_2d

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- axis1 : Direction[0..1]:  
The axis1 specifies the X axis direction.
- axis2 : Direction[0..1]:  
The axis2 specifies the Y axis direction.
- local\_origin : Cartesian\_point[1]:  
The local\_origin specifies the required translation. The actual translation included in the transformation is from the geometric origin to the local origin.

#### **8.5.1.15 Class Explicit\_transformation\_3d**

A geometric relationship between external models can be defined explicitly by using an Explicit\_transformation\_3d that has a local origin and a rotation matrix.

### **Base Class**

- Transformation\_3d

### **Attributes**

- none



### **Compositions**

- none

### **Associations**

- axis3 : Direction [0..1]  
The axis3 is the Z axis direction of the transformation target.
- axis2 : Direction [0..1]  
The axis2 is the Y axis direction of the transformation target.
- axis1 : Direction [0..1]  
The axis1 is the X axis direction of the transformation target.
- local\_origin : Cartesian\_point [1]  
The local\_origin is the required translation specified as a cartesian point. The actual translation included in the transformation is from the geometric origin to the local origin.

#### **8.5.1.16 Class External\_geometric\_model**

An External\_geometric\_model is the identification of a model that contains geometry in a 3D context only.

### **Base Class**

- External\_model (ABS)

### **Attributes**

- model\_extent : Numerical\_value [0..1]  
The model\_extent specifies the radius of a sphere that contains all elements of the model and whose centre is at the origin of the Cartesian\_coordinate\_space of the External\_geometric\_model. The model\_extent is specified using a length unit.

### **Compositions**

- none

### **Associations**

- none

#### **8.5.1.17 Class External\_geometric\_model\_with\_parameters**

An External\_geometric\_model\_with\_parameters is the identification of a model that contains geometry in a 3D context, and where some characteristics of the model may be controlled or changed in a predefined way in each occurrence of the model.

EXAMPLE: A geometric model may have predefined aspect ratios of its overall dimensions, but a parameterized volume. In each occurrence of the External\_geometric\_model\_with\_parameters, the volume may be specified by the user and the dimensions of the model are set automatically while the predefined aspect ratio requirements are met.

The usage of the controlled characteristics shall be the subject of an agreement of common understanding by partners sharing this information.

An External\_geometric\_model\_with\_parameters is a type of External\_geometric\_model.

### **Base Class**

- External\_geometric\_model

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- parameter\_value: General\_parameter[1]:  
The parameter\_value specifies the General\_parameter object or group of General\_parameter objects that may be varied in each occurrence.

### **8.5.1.18 Class External\_model (ABS)**

An External\_model is the identification of a model that is described in a Digital\_file and by the Cartesian\_coordinate\_space that is needed to further process the externally described information.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- model\_id : String [1]  
The model\_id specifies the identifier of the External\_model.

### **Compositions**

- geometric\_model\_relationship : Geometric\_model\_relationship [0..\*]  
The geometric\_model\_relationship specifies the geometric\_model\_relationship that relates the first of the two External\_model objects.
- description : String\_select [0..1]  
The description specifies additional information about the External\_model.

### **Associations**

- is\_defined\_in : Cartesian\_coordinate\_space (ABS) [1]  
The is\_defined\_in specifies the Cartesian\_coordinate\_space that defines the context for the externally described geometry. If the External\_model is an External\_picture, the context shall be a Cartesian\_coordinate\_space\_2d.

### **8.5.1.19 Class External\_picture**

An External\_picture is the identification of a model that is described by a two dimensional image.

### **Base Class**

- External\_model (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

## **8.5.1.20 Class Geometric\_model**

A Geometric\_model is a representation of geometry. A Geometric\_model that does not reference any Detailed\_geometric\_model\_element objects through one of the subtypes directly shall either reference at least one Template\_instance as 'additional\_element' or shall reference Axis\_placement objects exclusively.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- model\_id : String [1]  
The model\_id specifies the identifier of the Geometric\_model.
- model\_extent : Numerical\_value [0..1]  
The model\_extent specifies the radius of a sphere that contains all elements of the model and whose centre is at the origin of the Cartesian\_coordinate\_space of the Geometric\_model. The model\_extent is specified using a length unit.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Geometric\_model.
- geometric\_model\_relationship : Geometric\_model\_relationship [0..\*]  
The geometric\_model\_relationship specifies the geometric\_model\_relationship that relates the first of the two Geometric\_model objects.

### **Associations**

- is\_defined\_in : Cartesian\_coordinate\_space (ABS) [1]  
The is\_defined\_in specifies the Cartesian\_coordinate\_space in which the Geometric\_model is defined. The specified Cartesian\_coordinate\_space serves also as the reference coordinate space for the transformation of Template\_instance objects used as additional elements in the Geometric\_model.

## **8.5.1.21 Class Geometric\_model\_relationship**

A Geometric\_model\_relationship is a relationship between two models. The models may be either of type Geometric\_model or of type External\_model.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- `relation_type` : String [1]  
The `relation_type` specifies the meaning of the relationship.

### **Compositions**

- `description` : String\_select [0..1]  
The `description` specifies additional information about the `Geometric_model_relationship`.

### **Associations**

- `related` : Geometric\_or\_external\_model\_select [1]  
The `related` specifies the second of the two model objects related by the `Geometric_model_relationship`.

## **8.5.1.22 Class Geometric\_model\_relationship\_with\_transformation**

A `Geometric_model_relationship_with_transformation` is a relationship between two model objects with the additional information about a geometric Transformation. This Transformation defines the location and orientation of the related model relative to the relating model.

### **Base Class**

- `Geometric_model_relationship`

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- `model_placement` : Transformation (ABS) [1]  
The `model_placement` specifies the geometric Transformation that places and orients the related model relative to the relating model.

## **8.5.1.23 Class Geometric\_model\_version**

A `Geometric_model_version` is a particular version of a `Geometric_model`. A `Geometric_model_version` is a type of `Geometric_model`.

### **Base Class**

- `Geometric_model`

### **Attributes**

- `version_id`: string[1]:  
The `version_id` specifies the identification of a particular version of a `Geometric_model`.

### **Compositions**

- none

### **Association**

- none

#### **8.5.1.24 Class Geometrical\_relationship**

A Geometrical\_relationship is the relationship between two Design\_discipline\_item\_definition objects specifying two parts that are geometrically related.

### **Base Class**

- Item\_definition\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Geometrical\_relationship.

### **Associations**

- definition\_placement : Transformation\_select [1]  
The definition\_placement specifies the Geometric\_model\_relationship\_with\_transformation or the Template\_instance that has the Transformation to be applied to the relating Design\_discipline\_item\_definition in order to define the location and the orientation of the related Design\_discipline\_item\_definition. Translation, rotation, and mirroring, i.e., inversion, is included; scaling is not included. In the case of a Template\_instance, the scale factor shall be omitted or set to 1.0.

#### **8.5.1.25 Class Implicit\_transformation\_2d**

An Implicit\_transformation\_2d is a transformation that is defined by two instances of axis2\_placement\_2d in which one instance of axis2\_placement\_2d is the result of applying the transformation function to the other. The transformation function is not explicitly provided, but it is derived from the relationship between the instances of axis2\_placement\_2d. The transformation between the two instances of axis2\_placement\_2d applies to every element in the objects related by the Implicit\_transformation\_2d.

### **Base Class**

- Transformation\_2d

### **Attributes**

- none

### **Compositions**

- transformation\_target: axis2\_placement\_2d[1]  
The transformation\_target specifies the axis2\_placement\_2d defining the target of the transformation.
- transformation\_origin: axis2\_placement\_2d[1]  
The transformation\_origin specifies the axis2\_placement\_2d defining the origin of the transformation.

### **Associations**

- none

### **8.5.1.26 Class Implicit\_transformation\_3d**

A geometric relationship between external models can be defined implicitly by using an Implicit\_transformation\_3d that has two reference points to specify origin and target of the transformation.

### **Base Class**

- Transformation\_3d

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- transformation\_origin : Axis2\_placement\_3d [1]  
The transformation\_origin specifies the origin of the transformation.
- transformation\_target : Transformation\_target\_select [1]  
The transformation\_target specifies the target of the transformation.

### **8.5.1.27 Class Item\_shape**

An Item\_shape is the definition of the shape of a Design\_discipline\_item\_definition, an Item\_instance or of a Physical\_instance.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- none

### **Compositions**

- shape\_element : Shape\_element [0..\*]  
The shape\_element specifies the shape\_element that is part of this Item\_shape.
- shape\_description\_association : Shape\_description\_association [0..\*]  
The shape\_description\_association specifies the shape\_description\_association that is associated with this Item\_shape.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Item\_shape.
- description : String\_select [0..1]  
The description specifies additional information about the Item\_shape.

- `simple_property_value` : `Simple_property_value` (ABS) [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- `described_object` : `Item_information_select` [1]  
The `described_object` specifies the object whose shape the `Item_shape` defines.

### **8.5.1.28 Class Material**

A Material is the substance out of which an item is or can be made.

#### **Base Class**

- `PLM_root_object` (ABS)

#### **Attributes**

- `material_name` : `String` [1]  
The `material_name` specifies the word or group of words by which the Material is referred to.

#### **Compositions**

- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this Material.
- `material_property_association` : `Material_property_association` [0..\*]  
The `material_property_association` specifies the `material_property_association` in which a property value is assigned to this Material.

#### **Associations**

- `described_element` : `Item_property_select` [1..\*]  
The `described_element` specifies the objects the material information is provided for.

### **8.5.1.29 Class Point**

A Point is a location in a Cartesian coordinate space. A Point is a type of `Detailed_geometric_model_element`.

#### **Base Class**

- `Detailed_geometric_model_element` (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- none

### 8.5.1.30 Class Shape\_description\_association

A Shape\_description\_association is a mechanism to associate the definition of a shape or of a portion of a shape with a geometric representation.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- role : String [1]  
The role specifies the function performed by the referenced model. Where applicable the following values shall be used:

'detailed representation'	The geometry in the referenced model provides a detailed representation of the shape.
'idealized representation'	The geometry in the referenced model provides a simplified representation of the shape (e.g., for analysis purposes).

#### **Compositions**

- none

#### **Associations**

- defining\_geometry : Shape\_definition\_select [1]  
The defining\_geometry specifies the Geometric\_model or the External\_model that contains the shape information.

### 8.5.1.31 Class Shape\_element

A Shape\_element is a portion of shape that has to be identified explicitly to be associated with other information.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- element\_name : String [0..1]  
The element\_name specifies the word or group of words by which the Shape\_element is referred to.

#### **Compositions**

- change : Change [0..\*]  
The change specifies the change for which this object references a modified object and the corresponding original object.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Shape\_element.
- shape\_description\_association : Shape\_description\_association [0..\*]  
The shape\_description\_association specifies the shape\_description\_association that is associated with this Shape\_element.



- `shape_element_relationship` : `Shape_element_relationship` [0..\*]  
The `shape_element_relationship` specifies the `shape_element_relationship` that relates the first of the two `Shape_element` objects.
- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Shape_element`.
- `simple_property_value` : `Simple_property_value (ABS)` [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- none

### **8.5.1.32 Class Shape\_element\_relationship**

A `Shape_element_relationship` is a relationship between two `Shape_element` objects.

#### **Base Class**

- `PLM_object (ABS)`

#### **Attributes**

- `relation_type` : `String` [1]  
The `relation_type` specifies the meaning of the relationship.

#### **Compositions**

- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Shape_element_relationship`.
- `shape_description_association` : `Shape_description_association` [0..\*]  
The `shape_description_association` specifies the `shape_description_association` that is associated with this `Shape_element_relationship`.
- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Shape_element_relationship`.
- `simple_property_value` : `Simple_property_value (ABS)` [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- `related` : `Shape_element` [1]  
The `related` specifies the second of the two `Shape_element` objects related by a `Shape_element_relationship`.

### **8.5.1.33 Class Template\_instance**

A `Template_instance` is a replica of a `Geometric_model`, of a `Styled_model`, or of an `External_model`.

The location of the replica and any additional geometrical transformation to be applied to the replica, i.e., uniform scaling, rotation or mirroring, are specified by the attributes ‘`transformation`,’ ‘`origin`,’ and ‘`target`.’

NOTE 1: In the case where the units of the Cartesian coordinate space of the definition are different from the units to be applied to the `Template_instance`, unit conversion is required.

NOTE 2: In case of length unit conversion, this conversion applies in addition to the scale attribute.

EXAMPLE: In a technical drawing of a mechanical part with several identical drilling holes, the hole geometry (circle) together with its annotation elements (diameter dimension and centrelines) is defined once. This particular definition is instantiated several times at different locations by corresponding `Template_instance` objects.

A `Template_instance` is a type of `Detailed_model_element`.

### **Base Class**

- `Detailed_model_element` (ABS)

### **Attributes**

- `id`: `string[1]`:  
The `id` specifies the identifier of the `Template_instance`.
- `scale`: `double[0..1]`:  
The `scale` specifies the scaling factor for all Cartesian coordinate directions. The scaling factor shall be positive. If the scaling factor is omitted, it shall be 1.0.

### **Compositions**

- `target`: `Explicit_transformation[1]`  
The `target` specifies the geometrical transformation applied to the instance. All transformations that can be expressed by an orthonormal  $2 \times 2$  (for 2D) or  $3 \times 3$  (for 3D) matrix can be applied.  
EXAMPLE: Examples for suitable operations are rotation and mirroring.

### **Associations**

- `template_definition` : `Template_definition_select[1]`  
The `template_definition` specifies the object to be replicated.
- `origin` : `Axis_placement[1]`  
The `origin` specifies the `Axis_placement` that defines the origin of the replicated object.

#### **8.5.1.34 Class Transformation (ABS)**

A Transformation is a geometric transformation composed of translation and rotation. Scaling is not included.

### **Base Class**

- `PLM_root_object` (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

### **8.5.1.35 Class Transformation\_2d**

A Transformation\_2d is the definition of a geometric transformation in 2D space. A Transformation\_2d is a type of Transformation.

#### **Base Class**

- Transformation

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- none

### **8.5.1.36 Class Transformation\_3d**

A Transformation\_3d is the definition of a geometric transformation in 3D space.

#### **Base Class**

- Transformation (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- none

## **8.5.2 Interfaces**

### **8.5.2.1 Interface Accuracy\_select**

This empty interface is realized by the following classes:

- Geometric\_model
- External\_geometric\_model

### **8.5.2.2 Interface Geometric\_or\_external\_model\_select**

This empty interface is realized by the following classes:

- Geometric\_model
- External\_model (ABS)

#### **8.5.2.3 Interface Shape\_definition\_select**

This empty interface is realized by the following classes:

- Geometric\_model
- External\_geometric\_model

#### **8.5.2.4 Interface Shape\_information\_select**

##### ***Compositions***

- shape\_description\_association :  
    Shape\_description\_association [0..\*]

##### ***Implemented By***

- Shape\_element\_relationship
- Shape\_element

#### **8.5.2.5 Interface Transformation\_select**

This empty interface is realized by the following class:

- Template\_instance
- Geometric\_model\_relationship\_with\_transformation

#### **8.5.2.6 Interface Transformation\_target\_select**

This empty interface is realized by the following classes:

- Explicit\_transformation\_3d
- Axis2\_placement\_3d

## 8.6 Package Classification

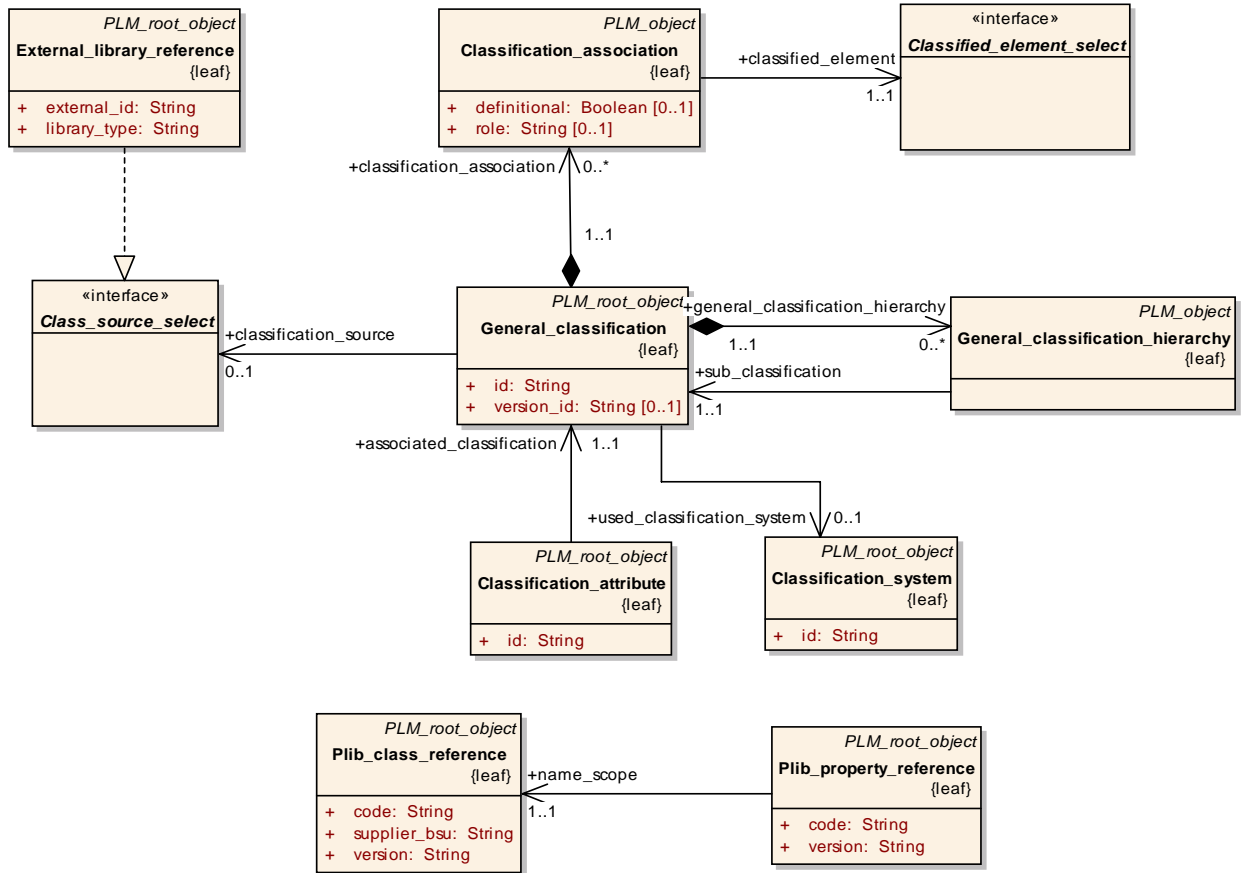


Figure 8.8 - Classification - General

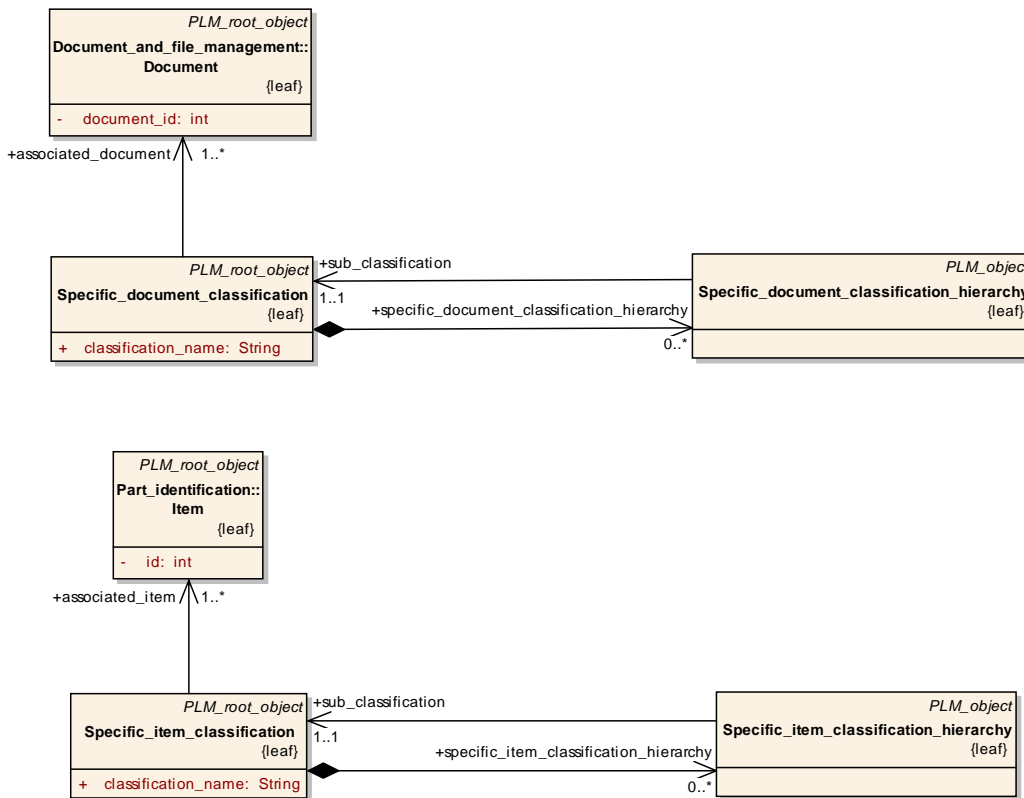


Figure 8.9 - Classification - Item and document

## 8.6.1 Classes

### 8.6.1.1 Class Classification\_association

A Classification\_association associates a General\_classification with an object.

#### Base Class

- PLM\_object (ABS)

#### Attributes

- role : String [0..1]  
The role specifies the relationship between the General\_classification and the associated object. Where applicable the following values shall be used:

‘electromagnetic compatibility’      The associated object is the classification that categorizes the classified element in respect of its ability to comply with requirements concerning electromagnetic interference.

‘environmental conditions’      The associated object is the classification that categorizes the classified element with respect to its ability to comply with environmental impact requirements.

- **definitional** : Boolean [0..1]  
The definitional specifies whether a General\_classification serves as definition. A value of 'true' indicates that the General\_classification is definitional. The 'associated\_classification' does not take precedence over the descriptions of the 'classified\_element' made using Property\_value or Geometric\_model objects.

### **Compositions**

- none

### **Associations**

- **classified\_element** : Classified\_element\_select [1..\*]  
The classified\_element specifies the objects that are classified.

#### **8.6.1.2 Class Classification\_attribute**

A Classification\_attribute is a characteristic used to classify an object associated with the corresponding General\_classification. The definition attribute of each 'allowed\_value' shall refer to the property identified within 'attribute\_definition.'

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- **id** : String [1]  
The id specifies the identifier of the Classification\_attribute that shall be unique within the scope of the associated General\_classification.

### **Compositions**

- **alias\_identification** : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Classification\_attribute.
- **document\_assignment** : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Classification\_attribute.
- **description** : String\_select [0..1]  
The description specifies additional information about the Classification\_attribute.
- **name** : String\_select [0..1]  
The name specifies the word or group of words by which the Classification\_attribute is referred to.

### **Associations**

- **attribute\_definition** : Property (ABS) [1]  
The attribute\_definition specifies the Property that characterizes the allowed values.
- **allowed\_value** : Property\_value\_representation [0..\*]  
The allowed\_value specifies the set of Property\_value\_representation objects that represent characteristic values of the Classification\_attribute.
- **associated\_classification** : General\_classification [1]  
The associated\_classification specifies the General\_classification the Classification\_attribute is a characteristic of.

### 8.6.1.3 Class Classification\_system

A Classification\_system is the scheme used to define the categorization of an item.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [1]  
The id specifies the identifier of the Classification\_system.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Classification\_system.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Classification\_system.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Classification\_system.

#### **Associations**

- none

### 8.6.1.4 Class External\_library\_reference

An External\_library\_reference is a mechanism to refer to an entry in an external library other than ISO 13584.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- external\_id : String [1]  
The external\_id specifies the unique identifier of the referenced entry in the external library.
- library\_type : String [1]  
The library\_type specifies the type of library that is used.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the External\_library\_reference.

#### **Associations**

- none



### 8.6.1.5 Class General\_classification

A General\_classification is a classification of an object that characterizes all objects of the same kind; such a classification is independent from the application of the classified object.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [1]  
The id specifies the identifier of the General\_classification.
- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of the General\_classification.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the General\_classification.
- general\_classification\_hierarchy : General\_classification\_hierarchy [0..\*]  
The General\_classification\_hierarchy specifies the General\_classification\_hierarchy for which this General\_classification is the higher level, and that includes the sub class.
- classification\_association : Classification\_association [0..\*]  
The Classification\_association specifies the Classification\_association for which this General\_classification object provides classification information.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this General\_classification.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this General\_classification.

#### **Associations**

- classification\_source : Class\_source\_select [0..1]  
The classification\_source specifies the External\_library\_reference or the Plib\_class\_reference that contains the specification of the General\_classification.
- used\_classification\_system : Classification\_system [0..1]  
The used\_classification\_system specifies the Classification\_system that contains the information about the definition of the classification and how to interpret the name of the General\_classification.

### 8.6.1.6 Class General\_classification\_hierarchy

A General\_classification\_hierarchy defines a hierarchical relationship between two instances of General\_classification.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- sub\_classification : General\_classification [1]  
The sub\_classification specifies the lower level of General\_classification in a General\_classification\_hierarchy that is included in the super class.

#### **8.6.1.7 Class Plib\_class\_reference**

A Plib\_class\_reference designates a class in a library compliant to ISO 13584 (Parts Library).

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- code: string[1]  
The code specifies the class in the PLIB library. The format of this code is defined in ISO 13584-42.
- supplier\_bsu: string[1]  
The supplier\_bsu (basic semantic unit) specifies the supplier of the class in a PLIB library, in which the class is defined. The format of this specification is defined in ISO 13584-26.
- version: string[1]:  
The version specifies the identification of a particular version of a class in a PLIB library. The format of this version is defined in ISO 13584-42.

### **Compositions**

- none

### **Associations**

- none

#### **8.6.1.8 Class Plib\_property\_reference**

A Plib\_property\_reference designates a property in a library compliant to ISO 13584.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- code: string [1]  
The code specifies the property in the PLIB library. The format of this code is defined in ISO 13584-42.
- version: string [1]  
The version specifies the identification of a particular version of a property in a PLIB library. The format of this version is defined in ISO 13584-42.

### **Compositions**

- none

### **Associations**

- name\_scope: Plib\_class\_reference[1]  
The name\_scope specifies the Plib\_class\_reference in which the property is visible.

#### **8.6.1.9 Class Specific\_document\_classification**

A Specific\_document\_classification is a classification of a Document with respect to specific criteria. The specific criteria are covered in the 'classification\_name' attribute.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- classification\_name : String [1]  
The classification\_name provides classification information. Where applicable the following values shall be used:

'catalogue'	The assigned document is the catalogue in which the associated object is listed.
'manual'	The assigned document is the handbook that is supplied for the associated object.
'specification'	The assigned document specifies the considerations that lead to the design finally chosen for the associated object.

### **Compositions**

- specific\_document\_classification\_hierarchy : Specific\_document\_classification\_hierarchy [0..\*]  
The Specific\_document\_classification\_hierarchy specifies the Specific\_document\_classification\_hierarchy for which this Specific\_document\_classification is the higher level, and that is included in the sub class.
- description : String\_select [0..1]  
The description specifies additional information about the Specific\_document\_classification.

### **Associations**

- associated\_document : Document [1..\*]  
The associated\_document specifies the Document with which a particular Specific\_document\_classification is associated.

#### **8.6.1.10 Class Specific\_document\_classification\_hierarchy**

A Specific\_document\_classification\_hierarchy is used to build up hierarchical structures of Specific\_document\_classification\_hierarchy objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- sub\_classification : Specific\_document\_classification [1]  
The sub\_classification specifies the lower level of Specific\_document\_classification in Specific\_document\_classification\_hierarchy that is included in the super class.

### **8.6.1.11 Class Specific\_item\_classification**

A Specific\_item\_classification is a classification of an Item with respect to specific criteria. The specific criteria are covered in the 'classification\_name' attribute.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- classification\_name : String [1]  
The classification\_name provides high level classification information. Where applicable the following values shall be used:

'application control'	This type of classification is used to indicate that an Item shall be considered under certification aspects; these aspects may be specified further by the 'description' attribute.
'assembly'	This type of classification shall be used for any Item that has an Assembly_definition provided for at least one of its versions (i.e., it is decomposed further).
'collection'	This type of classification shall be used for any Item that has a Collection_definition provided for at least one of its versions.
'completely knocked down'	This type of classification is used to indicate that an Item is used in a production site that has assembling facilities only.
'detail'	This type of classification shall be used for any Item that has no Assembly_definition provided for any of its versions (i.e., it is not further decomposed).
'in process'	This type of classification is used to indicate that the Item identifies an intermediate object in a manufacturing process.
'part'	The Item plays the role of a part.
'prototype'	This type of classification is used to indicate that the Item identifies a prototype and is not intended for serial production.
'raw material'	The Item plays the role of raw material.

'regulated'	This type of classification is used to indicate that for an Item certain regulations have to be considered.
'safety'	This type of classification is used to indicate that an Item is relevant for safety purposes.
'service'	This type of classification is used to indicate that an Item is relevant for service purposes.
'tool'	The Item plays the role of a tool.

### **Compositions**

- `specific_item_classification_hierarchy` : `Specific_item_classification_hierarchy` [0..\*]  
The `Specific_item_classification_hierarchy` specifies the `Specific_item_classification_hierarchy` for which this `Specific_item_classification` is the higher level, and that includes the sub class.
- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Specific_item_classification`.
- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Specific_item_classification`.

### **Associations**

- `associated_item` : `Item` [1..\*]  
The `associated_item` specifies the `Item` with which a particular `Specific_item_classification` is associated.

#### **8.6.1.12 Class `Specific_item_classification_hierarchy`**

A `Specific_item_classification_hierarchy` is used to build up hierarchical structures of `Specific_item_classification`.

#### **Base Class**

- `PLM_object` (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- `sub_classification` : `Specific_item_classification` [1]  
The `sub_classification` specifies the lower level of `Specific_item_classification` in a `Specific_item_classification_hierarchy` that is included in the super class.

## **8.6.2 Interfaces**

### **8.6.2.1 Interface `Class_source_select`**

This empty interface is realized by the following class:

- External\_library\_reference
- Plib\_class\_reference

### **8.6.2.2 Interface Classified\_element\_select**

This empty interface is realized by the following classes:

- Design\_constraint
- Item
- Approval\_status
- Security\_level
- Product\_class
- Document
- Contract
- Work\_request
- Work\_order
- Project
- Activity\_method
- Activity
- Effectivity
- Specification
- Specification\_category
- Product\_identification
- Product\_class
- Design\_constraint
- Complex\_product (ABS)
- Document\_version
- Document\_representation (ABS)
- Document\_file (ABS)
- Document
- Item\_version
- Design\_discipline\_item\_definition
- Item\_instance (ABS)
- Process\_plan
- Process\_operation\_occurrence
- Process\_operation\_definition

- Property\_value\_association (ABS)
- Property (ABS)
- Simple\_property\_association
- Shape\_element
- Material

## 8.7 Package Properties

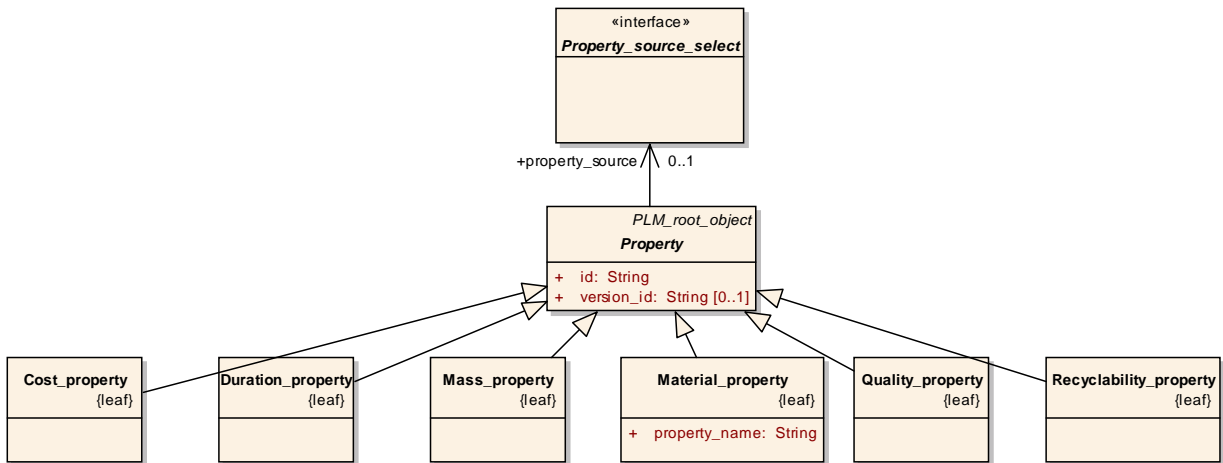


Figure 8.10 - Properties

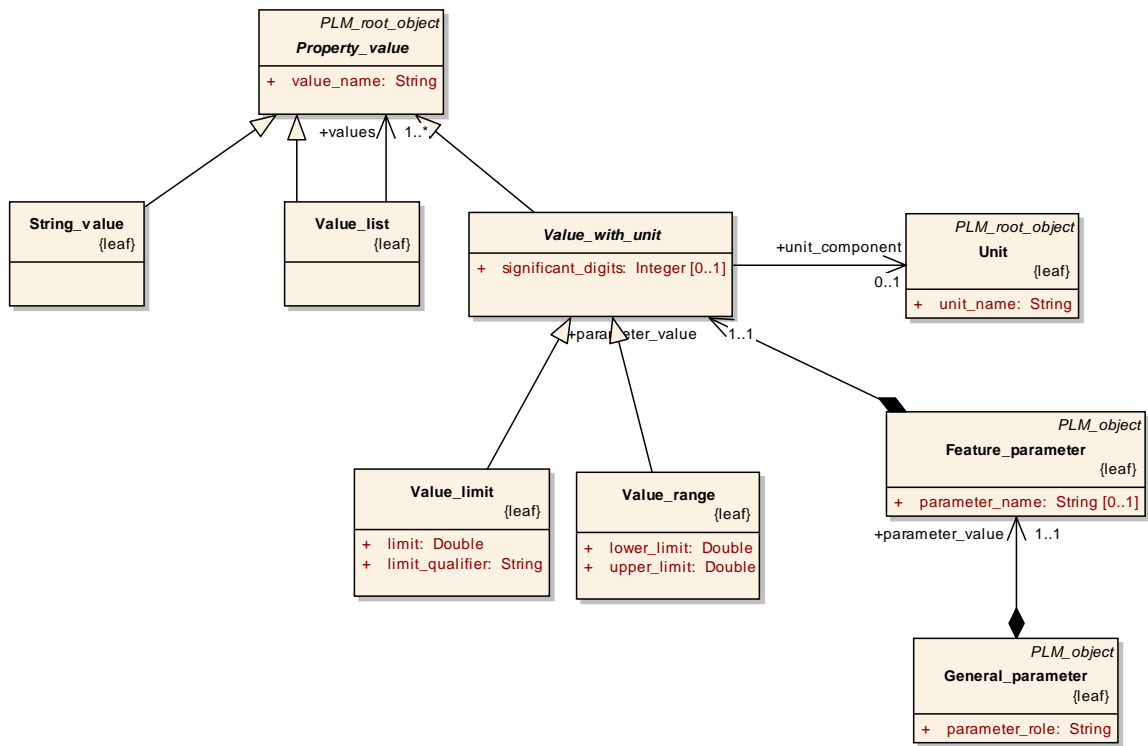


Figure 8.11 - Property value

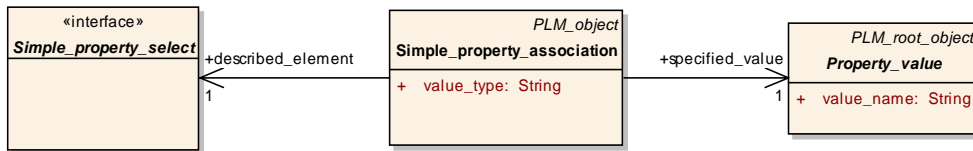


Figure 8.12 - Simple property

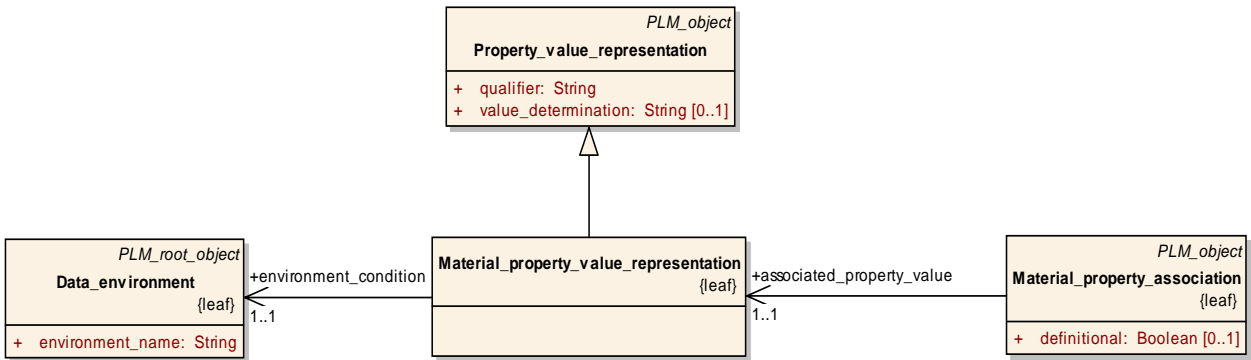


Figure 8.13 - Material property



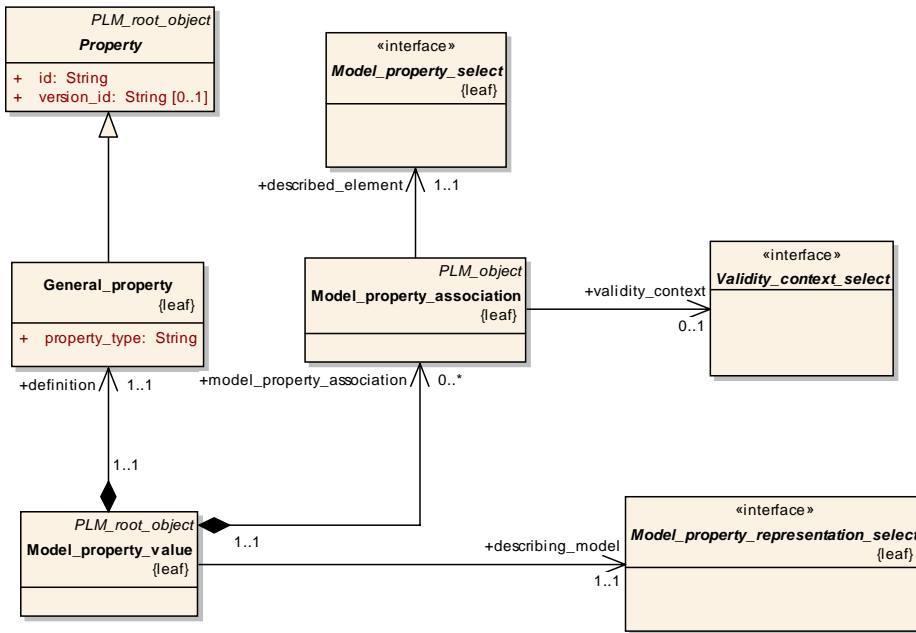


Figure 8.14 - Model properties

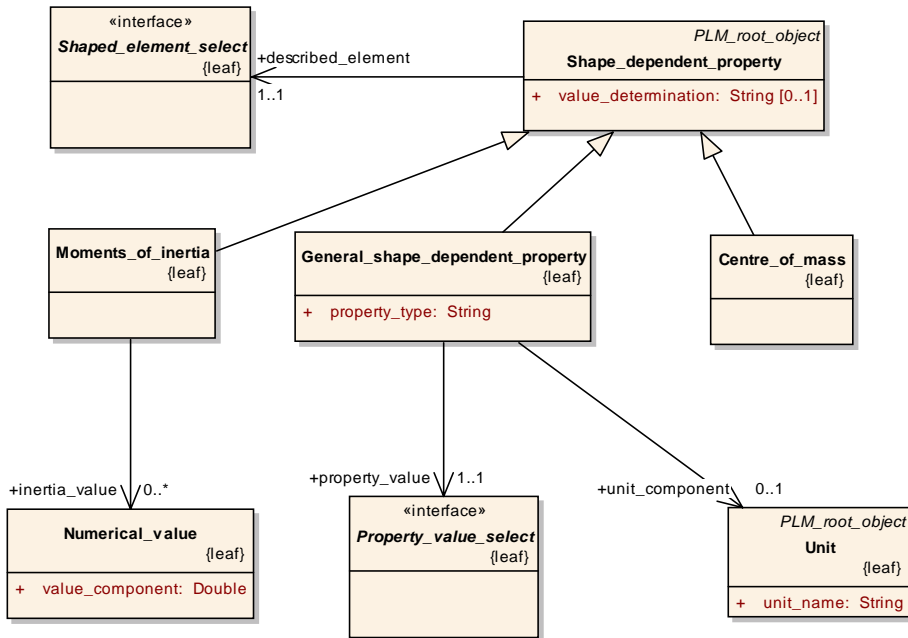


Figure 8.15 - Shape properties

## 8.7.1 Classes

### 8.7.1.1 Class Centre\_of\_mass

A Centre\_of\_mass is the centre of the mass of a body. The relative position of the Centre\_of\_mass within the body is an invariant datum relative to rotation and translation. The Centre\_of\_mass is a characteristic of an item, not its shape.

NOTE 1: If a shape representation is present, the Centre\_of\_mass is usually derivable from the shape representation. However, as a characteristic of an item, it may be pre-defined in order to reflect a requirement.

NOTE 2: In order to convey the information about the actual centre of mass of a geometric model, the use of General\_shape\_dependent\_property with value 'centroid' for its attribute 'property\_type' is appropriate.

A Centre\_of\_mass is a type of Shape\_dependent\_property.

#### **Base Class**

- Shape\_dependent\_property (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- centre\_point: Cartesian\_point[1]  
The centre\_point specifies a point in three-dimensional space that defines the location of a Centre\_of\_mass in the Cartesian\_coordinate\_space.

### 8.7.1.2 Class Cost\_property

A Cost\_property is a property that specifies costs.

#### **Base Class**

- Property (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- none

### 8.7.1.3 Class Data\_environment

A Data\_environment is the specification of the conditions under which a Material\_property\_value\_representation is valid.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- environment\_name : String [1]  
The environment\_name specifies the word or group of words by which the Data\_environment is referred to.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Data\_environment.

#### **Associations**

- none

### 8.7.1.4 Class Duration\_property

A Duration\_property is a property that specifies a period of time during which a given object is used or will last.

#### **Base Class**

- Property (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- none

### 8.7.1.5 Class Feature\_parameter

A Feature\_parameter is the representation of a characteristic of a Feature\_definition.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- parameter\_name: string[0..1]  
The parameter\_name specifies the character, abbreviation, or word by which the Feature\_parameter is referred to.  
EXAMPLE: 'a,' 'b1,' and 'r' are typical examples for the parameter\_name. The parameter\_name need not be specified for a particular Feature\_parameter.

### **Compositions**

- parameter\_value: value\_with\_unit[1]:  
The parameter\_value specifies the value of the Feature\_parameter.  
NOTE: This includes information about units and possibly about limitations such as tolerances.

### **Associations**

- none

#### **8.7.1.6 Class General\_parameter**

A General\_parameter is the association of a Feature\_parameter with a General\_feature in a particular role.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- parameter\_role: string[1]  
The parameter\_role specifies the kind of parameter that is represented by the General\_parameter.  
EXAMPLE: 'width,' 'diameter,' or 'font size' are examples for the parameter\_role.

### **Compositions**

- parameter\_value: feature\_parameter[1]  
The parameter\_value specifies the Feature\_parameter that has the value and possibly a name of the General\_parameter.

### **Associations**

- none

#### **8.7.1.7 Class General\_property**

A General\_property is the definition of a property that is specified by the attribute 'property\_type.'

### **Base Class**

- Property (ABS)

### **Attributes**

- property\_type : String [1]  
The property\_type specifies the kind of property the General\_property defines. Where applicable the following values shall be used:

'overall axle distance'	The overall axle distance is the distance between the first front axle and the rear most axle of the vehicle combination.
'positioning'	The General_property is the definition of a Model_property_value that provides a geometric model for a Product_component or an Item_instance for the purpose of placement.
'theoretical wheelbase'	The theoretical wheelbase is the distance between the resolved weight lines of front and rear axle combinations.
'track'	The track is the distance between the centre of the tires mounted on an axle of a vehicle.
'wheel space'	The wheel space is the distance between the perpendicular lines constructed to the longitudinal median plane of the vehicle from two points that represent the wheels situated at the same side of the axle that is of interest.

### **Compositions**

- none

### **Associations**

- none

#### **8.7.1.8 Class General\_shape\_dependent\_property**

A General\_shape\_dependent\_property is a user-defined characteristic of an object. The 'property\_value' specified by a General\_shape\_dependent\_property is derived from geometry.

NOTE: The General\_shape\_dependent\_property may be used for the purpose of validation of geometric models.

EXAMPLE: If a geometry model is exchanged between two partners, the calculation of various properties of the bodies, such as centre of mass, overall surface, or overall volume, is performed by originator and recipient. When comparing these set of values, the existence of deficiencies in the received model can be easily detected.

A General\_shape\_dependent\_property is a type of Shape\_dependent\_property.

### **Base Class**

- Shape\_dependent\_property (ABS)

### **Attributes**

- property\_type : string[1]  
The property\_type defines the type of characteristic that is specified for an object.

### **Compositions**

- none

### **Associations**

- `property_value` : `Property_value_select`[1]  
The `property_value` specifies the value that is given for a particular characteristic.
- `unit_component` : `Unit`[0..1]  
The `unit_component` specifies the unit in which the `General_shape_dependent_property` is expressed.

#### **8.7.1.9 Class `Item_property_association`**

An `Item_property_association` is a mechanism to associate a property value with an object.

### **Base Class**

- `Property_value_association` (ABS)

### **Attributes**

- `definitional` : `Boolean` [0..1]  
The `definitional` specifies whether the associated `Property_value_representation` object may be used to distinguish the `described_element` from others of the same kind. A value of 'true' indicates that the associated `Property_value_representation` distinguishes it from others.

### **Compositions**

- none

### **Associations**

- `described_element` : `Item_property_select` [1]  
The `described_element` specifies the object that is characterized by the `Property_value`.

#### **8.7.1.10 Class `Mass_property`**

A `Mass_property` is a quantity of matter that an object consists of.

### **Base Class**

- `Property` (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

#### **8.7.1.11 Class `Material_property`**

A `Material_property` is a characteristic that depends on material aspects.

### **Base Class**

- Property (ABS)

### **Attributes**

- property\_name : String [1]  
The property\_name specifies the kind of Material\_property.

### **Compositions**

- none

### **Associations**

- none

### **8.7.1.12 Class Material\_property\_association**

A Material\_property\_association is an object that associates a Material object with a Material\_property\_value\_representation object.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- definitional : Boolean [0..1]  
The definitional specifies whether the associated\_property\_value may be used to distinguish the described\_material from others of the same kind. A value of 'true' indicates that the Material\_property\_value\_representation distinguishes the 'described\_element' from others.

### **Compositions**

- none

### **Associations**

- associated\_property\_value : Material\_property\_value\_representation [1]  
The associated\_property\_value specifies the associated Material\_property\_value\_representation.

### **8.7.1.13 Class Material\_property\_value\_representation**

A Material\_property\_value\_representation is the representation of a characteristic of a material.

### **Base Class**

- Property\_value\_representation

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- environment\_condition : Data\_environment [1]  
The environment\_condition specifies the environmental conditions in which the defining Material\_property\_value\_representation is applicable.

#### **8.7.1.14 Class Model\_property\_association**

A Model\_property\_association is an association of a Model\_property\_value with objects.

EXAMPLE: In the case of a welding operation, the welding points as a part of a wireframe model can be associated with a process operation to describe the locations where the robot has to weld for a welding process operation.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- described\_element: Model\_property\_select[1]  
The described\_element specifies the application object for which a Model\_property\_value is provided.
- validity\_context: Validity\_context\_select[0..1]  
The validity\_context specifies the context in which a Model\_property\_association is applicable.

#### **8.7.1.15 Class Model\_property\_value**

A Model\_property\_value is a mechanism to assign a General\_property to an External\_model, a Geometric\_model, or a Styled\_model.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- none

### **Compositions**

- definition : general\_property[1]  
The definition specifies the General\_property that is assigned to an External\_model, a Geometric\_model, or a Styled\_model.



- model\_property\_association : model\_property\_association[0..\*]

### **Associations**

- describing\_model : Model\_property\_representation\_select[1]  
The describing\_model specifies the model to which a General\_property is assigned.

### **8.7.1.16 Class Moments\_of\_inertia**

A Moments\_of\_inertia describes the matrix of inertia of a rigid body. The moments of inertia I are described as follows

$$I = \begin{pmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{pmatrix}$$

A Moments\_of\_inertia is a type of Shape\_dependent\_property.

### **Base Class**

- Shape\_dependent\_property (ABS)

### **Attributes**

- none

### **Compositions**

- reference\_axis\_placement : axisplacement[1]  
The reference\_axis\_placement specifies the Axis\_placement that was used to calculate the 'inertia\_value.'

### **Associations**

- inertia\_value : Numerical\_value[3][3]  
The inertia\_value specifies the 9 inertia coefficients. NOTE: The matrix defined by the 'inertia\_value' is symmetric with respect to the main diagonal. For that reason, the values of the coefficients are equal, if mirrored along this diagonal. As a consequence, there are 6 different values in the 'inertia\_value.'

### **8.7.1.17 Class Numerical\_value**

A Numerical\_value is a quantity expressed with a numerical value and a unit.

### **Base Class**

- Value\_with\_unit (ABS)

### **Attributes**

- value\_component : Double [1]  
The value\_component specifies the quantity of the Numerical\_value.

### **Compositions**

- none

### **Associations**

- none

### **8.7.1.18 Class Property (ABS)**

A Property is the definition of a particular quality.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [1]  
The id specifies the identifier of the Property.
- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of a Property.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Property.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Property.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Property.

#### **Associations**

- property\_source : Property\_source\_select [0..1]  
The property\_source specifies the External\_library\_reference or Plib\_property\_reference object that defines this kind of property.
- allowed\_unit : Unit [0..\*]  
The allowed\_unit specifies the unit or set of units that are accepted.

### **8.7.1.19 Class Property\_change**

A Property\_change is a mechanism to describe the differences between two objects concerning the properties of these objects.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- id: string[0..1]  
The id specifies the identifier of the Property\_change. The id need not be specified for a particular Property\_change.

### **Compositions**

- description: String\_select[1]  
The description specifies additional information about the Property\_change.

### **Associations**

- is\_describing: Change[1]:  
The is\_describing specifies the Change that collects the Property\_change objects and the Model\_change objects.
- added\_property: Property\_value\_representation[0..\*]  
The added\_property specifies the set of Property\_value\_representation objects that have been added to that object of the described relationship that is identified as relating.
- deleted\_property: Property\_value\_representation[0..\*]  
The deleted\_property specifies the Property\_value\_representation objects that have been deleted from that object of the described relationship that is identified as relating.

#### **8.7.1.20 Class Property\_relationship**

A Property\_relationship is a relationship between two Property objects.

EXAMPLE: Property\_relationship may be used to indicate that the value of one Property can be derived from the value of another Property.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- relation\_type: string[1]:  
The relation\_type specifies the meaning of the relationship.

### **Compositions**

- description: String\_select[0..1]:  
The description specifies additional information about the Property\_relationship.

### **Associations**

- related : Property[1]  
The related specifies the second of the two Property objects related by the Property\_relationship.  
NOTE: The semantics of this attribute are defined by the attribute relation\_type.

#### **8.7.1.21 Class Property\_value (ABS)**

A Property\_value is the numerical or textual value of a Property\_value\_representation.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- value\_name : String [1]  
The value\_name specifies the word or group of words by which the Property\_value is referred to.

### **Compositions**

- property\_value\_representation : Property\_value\_representation [0..\*]  
The property\_value\_representation specifies the property\_value\_representation that is qualified by this Property\_value, by a Value\_with\_unit, a String\_value, or an arbitrary aggregate thereof.

### **Associations**

- none

### **8.7.1.22 Class Property\_value\_association (ABS)**

A Property\_value\_association is a mechanism to assign a Property\_value\_representation to an object.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- none

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Property\_value\_association.

#### **Associations**

- validity\_context : Validity\_context\_select [0..1]  
The validity\_context specifies the context in which a Property\_value\_association is applicable.

### **8.7.1.23 Class Property\_value\_representation**

A Property\_value\_representation is the representation of Property.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- value\_determination : String [0..1]The value\_determination specifies information on how the Property\_value\_representation shall be interpreted. Where applicable the following values shall be used:

'calculated'	The value has been calculated.
'designed'	The value represents a value intended by the design.
'estimated'	The value has been estimated.

'measured'	The value has been measured.
'required'	The value represents a requirement.
'set point'	The value is used as the initialization value.

- qualifier : String [0..1]  
The qualifier specifies the kind of the Property\_value\_representation. The following values shall be used:

'nominal'	The value is the nominal value.
'specified'	The value is specified.
'typical'	The value is a typical value.

### **Compositions**

- property\_value\_association : Property\_value\_association (ABS) [0..\*]  
The property\_value\_association specifies the property\_value\_association that this object is assigned to.

### **Associations**

- definition : Property (ABS) [1]  
The definition specifies the Property that the Property\_value\_representation characterizes.  
If the Property\_value\_representation is a Material\_property\_value\_representation, the definition shall specify a Material\_property.
- global\_unit : Unit [0..1]  
The global\_unit specifies a unit that is valid for all Property\_value that are referenced as 'specified\_value' by the Property\_value\_representation.

#### **8.7.1.24 Class Property\_value\_representation\_relationship**

A Property\_value\_representation\_relationship is a relationship between two Property\_value objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- relation\_type: string[1]  
The relation\_type specifies the meaning of the relationship.  
EXAMPLE: Property\_value\_representation objects representing 'cycle time,' 'feed time,' and 'drill time' for a process may be related by a Property\_value\_representation\_relationship with relation\_type 'dependency.'

### **Compositions**

- description: String\_select[0..1]  
The description specifies additional information about the Property\_value\_representation\_relationship.

### **Associations**

- related : Property\_value\_representation[1]  
The related specifies the second of the two objects related by the Property\_value\_representation\_relationship.  
NOTE: The semantics of this attribute is defined by the attribute relation\_type.

#### **8.7.1.25 Class Quality\_property**

A Quality\_property is a property that enables to provide information about the level of quality of products or processes.

### **Base Class**

- Property (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

#### **8.7.1.26 Class Recyclability\_property**

A Recyclability\_property is information concerning the ability to reuse objects or components of objects after their primarily intended usage.

### **Base Class**

- Property (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- none

#### **8.7.1.27 Class Shape\_dependent\_property**

A Shape\_dependent\_property is a characteristic of the shape, or of a portion of the shape of an object. This object is either an item, an occurrence of an item in the product structure, or the physical realization of an item.

NOTE: A Shape\_dependent\_property is independent of the representations of the shape.

Each Shape\_dependent\_property is a Centre\_of\_mass, a Moments\_of\_inertia, or a General\_shape\_dependent\_property.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- value\_determination: string[0..1]  
The value\_determination specifies information on how the Shape\_dependent\_property shall be interpreted.  
NOTE: A Shape\_dependent\_property may be specified in the design stage of a product but it may also be documented as measured on a prototype. The value\_determination need not be specified for a particular Shape\_dependent\_property.

### **Compositions**

- description: String\_select[0..1]  
The description specifies additional information about the Shape\_dependent\_property.

### **Associations**

- described\_element : Shaped\_element\_select[1]  
The described\_element specifies the object that the Shape\_dependent\_property is associated to.
- is\_defined\_in : Cartesian\_coordinate\_space[0..1]  
The is\_defined\_in specifies the Cartesian\_coordinate\_space in which the property is applicable.  
NOTE: The values of geometry related properties are specified relative to a Cartesian\_coordinate\_space.  
EXAMPLE: A Centre\_of\_mass may be specified in different coordinate spaces for different dimensioning applications.

## **8.7.1.28 Class Simple\_property\_association**

A Simple\_property\_association holds a type and relates a property value by one of the subtypes of property\_value to an instance of simple\_property\_select.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- value\_type : String [1]  
The property\_type specifies the kind of property the Property\_value defines. Where applicable the following values shall be used:

'cost'	The cost of an object.
'duration'	The duration specifies a period of time during which a given object is used or will last.
'mass':	The mass is the quantity of matter that an object consists of.
'overall axle distance'	The overall axle distance is the distance between the first front axle and the rear most axle of the vehicle combination.

'positioning'	The General_property is the definition of a Model_property_value that provides an a geometric model for a Product_component or an Item_instance for the purpose of placement.
'quality'	The quality of products or processes.
'recyclability'	The recyclability is the ability to reuse objects or components of objects after their primarily intended usage.
'theoretical wheelbase'	The theoretical wheelbase is the distance between the resolved weight lines of front and rear axle combinations.
'track'	The track is the distance between the centre of the tyres mounted on an axle of a vehicle.
'wheel space'	The wheel space is the distance between the perpendicular lines constructed to the longitudinal median plane of the vehicle from two points that represent the wheels situated at the same side of the axle that is of interest.

### **Compositions**

- none

### **Associations**

- specified\_value : Property\_value[1]  
The specified\_value denotes the concrete subtype instance of Property\_value this simple\_property\_association relates to the described simple\_property\_select.

#### **8.7.1.29 Class String\_value**

A String\_value represents a sequence of one or more alphanumeric characters.

### **Base Class**

- Property\_value (ABS)

### **Attributes**

- none

### **Compositions**

- value\_specification : String\_select [1]  
The value\_specification specifies the string represented by the String\_value.

### **Associations**

- none

#### **8.7.1.30 Class Unit**

A Unit is a quantity chosen as a standard in terms of which other quantities may be expressed.



### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- unit\_name : String [1]  
The unit\_name specifies the term representing the kind of unit.

### **Compositions**

- none

### **Associations**

- none

### **8.7.1.31 Class Value\_limit**

A Value\_limit is a qualified numerical value representing either the lower limit or the upper limit of a particular physical characteristic.

### **Base Class**

- Value\_with\_unit (ABS)

### **Attributes**

- limit\_qualifier : String [1]  
The limit\_qualifier specifies the kind of limit.
- limit : Double [1]  
The limit specifies the value of the limit.

### **Compositions**

- none

### **Associations**

- none

### **8.7.1.32 Class Value\_list**

A Value\_list is an ordered collection of Property\_value objects.

### **Base Class**

- Property\_value (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- values : Property\_value (ABS) [1..\*]  
The values specifies the ordered collection of Property\_value objects that together are provided as a Property\_value.

### **8.7.1.33 Class Value\_range**

A Value\_range is a pair of numerical values representing the range in which the value shall lie.

#### **Base Class**

- Value\_with\_unit (ABS)

#### **Attributes**

- upper\_limit : Double [1]  
The upper\_limit specifies the maximum acceptable value that is constrained by the Value\_range.
- lower\_limit : Double [1]  
The lower\_limit specifies the minimum acceptable value that is constrained by the Value\_range.

#### **Compositions**

- none

#### **Associations**

- none

### **8.7.1.34 Class Value\_with\_unit (ABS)**

A Value\_with\_unit is either a single numerical measure, or a range of numerical measures with upper, lower, or upper and lower bounds.

#### **Base Class**

- Property\_value (ABS)

#### **Attributes**

- significant\_digits : Integer [0..1]  
The significant\_digits specifies the number of decimal digits that are relevant for the use of the Value\_with\_unit. If present, the numerical measure or range may be specified using more digits than the significant digits but shall not be specified using less digits.

#### **Compositions**

- none

#### **Associations**

- unit\_component : Unit [0..1]  
The unit\_component specifies the unit in which the Value\_with\_unit is expressed.

## 8.7.2 Interfaces

### 8.7.2.1 Interface Item\_property\_select

This empty interface is realized by the following classes:

- Product\_structure\_relationship
- Product\_identification
- Product\_class
- Physical\_instance
- Design\_constraint
- Complex\_product (ABS)
- Document\_representation (ABS)
- Document\_file (ABS)
- Item\_definition\_relationship (ABS)
- Design\_discipline\_item\_definition
- Item\_instance\_relationship (ABS)
- Item\_instance (ABS)
- Item\_definition\_instance\_relationship (ABS)
- Shape\_element\_relationship
- Shape\_element
- Item\_shape

### 8.7.2.2 Interface Simple\_property\_select

#### **Compositions**

- simple\_property\_association : Simple\_property\_association [0..\*]

#### **Extended by**

- Item\_property\_select
- Process\_property\_select

### 8.7.2.3 Interface Property\_source\_select

This empty interface is realized by the following class:

- External\_library\_reference
- Plib\_property\_reference

### 8.7.2.4 Interface Validity\_context\_select

This empty interface is realized by the following classes:

- Organization
- Product\_identification
- Product\_class

## 8.8 Package Alias\_identification

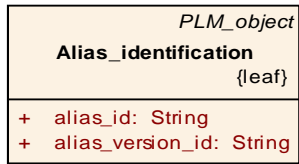


Figure 8.16 - Alias identification

### 8.8.1 Classes

#### 8.8.1.1 Class Alias\_identification

An Alias\_identification is a mechanism to associate an object with an additional identifier used to identify the object of interest in a different context, either in another Organization, or in some other context. The scope of the Alias\_identification shall be specified either by the attribute 'alias\_scope' or by the attribute 'description.'

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- alias\_id : String [1]  
The alias\_id specifies the identifier used in the context specified by the alias\_scope or by the description.
- alias\_version\_id : String [0..1]  
The alias\_version\_id specifies the version of the object as known in the context of the Alias\_identification.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies the type of the Alias\_identification.

#### **Associations**

- alias\_scope : Organization [0..1]  
The alias\_scope specifies the Organization in which the Alias\_identification is valid.

## 8.8.2 Interfaces

### 8.8.2.1 Interface Alias\_select

#### **Compositions**

- alias\_identification : Alias\_identification [0..\*]

#### **Implemented By**

- Organization
- Complex\_product
- Classification\_attribute
- Item
- Document\_type\_property
- Product\_class
- Document\_version
- Specification\_category
- Document
- Specification
- Security\_level
- Item\_version
- Classification\_system
- Item\_instance
- Document\_representation
- Geometric\_model
- Property
- General\_classification
- Design\_discipline\_item\_definition
- Physical\_instance
- Approval\_status

## 8.9 Package Authorization

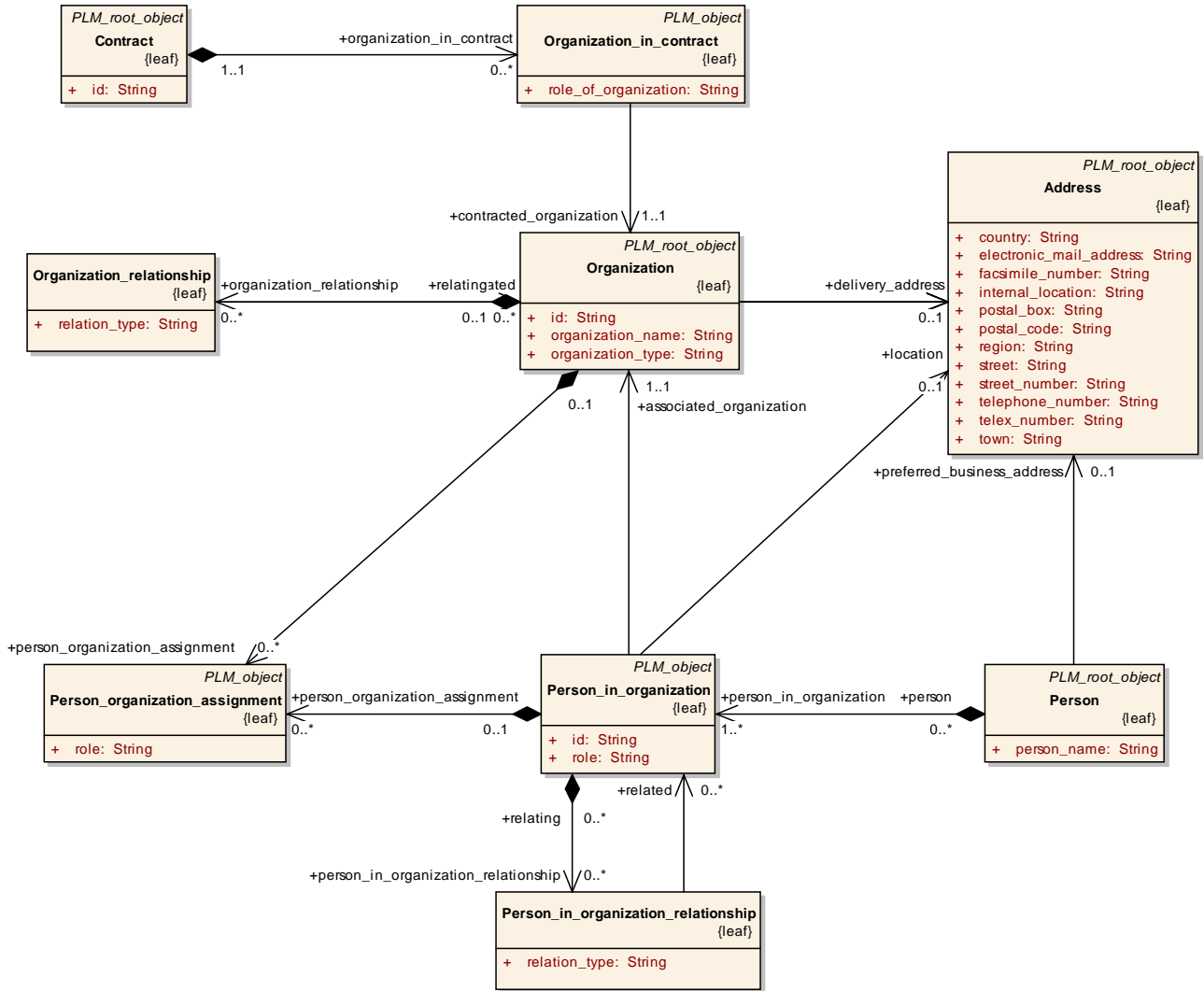


Figure 8.17 - Authorization - Person and organization

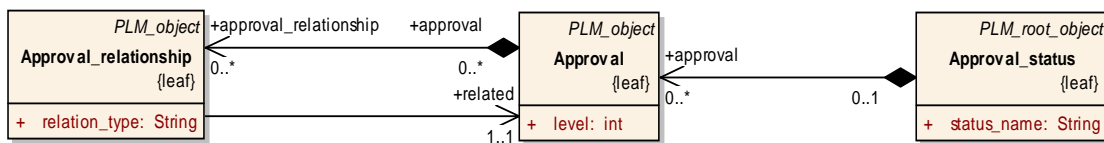


Figure 8.18 - Authorization - Approval

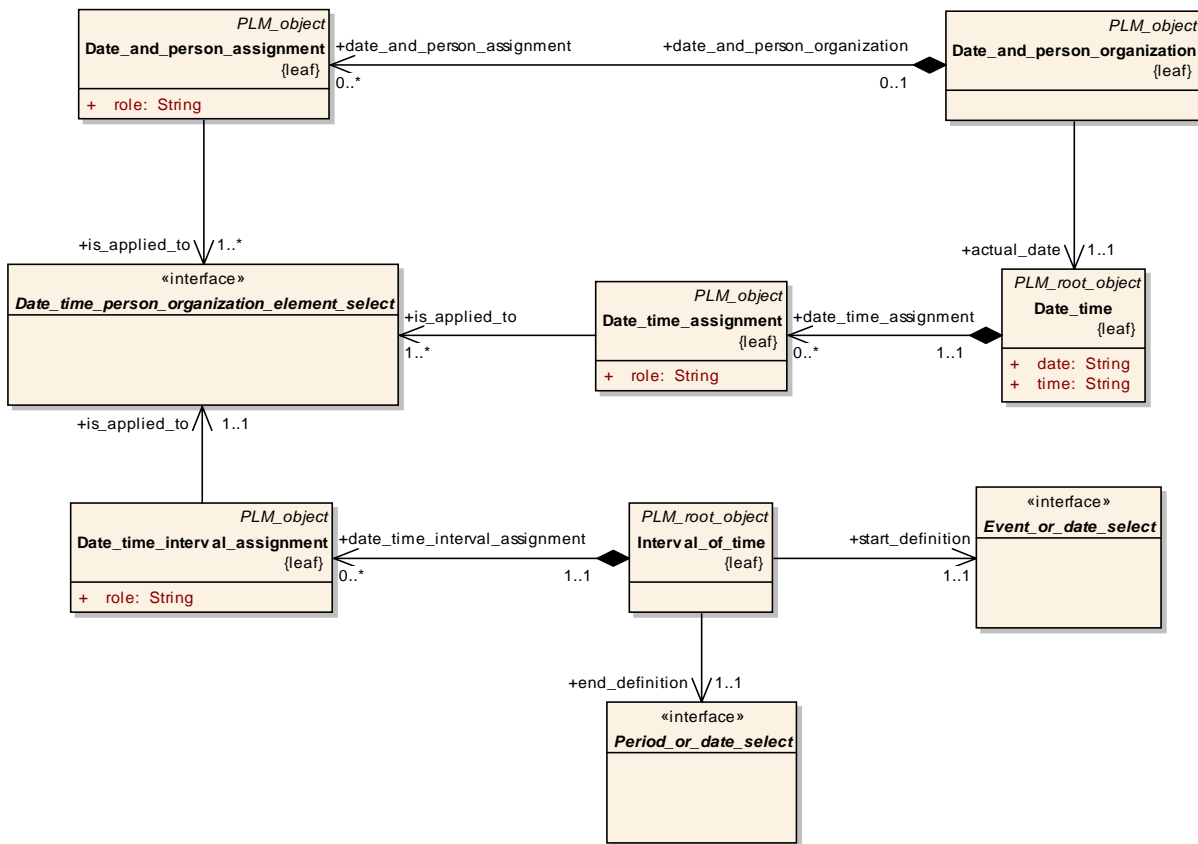


Figure 8.19 - Authorization - Date and time

## 8.9.1 Classes

### 8.9.1.1 Class Address

An Address contains information about how a person or an organization can be contacted.

#### Base Class

- PLM\_root\_object (ABS)

#### Attributes

- internal\_location : String [0..1]  
The internal location.
- street\_number : String [0..1]  
The street number.
- street : String [0..1]  
The street.
- postal\_box : String [0..1]  
The postal box.

- town : String [0..1]  
The town.
- region : String [0..1]  
The region.
- postal\_code : String [0..1]  
The postal code.
- country : String [0..1]  
The country.
- facsimile\_number : String [0..1]  
The fax number.
- telephone\_number : String [0..1]  
The telephone number.
- electronic\_mail\_address : String [0..1]  
The e-mail address.
- telex\_number : String [0..1]  
The telex number.

### **Compositions**

- none

### **Associations**

- none

### **8.9.1.2 Class Approval**

An Approval is a judgement concerning the quality of those product data that are subject of the Approval. An Approval represents a statement made by technical personnel or management personnel whether certain requirements are met. The absence of approval information does not imply any approval status by default.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- level : String [0..1]  
The level represents the aspect for which the object subject to approval, by reference as 'is\_applied\_to,' is endorsed. Where applicable the following values shall be used:

'disposition'	The referenced object is approved for series production.
'equipment order'	The referenced object has reached a status in which changes are subject to a defined change process and tools and other equipment required for production may be ordered.
'planning'	The referenced object is technically complete and has reached a status sufficiently stable so that other designs may be based on it.



### **Compositions**

- approval\_relationship : Approval\_relationship [0..\*]  
The Approval\_relationship specifies the Approval\_relationship that relates the first of the two Approval objects.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Approval.

### **Associations**

- scope : Organization [0..\*]  
The scope specifies the set of Organization objects for which the Approval is valid.
- actual\_date : Date\_time [0..1]  
The actual\_date specifies the date when the Approval actually became valid. If this attribute is absent, the approval has not yet occurred, i.e., it is pending.
- planned\_date : Date\_time [0..1]  
The planned\_date specifies the date when the Approval is or was supposed to be performed.
- is\_approved\_by : Date\_and\_person\_organization [0..\*]  
The is\_approved\_by specifies personnel responsible for the Approval and the dates of the Approval.
- is\_applied\_to : Approval\_element\_select [1..\*]  
The is\_applied\_to specifies the objects to which the Approval is assigned.

### **8.9.1.3 Class Approval\_relationship**

An Approval\_relationship is a relationship between two Approval objects.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship. Where applicable the following values shall be used:

'decomposition'	The Approval_relationship defines a relationship where the related Approval is one of the components into which the relating Approval is broken down with no implication of 'sequence' or 'dependency.'
'dependency'	The Approval_relationship defines a relationship where the issuing of the related Approval is dependent on the issuing of the relating Approval.
'precedence'	The Approval_relationship defines a relationship where the related Approval has higher priority than the relating Approval.
'sequence'	The Approval_relationship defines a relationship where the relating Approval shall be completed before the related Approval is given.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Approval\_relationship.

### **Associations**

- related : Approval [1]  
The related specifies the second of the two Approval objects related by the Approval\_relationship.

#### **8.9.1.4 Class Approval\_status**

An Approval\_status is the state of acceptance of some product data.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- status\_name : String [1]  
The status\_name specifies the terms characterizing the Approval\_status.

### **Compositions**

- approval : Approval [0..\*]  
The Approval indicates the approval that is applied to the level of acceptance of this Approval\_status, for the specified 'level.'
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Approval\_status.

### **Associations**

- used\_classification\_system : Classification\_system [0..1]  
The used\_classification\_system specifies the Classification\_system that contains the information about how to interpret the Approval\_status.

#### **8.9.1.5 Class Certification**

A Certification is a certificate for an object.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- certification\_type: string[1]  
The certification\_type specifies the kind of certification.  
EXAMPLE: 'supplier certificate' is an example for the certification\_type.

### **Compositions**

- name: String\_select[1]:  
The name specifies the word or group of words by which the Certification is referred to.

- purpose: String\_select[0..1]:  
The purpose specifies the objective of the Certification.

### **Associations**

- is\_applied\_to: Certification\_select [1]  
The is\_applied\_to specifies the set of objects that the certificate is applied to.

### **8.9.1.6 Class Contract**

A Contract is a binding agreement concerning the design of Item\_version objects, the delivery of Drawing objects, or the execution of other Activity objects.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id: string[1]  
The id specifies the identifier of the Contract that shall be unique within the scope of an organization.

#### **Compositions**

- description: String\_select[0..1]:  
The description specifies additional information for an Item\_version, a Drawing, or an Activity object according to the underlying Contract.  
EXAMPLE: The description of the Contract may be the ordered price.
- organization\_in\_contract: organization\_in\_contract[0..\*]

#### **Associations**

- contracted\_element: Contracted\_element\_select[0..\*]  
The contracted\_element specifies the object that is subject of the Contract.

### **8.9.1.7 Class Date\_and\_person\_assignment**

A Date\_and\_person\_assignment is an object that associates a Date\_and\_person\_organization with product data. This assignment provides additional information for the associated object.

#### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- role : String [1]  
The role specifies the relationship between the date or time and the person or organization in the Date\_and\_person\_assignment. Where applicable the following values shall be used:

'creation'	The assignment specifies that the referenced object has been created by the given person or organization at the given date and time.
'update'	The assignment specifies that the referenced object has been altered by the given person or organization at the given date and time.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Date\_and\_person\_assignment.

### **Associations**

- is\_applied\_to : Date\_time\_person\_organization\_element\_select [1..\*]  
The is\_applied\_to specifies the set of objects with which the Date\_and\_person\_assignment is associated.

#### **8.9.1.8 Class Date\_and\_person\_organization**

A Date\_and\_person\_organization is a Person\_in\_organization or an Organization associated with a Date\_time or an Event\_reference.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- date\_and\_person\_assignment : Date\_and\_person\_assignment [0..\*]  
The Date\_and\_person\_assignment specifies the Date\_and\_person\_assignment for this Date\_and\_person\_organization.

### **Associations**

- actual\_date : Date\_time [1]  
The actual\_date specifies the date and an optional time of day component of a Date\_and\_person\_organization, or alternatively a discrete point in time as an Event\_reference.

#### **8.9.1.9 Class Date\_time**

A Date\_time is the specification of a date and an optional time of day.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- time : String [0..1]  
The time specifies a moment of occurrence measured by hour, minute, and second.
- date : String [1]  
The date specifies the calendar time, defined according to the Gregorian calendar, conveying information about the year, the month, and the day in no specific order. The representation of a date shall be complete, i.e., millennium, century, and year-within-century data shall be included.

### **Compositions**

- date\_time\_assignment : Date\_time\_assignment [0..\*]  
The Date\_time\_assignment specifies the Date\_time\_assignment that this Date\_time is assigned to.

### **Associations**

- none

### **8.9.1.10 Class Date\_time\_assignment**

A Date\_time\_assignment is an association of point in time specified as a Date\_time or an Event\_reference with product data.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- role : String [1]  
The role specifies the action associated with the Date\_time\_assignment. Where applicable the following values shall be used:

'classification date'	The assignment specifies that the specified object is classified at the given date and time. This value shall only be used, if the Date_time_assignment refers to instances of Classification_association as 'is_applied_to.'
'creation'	The assignment specifies that the referenced object was created at the given date and time.
'installation'	The assignment specifies that the referenced object was mounted in a product at the given date and time.
'production'	The assignment specifies that the referenced object was produced at the given date and time.
'registration'	The assignment specifies that the referenced object was determined at the given date and time.
'update'	The assignment specifies that the referenced object was altered at the given date and time.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Date\_time\_assignment.

### **Associations**

- is\_applied\_to : Date\_time\_person\_organization\_element\_select [1..\*]  
The is\_applied\_to specifies the set of objects of product data with which the Date\_time\_assignment is associated.

#### **8.9.1.11 Class Date\_time\_interval\_assignment**

A Date\_time\_interval\_assignment is an association of an Interval\_of\_time with some product data. The Date\_time\_interval\_assignment shall neither be used to convey effectivity information, i.e., information concerning valid use of product data, nor to convey retention information, nor duration information.

NOTE: Such information is conveyed using Effectivity, Retention\_period, and Duration respectively.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- role: string[1]  
The role specifies the action associated with the Date\_time\_interval\_assignment.

### **Compositions**

- description: String\_select[0..1]  
The description specifies additional information about the Date\_time\_interval\_assignment.

### **Associations**

- is\_applied\_to: Date\_time\_person\_organization\_select[1]  
The is\_applied\_to specifies the set of objects of product data with which the Date\_time\_interval\_assignment is associated.

#### **8.9.1.12 Class Duration**

A Duration is the definition of a period of time.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- time : String [1]  
The time specifies the extend of the Duration.
- time\_unit : String [1]  
The time\_unit specifies the unit in which the time is specified.

### **Compositions**

- none

### **Associations**

- none

#### **8.9.1.13 Class Event\_reference**

An Event\_reference is the definition of a point in time established relative to an event.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- event\_type : String [1]  
The event\_type specifies the kind of event that serves as reference.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Event\_reference.

### **Associations**

- event\_context : General\_organizational\_data\_select [0..1]  
The event\_context specifies the piece of product data the Event\_reference refers to.
- offset : Duration [0..1]  
The offset specifies the amount of time before or after the defined event that shall be used to calculate the actual point in time.

#### **8.9.1.14 Class Interval\_of\_time**

An Interval\_of\_time specifies a period of time.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- none

### **Compositions**

- date\_time\_interval\_assignment: date\_time\_interval\_assignment[0..\*]

### **Associations**

- end\_definition : Period\_or\_data\_select[1]  
The end\_definition defines the end of the Interval\_of\_time, either by referring to a bound, or specifying the extend of

the Interval\_of\_time by a Duration. If the end\_definition refers to an Event\_reference or Date\_time, this particular bound of the resulting interval is excluded from it.

- start\_definition: Event\_or\_data\_select[1]  
The start\_definition defines the beginning of the Interval\_of\_time. The bound specified by the start\_definition is included in the resulting interval.

### 8.9.1.15 Class Organization

An Organization is a group of people involved in a particular business process.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- organization\_name : String [1]  
The organization\_name specifies the word or group of words used to refer to the Organization.
- organization\_type : String [0..1]  
The organization\_type specifies the type of the Organization. Where applicable the following values shall be used:

'company'	The organization_type specifies that the Organization is a company.
'department'	The organization_type specifies that the Organization is a department.
'plant'	The organization_type specifies that the Organization is a plant.
- id : String [1]  
The id specifies the identifier of the Organization.

#### **Compositions**

- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Organization.
- person\_organization\_assignment : Person\_organization\_assignment [0..\*]  
The Person\_organization\_assignment specifies the Person\_organization\_assignment that concerns this Organization.
- date\_and\_person\_organization : Date\_and\_person\_organization [0..\*]  
The Date\_and\_person\_organization specifies the Date\_and\_person\_organization that this Organization is part of.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Organization.

#### **Associations**

- postal\_address : Address [0..1]  
The postal\_address specifies the address where letter mail is delivered.
- delivery\_address : Address [0..1]  
The delivery\_address specifies the address where goods are delivered.
- visitor\_address : Address [0..1]  
The visitor\_address specifies the address where the organization receives visitors.



### 8.9.1.16 Class Organization\_in\_contract

An Organization\_in\_contract is a mechanism to associate the person who is signing a contract and the organization that the person is signing for, with a Contract.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- role\_of\_organization: string[1]  
The role\_of\_organization specifies the function performed by the signing Organization with respect to the contract.

#### **Compositions**

- none

#### **Associations**

- contracted\_organization: Organization[1]  
The contracted\_organization specifies the organization that participates in the contract.
- signature : Date\_and\_person\_organization[0..1]  
The signature specifies the set of Date\_and\_person\_organization objects representing the personnel that signed the contract on behalf of the contracted organization and the date of the signature.

### 8.9.1.17 Class Person

A Person is an individual human being who has some relationship to product data. The Person shall always be identified in the context of one or more organizations.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- person\_name : String [1]  
The person\_name specifies the word or group of words used to refer to the Person.

#### **Compositions**

- person\_in\_organization : Person\_in\_organization [1..\*]  
The Person\_in\_organization specifies the person\_in\_organization that this Person is assigned to.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Person.

#### **Associations**

- preferred\_business\_address : Address [0..1]  
The preferred\_business\_address specifies the location of the office of the Person.

### 8.9.1.18 Class Person\_in\_organization

A Person\_in\_organization is the specification of a Person in the context of an Organization.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- role : String [1]  
The role specifies the relationship between the Person and the Organization.
- id : String [0..1]  
The id specifies an identifier of the person. The identifier shall be unique within the scope of the 'associated\_organization.'
- description : String [0..1]  
The description specifies additional information about the Person\_in\_organization\_relationship. The description need not be specified for a particular Person\_in\_organization\_relationship. If present, there shall be exactly one object that defines the description for a Person\_in\_organization\_relationship.

#### **Compositions**

- person\_organization\_assignment : Person\_organization\_assignment [0..\*]  
The Person\_organization\_assignment specifies the Person\_organization\_assignment that concerns this Person\_in\_organization.
- date\_and\_person\_organization : Date\_and\_person\_organization [0..\*]  
The Date\_and\_person\_organization specifies the Date\_and\_person\_organization that this Person\_in\_organization is part of.

#### **Associations**

- location : Address [0..1]  
The location specifies the relevant address of the Person\_in\_organization.
- associated\_organization : Organization [1]  
The associated\_organization specifies the Organization with which the Person is associated.

### 8.9.1.19 Class Person\_in\_organization\_relationship

A Person\_in\_organization\_relationship is a mechanism that allows specifying a relationship between two persons in an Organization.

The data associated with a Person\_in\_organization\_relationship are the following:

- description
- related
- relating
- relation\_type

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- description : String [1]  
The description specifies additional information about the Person\_in\_organization\_relationship. See Person\_in\_organization\_relationship to Multi\_language\_string for the application assertion. The description need not be specified for a particular Person\_in\_organization\_relationship. If present, there shall be exactly one object that defines the description for a Person\_in\_organization\_relationship.
- related : String [1]  
The related specifies the second of the two objects related by the Person\_in\_organization\_relationship. See Person\_in\_organization\_relationship to Person\_in\_organization for the application assertion.
- relating : String [1]  
The relating specifies the first of the two objects related by the Person\_in\_organization\_relationship. See Person\_in\_organization\_relationship to Person\_in\_organization for the application assertion.
- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship. Where applicable the following values shall be used:  
- 'successor': The related Person\_in\_organization is the successor of the relating Person\_in\_organization.

### **Compositions**

None

### **Associations**

None

## **8.9.1.20 Class Person\_organization\_assignment**

A Person\_organization\_assignment is an object that associates an Organization or a Person\_in\_organization with product data.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- role : String [1]  
The role specifies the responsibility of the assigned Person or Organization with respect to the object that it is applied to. Where applicable the following values shall be used:

'author':	The referenced object has been created by the assigned Person or Organization. The author holds the copyright.
'classification officer'	The assigned Person or Organization is formally responsible for the classification of the referenced object.
'creator'	The referenced object has been created by the assigned Person or Organization.

'custodian'	The assigned Person or Organization is responsible for the existence and integrity of the referenced object.
'customer'	The assigned Person or Organization acts as a purchaser or consumer of the referenced object.
'design supplier'	The assigned Person or Organization is the one who delivers the data de-scribing the referenced object.
'editor'	The assigned Person or Organization is responsible for making any changes to any at-tribute of the referenced object.
'id owner'	The assigned Person or Organization is the one responsible for the designation of an identifier.
'location'	The assigned Organization is the place where the referenced object can be found or where it takes place.
'manufacturer'	The assigned Person or Organization is the one who produces the actual (physical) object.
'owner'	The assigned Person or Organization owns the referenced object, and has final say over its disposition and any changes to it.
'supplier':	The assigned Person or Organization is the one who delivers the actual (physical) object (e.g., a dealer).
'wholesaler'	The assigned Person or Organization is the one who is in the sales chain be-tween the manufacturer and the supplier.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Person\_organization\_assignment.

### **Associations**

- is\_applied\_to : Date\_time\_person\_organization\_element\_select [1..\*]  
The is\_applied\_to specifies the object with which the Person\_organization\_assignment is associated.

### **8.9.1.21 Class Security\_classification**

A Security\_classification is the level of confidentiality that is required in order to protect product data against unauthorized usage.

#### **Base Class:**

- PLM\_object (ABS)

#### **Attributes**

- none

### **Compositions**

- name: String\_select[1]  
The name specifies the word or group of words used to refer to the Security\_classification.

- purpose: String\_select[0..1]  
The purpose specifies the rationale behind the Security\_classification.

### **Associations**

- is\_applied\_to: Security\_element\_select[1]  
The is\_applied\_to specifies the objects with which the Security\_classification is associated.

### **8.9.1.22 Class Security\_level**

A Security\_level is the specification of a level of security within some security classification scheme.

NOTE: The values of Security\_level are company specific.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- level\_name: string[1]  
The level\_name specifies the word or abbreviation used to refer to the Security\_level.

### **Compositions**

- security\_classification: security\_classification[0..\*]

### **Associations**

- used\_classification\_system : Classification\_system[0..1]  
The used\_classification\_system specifies the Classification\_system that contains the information about how to interpret the level of the Security\_level.

## **8.9.2 Interfaces**

### **8.9.2.1 Interface Approval\_element\_select**

This empty interface is realized by the following classes:

- Work\_request
- Work\_order
- Project
- Activity\_method\_assignment
- Activity\_element
- Activity
- General\_classification
- Classification\_system
- Classification\_association
- Specification\_inclusion
- Specification\_expression

- Specification\_category
- Specification
- Product\_structure\_relationship
- Product\_class
- Physical\_instance\_test\_result
- Physical\_instance
- Manufacturing\_configuration (ABS)
- Design\_constraint
- Configuration
- Complex\_product (ABS)
- Class\_structure\_relationship
- Class\_specification\_association
- Class\_inclusion\_association
- Class\_condition\_association
- Class\_category\_association
- Document\_version
- Document\_representation (ABS)
- Document\_file (ABS)
- Document
- Item\_version
- Item\_definition\_relationship (ABS)
- Design\_discipline\_item\_definition
- Physical\_assembly\_relationship
- Item\_instance\_relationship (ABS)
- Item\_instance (ABS)
- Item\_definition\_instance\_relationship (ABS)
- Assembly\_substitute\_relationship
- Process\_plan
- Certification
- Contract
- Property\_value\_association (ABS)
- Simple\_property\_association
- Property (ABS)
- Material
- Geometric\_model

### **8.9.2.2 Interface Date\_time\_person\_organization\_element\_select**

This empty interface is realized by the following classes:

- Person\_in\_organization
- Event\_reference
- Approval\_status
- Work\_request
- Work\_order
- Project
- Activity\_method\_assignment
- Activity\_element
- Activity
- General\_classification
- Classification\_system
- Classification\_association
- Specification\_inclusion
- Specification\_expression
- Specification\_category
- Specification
- Security\_level
- Contract
- Product\_structure\_relationship
- Product\_identification
- Product\_class
- Physical\_instance\_test\_result
- Physical\_instance
- Manufacturing\_configuration (ABS)
- Design\_constraint
- Configuration
- Complex\_product\_relationship
- Complex\_product (ABS)
- Class\_structure\_relationship
- Class\_specification\_association
- Class\_inclusion\_association
- Class\_condition\_association
- Class\_category\_association
- Document\_version
- Document\_representation (ABS)
- Document\_file (ABS)
- Document
- Item\_version\_relationship

- Item\_version
- Item\_definition\_relationship (ABS)
- Item
- Design\_discipline\_item\_definition
- Physical\_assembly\_relationship
- Item\_instance\_relationship (ABS)
- Item\_instance (ABS)
- Item\_definition\_instance\_relationship (ABS)
- Process\_plan
- Process\_operation\_resource\_assignment
- Process\_operation\_occurrence
- Process\_operation\_definition
- Property\_value\_association (ABS)
- Simple\_property\_association
- Property (ABS)
- Material
- Assembly\_substitute\_relationship
- Geometric\_model

### **8.9.2.3 Interface Event\_or\_date\_select**

This empty interface is realized by the following classes:

- Event\_reference
- Date\_time

### **8.9.2.4 Interface General\_organizational\_data\_select**

This empty interface is realized by the following classes:

- Person\_in\_organization
- Approval\_status
- Work\_request
- Work\_order
- Project
- Activity\_method\_assignment
- Activity\_element
- Activity
- General\_classification
- Classification\_system
- Classification\_association
- Specification\_inclusion
- Specification\_expression



- Specification\_category
- Specification
- Product\_structure\_relationship
- Product\_identification
- Product\_class
- Physical\_instance\_test\_result
- Physical\_instance
- Manufacturing\_configuration (ABS)
- Design\_constraint
- Configuration
- Complex\_product\_relationship
- Complex\_product (ABS)
- Class\_structure\_relationship
- Class\_specification\_association
- Class\_inclusion\_association
- Class\_condition\_association
- Class\_category\_association
- Document\_version
- Document\_representation (ABS)
- Document\_file (ABS)
- Document
- Item\_version\_relationship
- Item\_version
- Item\_definition\_relationship (ABS)
- Item
- Design\_discipline\_item\_definition
- Physical\_assembly\_relationship
- Item\_instance\_relationship (ABS)
- Item\_instance (ABS)
- Item\_definition\_instance\_relationship (ABS)
- Process\_plan
- Process\_operation\_resource\_assignment
- Process\_operation\_occurrence
- Process\_operation\_definition
- Property\_value\_association (ABS)
- Property (ABS)
- Material
- Geometric\_model

### 8.9.2.5 Interface Period\_or\_date\_select

This empty interface is realized by the following classes:

- Event\_reference
- Duration
- Date\_time

### 8.9.2.6 Interface Person\_organization\_select

#### Compositions

- date\_and\_person\_organization : Date\_and\_person\_organization [0..\*]
- person\_organization\_assignment : Person\_organization\_assignment [0..\*]

#### Implemented By

- Person\_in\_organization
- Organization

## 8.10 Package Configuration\_management

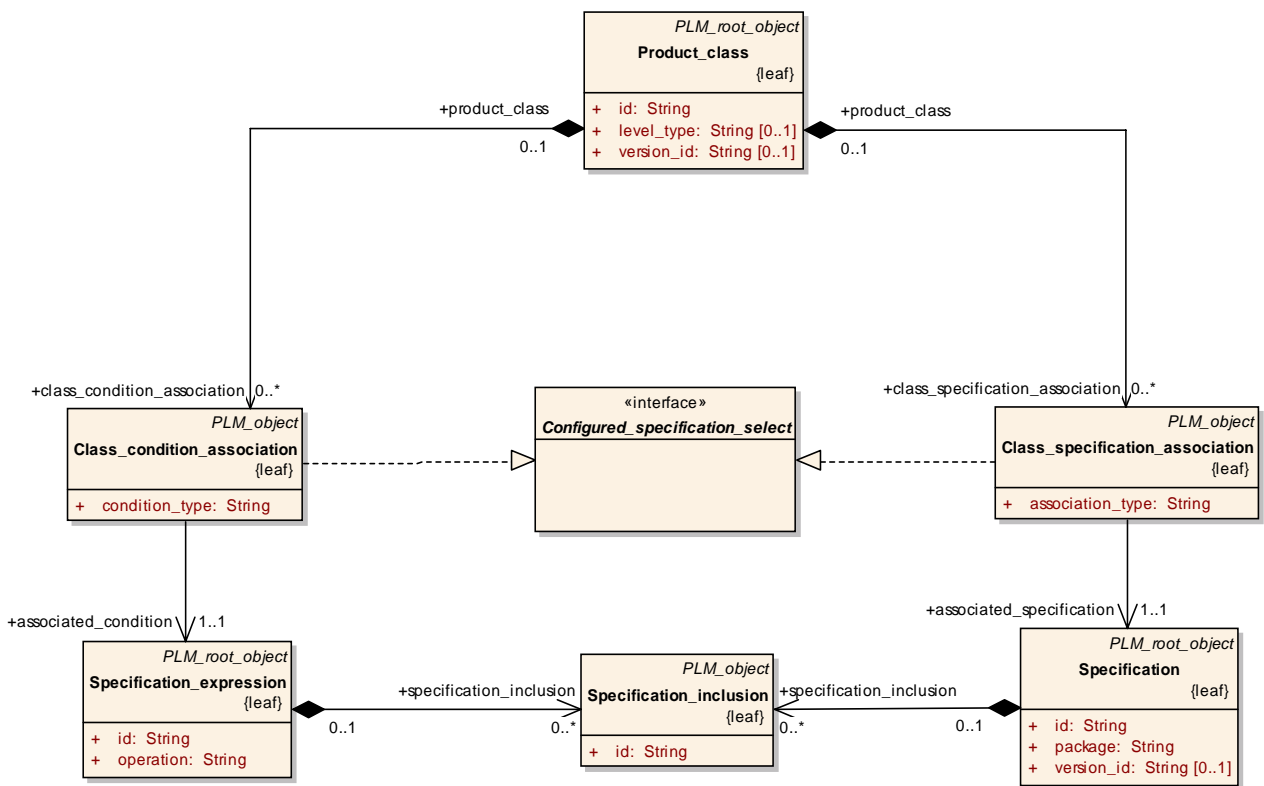


Figure 8.20 - Configuration management - Product class condition and specification

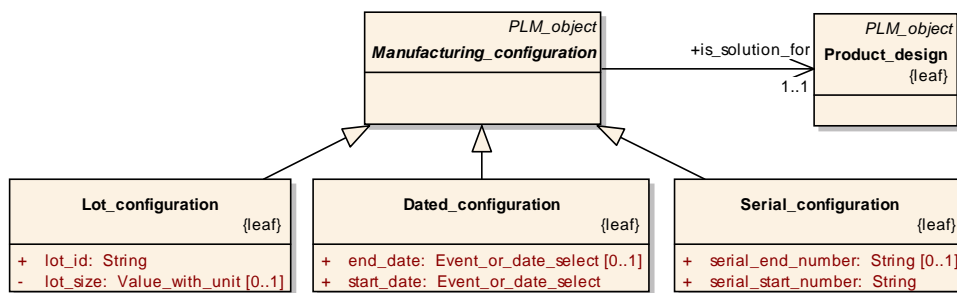


Figure 8.21 - Configuration management - manufacturing configuration

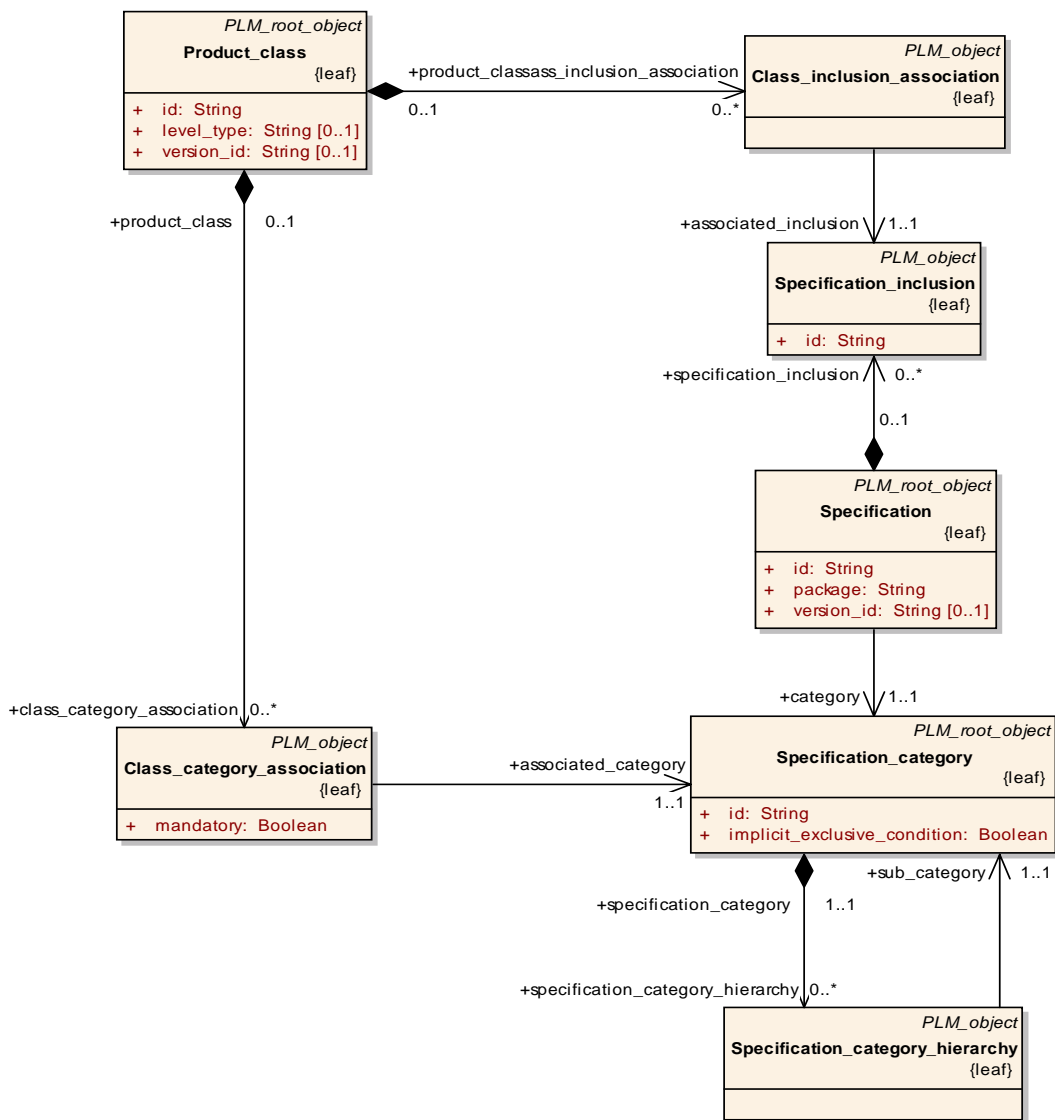


Figure 8.22 - Configuration management - specification category and inclusion

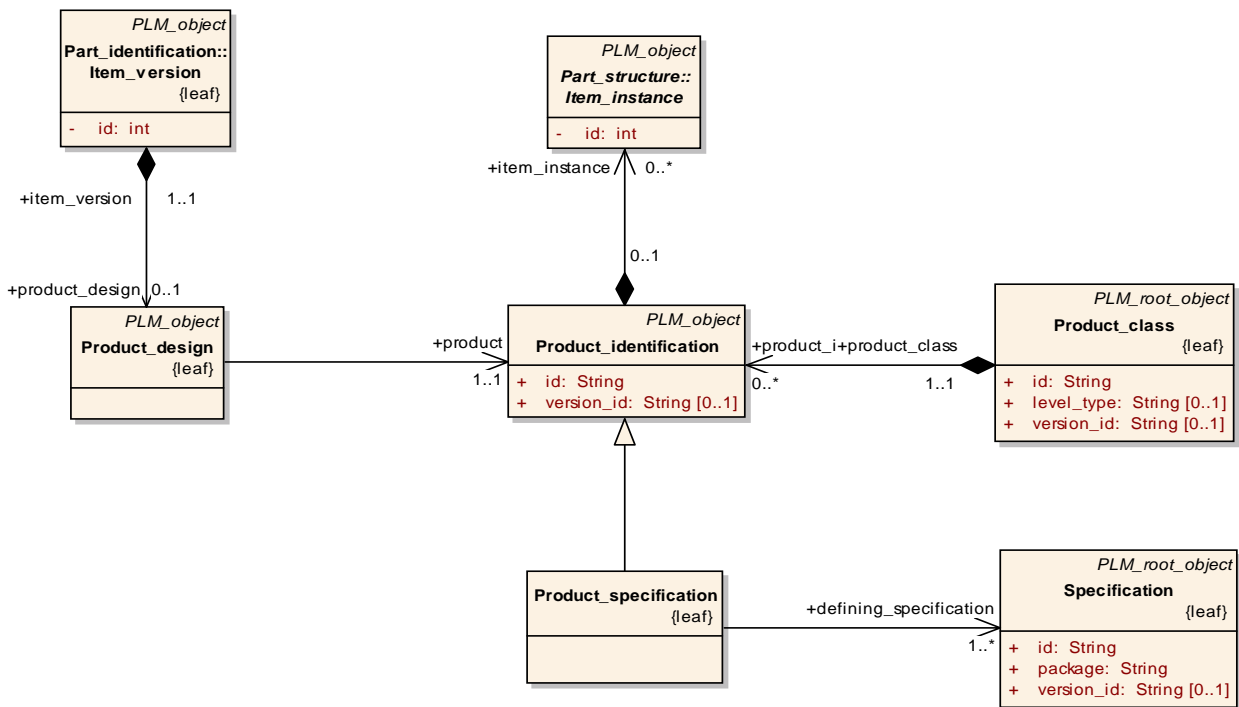


Figure 8.23 - Configuration management - Product identification

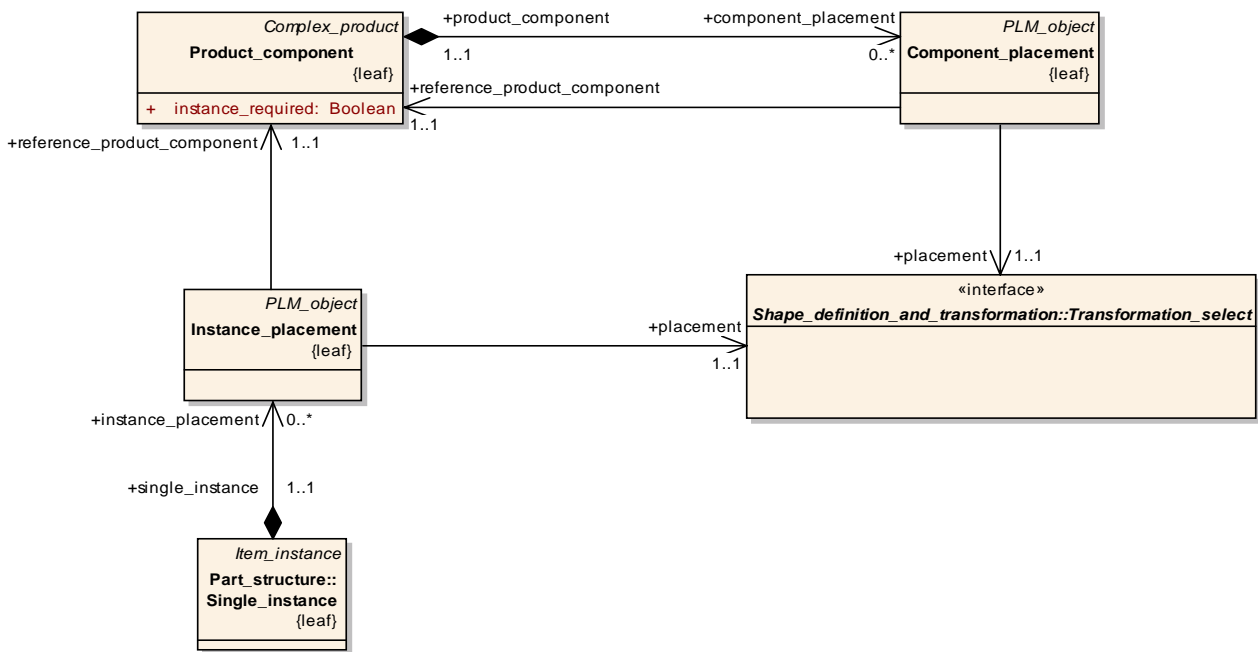


Figure 8.24 - Configuration management - Component and instance placement

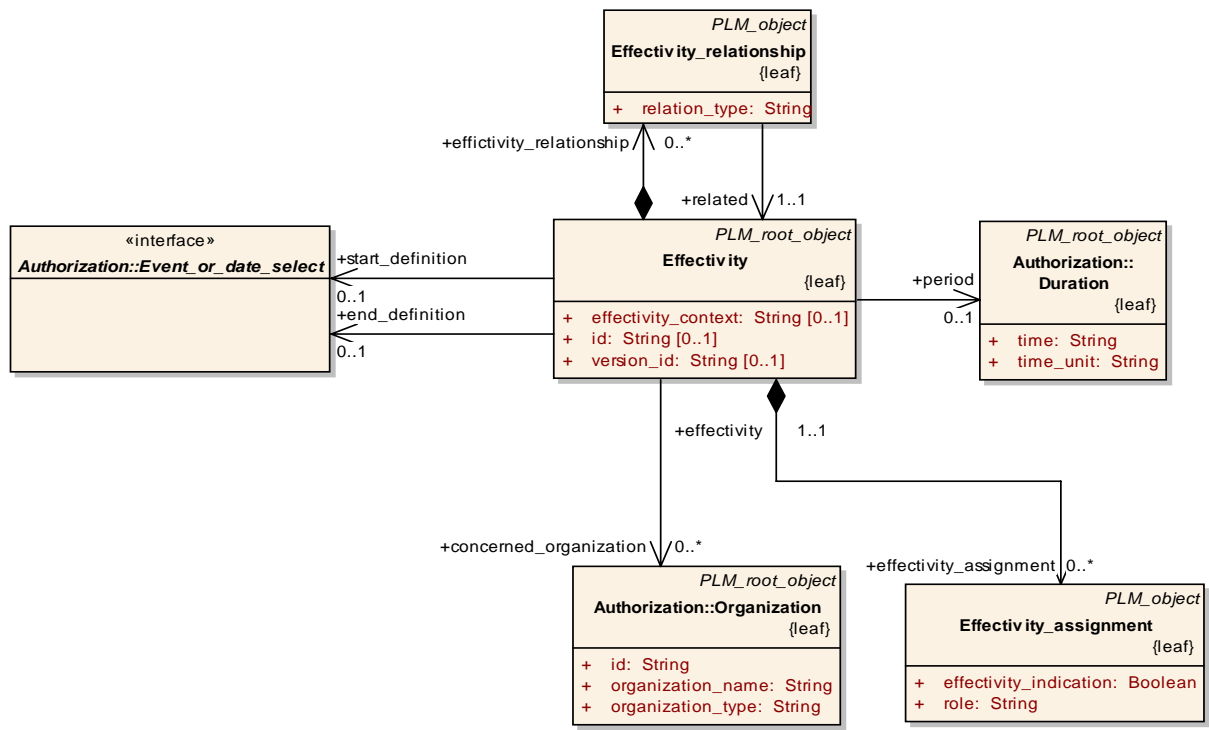


Figure 8.25 - Configuration management - Effectivity

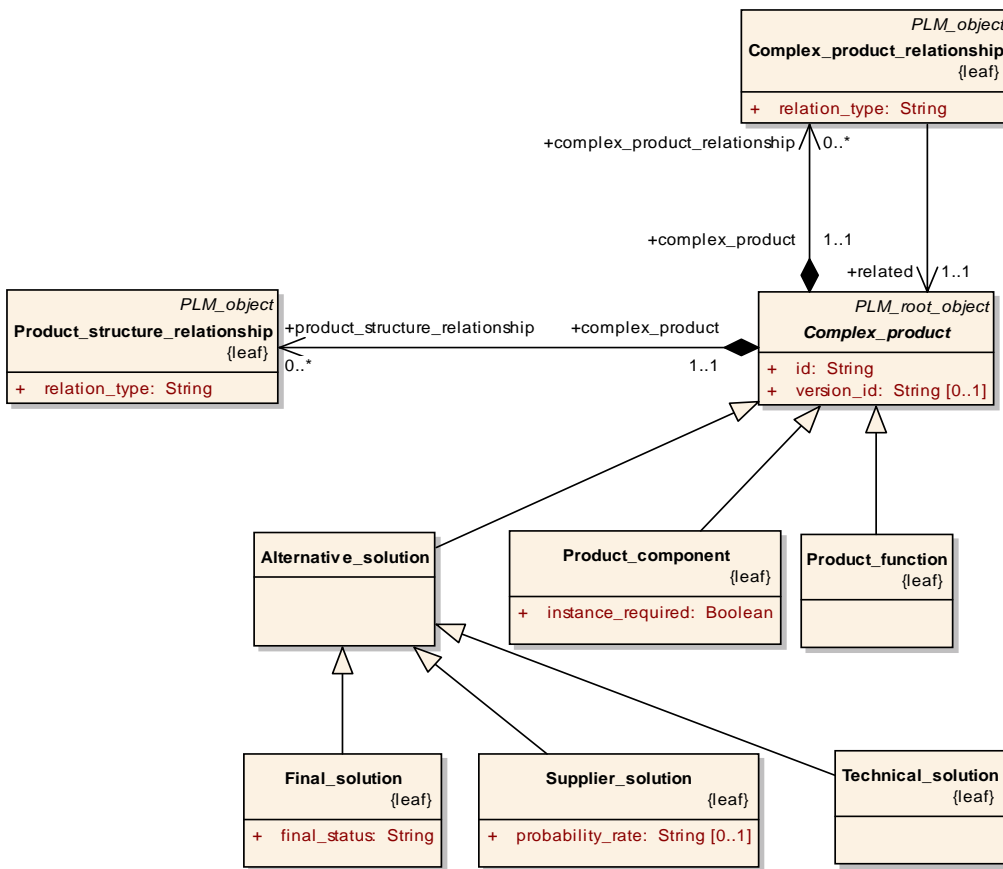


Figure 8.26 - Configuration management - Complex product

## 8.10.1 Classes

### 8.10.1.1 Class Alternative\_solution

An Alternative\_solution is the identification of one of potentially many mutually exclusive implementations of a Product\_function or of a Product\_component.

#### Base Class

- Complex\_product (ABS)

#### Attributes

- none

#### Compositions

- configuration : Configuration [0..\*]  
The configuration specifies the configuration that controls this Alternative\_solution for its valid usage.

### **Associations**

- `base_element` : `Complex_product_select` [1]  
The `base_element` specifies the object, for which the `Alternative_solution` provides a design alternative. All `Alternative_solution` objects for the same `base_element` are mutually exclusive.

#### **8.10.1.2 Class `Class_category_association`**

A `Class_category_association` is the association of a `Specification_category` with a `Product_class`. Additionally, this assignment specifies if the usage of one or more `Specification` objects belonging to this `Specification_category`, is mandatory or optional for all products of that `Product_class`.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `mandatory` : `Boolean` [1]  
The `mandatory` specifies whether the `Specification` objects referring to the associated `Specification_category` have to be used or may be used (optional) for products within the referenced `Product_class`. A value of 'true' indicates that the usage is mandatory.

### **Compositions**

- none

### **Associations**

- `associated_category` : `Specification_category` [1]  
The `associated_category` specifies the `Specification_category` that is associated with the `Product_class`.

#### **8.10.1.3 Class `Class_condition_association`**

A `Class_condition_association` is the association of a `Specification_expression` with a `Product_class`.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `condition_type` : `String` [1]  
The `condition_type` specifies the meaning of the association. Where applicable the following values shall be used:

'design case'	The <code>Specification_expression</code> specifies a condition when a given object has to be designed and verified. This value of the <code>condition_type</code> 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 'configuration_type' of <code>Configuration</code> shall be evaluated.
---------------	--

'identification':	The Specification_expression specifies a condition that enables to distinguish the associated Product_class from other Product_class objects. This value is not applicable for a top level node in a hierarchy of Product_class objects. This identification is part of the identification of all sub classes of this Product_class.
'part usage':	The Specification_expression specifies a condition for the usage of the components of an Alternative_solution, the usage of an Item_instance or for the application of a Process_plan or a Process_operation_occurrence in the products of the associated Product_class. In this case, the Class_condition_association shall be referenced by at least one Configuration object.
'validity':	The Specification_expression specifies a condition used to verify a Product_specification for the associated Product_class. That means that the Specification_expression 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 Product_class. It is valid for all products belonging to the 'associated_product_class' in case of the condition types 'identification' and 'validity.'

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Class\_condition\_association.

### **Associations**

- associated\_condition : Specification\_expression [1]  
The associated\_condition specifies the Specification\_expression that is assigned to the Product\_class.

#### **8.10.1.4 Class Class\_inclusion\_association**

A Class\_inclusion\_association is the assignment of a Specification\_inclusion to a Product\_class. This assignment contains the information that a particular Specification\_inclusion applies for all products of that Product\_class.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Class\_inclusion\_association.

### **Association**

- associated\_inclusion : Specification\_inclusion [1]  
The associated\_inclusion specifies the Specification\_inclusion that is associated with the Product\_class.



### 8.10.1.5 Class `Class_specification_association`

A `Class_specification_association` is an association of a `Specification` with a `Product_class`. This `Specification` serves as a potential characteristic of all products belonging to the `Product_class`.

#### **Base Class**

- `PLM_object` (ABS)

#### **Attributes**

- `association_type` : `String` [1]  
The `association_type` specifies the kind of availability of a particular `Specification` in a `Product_class`.

#### **Compositions**

- `none`

#### **Associations**

- `associated_specification` : `Specification` [1]  
The `associated_specification` specifies the `Specification` that is associated with the `Product_class`.

### 8.10.1.6 Class `Class_structure_relationship`

A `Class_structure_relationship` is an association between a `Product_class` object and either a `Product_component` or a `Product_function` object.

#### **Base Class**

- `PLM_object` (ABS)

#### **Attributes**

- `relation_type` : `String` [1]  
The `relation_type` specifies the meaning of the relationship. Where applicable the following values shall be used:

'functionality'	The related <code>Product_function</code> is an element of the functional structure of the relating <code>Product_class</code> . This relation type shall only be used if the related object is a <code>Product_function</code> .
'realization'	The related <code>Product_component</code> fulfils, partially or fully, the requirements identified with the relating <code>Product_class</code> . This relation type shall only be used if the related object is a <code>Product_component</code> .

#### **Compositions**

- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Class_structure_relationship`.

#### **Associations**

- `related` : `Product_function_component_select` [1]  
The `related` specifies the `Product_component` or `Product_function` object related by the `Class_structure_relationship`.

### 8.10.1.7 Class Complex\_product (ABS)

A Complex\_product is an object with the capability that it can be realized by, decomposed into or specialized as Product\_constituent objects in a functional, logical, or physical way.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [1]  
The id specifies the identifier of the Complex\_product.
- version\_id : String [0..1]  
The version\_id identifies a version of the concept represented by a Complex\_product.

#### **Compositions**

- product\_structure\_relationship : Product\_structure\_relationship [0..\*]  
The product\_structure\_relationship specifies the product\_structure\_relationship where this Complex\_product is decomposed functionally, logically, or physically into or realized by the related Product\_constituent.
- design\_constraint\_association : Design\_constraint\_association [0..\*]  
The design\_constraint\_association specifies the design\_constraint\_association so that the Design\_constraint.affects this object.
- complex\_product\_relationship : Complex\_product\_relationship [0..\*]  
The complex\_product\_relationship specifies the complex\_product\_relationship that relates the first of the two Complex\_product objects.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Complex\_product.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Complex\_product.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

#### **Associations**

- none

### 8.10.1.8 Class Complex\_product\_relationship

A Complex\_product\_relationship is a relationship between two Complex\_product objects.

#### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- `relation_type` : String [1]  
The `relation_type` specifies the meaning of the relationship. Where applicable the following values shall be used:

'derivation'	The <code>Complex_product_relationship</code> defines a relationship where the related <code>Complex_product</code> is derived from the relating <code>Complex_product</code> .
'replacement'	The <code>Complex_product_relationship</code> defines a relationship where the related <code>Complex_product</code> is used in place of the relating <code>Complex_product</code> .
'version hierarchy'	The <code>Complex_product_relationship</code> defines a relationship where the related <code>Complex_product</code> is a sub version of the relating <code>Complex_product</code> .
'version sequence'	The <code>Complex_product_relationship</code> defines a relationship where the relating <code>Complex_product</code> is the preceding version and the related <code>Complex_product</code> is the following version.

### **Compositions**

- `description` : String\_select [0..1]  
The description specifies additional information about the `Complex_product_relationship`.

### **Associations**

- `related` : `Complex_product` (ABS) [1]  
The `related` specifies the second of the two objects related by the `Complex_product_relationship`.

### **8.10.1.9 Class Component\_placement**

A `Component_placement` is the information pertaining to the placement of a `Product_component`, which is defined in its own `Cartesian_coordinate_space`, in the coordinate space of a reference `Product_component`.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- `reference_product_component` : `Product_component` [1]  
The `reference_product_component` specifies the high level `Product_component` that is defined in the reference coordinate space. A `Model_property_association` shall be assigned to the `reference_product_component` to define this reference coordinate space.

- placement : Transformation\_select [1]  
The placement specifies the Geometric\_model\_relationship\_with\_transformation or the Template\_instance that defines the position of the 'placed\_component' relatively to the 'reference\_product\_component.' In the case of Template\_instance, the scale shall be omitted or set to 1.0.

### 8.10.1.10 Class Configuration

A Configuration is the association of a Class\_condition\_association or a Class\_specification\_association object with a design or with a process in order to define a valid usage of it in the context of a certain Product\_class.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- configuration\_type : String [1]  
The configuration\_type specifies the valid usage of a Configuration object that is applied to the application object as configured\_element. The following values shall be used:

'design'	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 Class_condition_association and Class_specification_association objects referenced as the 'is_solution_for.'
----------	--

'usage'	The object referenced as the 'configured_element' is controlled by a Configuration. The Class_condition_association and Class_specification_association objects specify the usage cases and are referenced as the 'is_solution_for.'
---------	--

- inheritance\_type : String [1]  
The inheritance\_type specifies whether or not an inheritance scheme for the configuration information in a hierarchical structure is applied to the application object referenced as the configured\_element. The levels within such a hierarchy are defined through Product\_structure\_relationship objects or the attribute 'base\_element' of Alternative\_solution. The following values shall be used:

'exception'	No inheritance scheme is applicable and all required configuration information must be attached locally at the application 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 configured_element and using the logical OR to combine alternatives. In addition, this evaluation shall consider related effectivity information. 'inherited' only applies for objects for which the same value of 'configuration_type' is defined.
'local'	No inheritance scheme is applicable and all required configuration information must be attached locally at the application object. Nevertheless, any potentially inherited configuration information of a higher level shall be consistent, i.e., be a subset of the locally defined configuration information.

### **Compositions**

- none

### **Associations**

- is\_solution\_for : Configured\_specification\_select [1]  
The is\_solution\_for specifies the characteristic or combination of characteristics for which the object referenced as the configured\_element provides a solution or which is needed to control a process operation. These characteristics are defined by a Class\_specification\_association and combinations of characteristics are defined by a Class\_condition\_association where the attribute 'condition type' is 'part usage.'

#### **8.10.1.11 Class Dated\_configuration**

A Dated\_configuration is a Manufacturing\_configuration that applies onwards from a given date, or between a start and an end date.

### **Base Class**

- Manufacturing\_configuration (ABS)

### **Attributes**

- start\_date : Event\_or\_date\_select [1]  
The start\_date specifies the first date when the Dated\_configuration is valid.
- end\_date : Event\_or\_date\_select [0..1]  
The end\_date specifies the date and time when the validity of the 'configured\_element' is not defined any longer.

### **Compositions**

- none

### **Associations**

- none

### 8.10.1.12 Class Descriptive\_specification

A Descriptive\_specification is a textual description of an object.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [0..1]  
The id specifies the identifier of the Descriptive\_specification.

#### **Compositions**

- description : String\_select [1]  
The description specifies the Descriptive\_specification.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Descriptive\_specification.

#### **Associations**

- none

### 8.10.1.13 Class Design\_constraint

A Design\_constraint is a requirement that has to be considered in the design process of a Complex\_product. This constraint may be geometry based.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- constraint\_id : String [1]  
The constraint\_id specifies the identifier of the Design\_constraint.

#### **Compositions**

- design\_constraint\_relationship : Design\_constraint\_relationship [0..\*]  
The design\_constraint\_relationship specifies the design\_constraint\_relationship that relates the first of the two Design\_constraint objects.
- description : String\_select [0..1]  
The description specifies additional information about the Design\_constraint.
- name : String\_select [0..1]  
The name specifies the word or group of words by which the Design\_constraint is referred to.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Design\_constraint.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- `is_valid_for` : `Product_class` [0..\*]  
The `is_valid_for` specifies the set of `Product_class` objects that are affected by the `Design_constraint`.

#### **8.10.1.14 Class `Design_constraint_association`**

A `Design_constraint_association` is a mechanism to associate a `Design_constraint` with an object that is subject to the constraint indicated.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- none

### **Compositions**

- `name` : `String_select` [0..1]  
The `name` specifies the word or group of words by which the `Design_constraint_association` is referred to.

### **Associations**

- `is_based_on` : `Design_constraint` [1]  
The `is_based_on` specifies the `Design_constraint` that represents the constraint.

#### **8.10.1.15 Class `Design_constraint_relationship`**

A `Design_constraint_relationship` is a relationship between two `Design_constraint` objects.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `relation_type` : `String` [1]  
The `relation_type` specifies the meaning of the relationship.

### **Compositions**

- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Design_constraint_relationship`.

### **Associations**

- `related` : `Design_constraint` [1]  
The `related` specifies the second of the two `Design_constraint` objects related by the `Design_constraint_relationship`.

#### **8.10.1.16 Class `Design_constraint_version`**

A `Design_constraint_version` is a particular version of a `Design_constraint`.

### **Base Class**

- Design\_constraint

### **Attributes**

- version\_id : String [1]  
The version\_id specifies the identification of a particular version of a Design\_constraint. The version\_id shall be unique within the scope of a Design\_constraint.

### **Compositions**

- none

### **Associations**

- none

## **8.10.1.17 Class Effectivity**

An Effectivity is the identification of the valid use of an aspect of product data tracked by date or event.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- id : String [0..1]  
The id specifies the identifier of the Effectivity.
- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of the Effectivity.
- effectivity\_context : String [0..1]  
The effectivity\_context specifies the life cycle stage for which the Effectivity is valid.

### **Compositions**

- effectivity\_assignment : Effectivity\_assignment [0..\*]  
The effectivity\_assignment specifies the effectivity\_assignment that this Effectivity is assigned to.
- description : String\_select [0..1]  
The description specifies additional information about the Effectivity.

### **Associations**

- end\_definition : Event\_or\_date\_select [0..1]  
The end\_definition specifies the end of the period. The bound specified by the end\_definition is excluded from the interval of effectivity.
- start\_definition : Event\_or\_date\_select [0..1]  
The start\_definition specifies the start of the period. The bound specified by the start\_definition is included in the interval of effectivity.



- **period** : Duration [0..1]  
The period specifies the period of time in which the Effectivity is defined, either starting at the point in time specified by 'start\_definition' or ending at the point in time specified by 'end\_definition' (period shall be specified with a positive value).
- **concerned\_organization** : Organization [0..\*]  
The concerned\_organization specifies the set of Organization objects in which the Effectivity is valid.

### 8.10.1.18 Class Effectivity\_assignment

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- **role** : String [1]  
The role specifies the relationship between the Effectivity and the object that has an effectivity assigned to it. Where applicable the following values shall be used:
 

'actual'	The actual period during which the Effectivity lasted.
'planned'	The period associated with the Effectivity defines a planned period of time during which the associated object is or was supposed to be effective.
'required'	The associated object must be kept effective for this period.
- **effectivity\_indication** : Boolean [1]  
The effectivity\_indication specifies whether the assigned\_effectivity defines a period of effectivity (value equal 'TRUE') or a period of ineffectivity (value equal 'FALSE') for the effective\_element. In the first case, use of the effective\_element is or was valid during the considered period.

#### **Compositions**

- none

#### **Associations**

- **effective\_element** : Effective\_element\_select [1..\*]  
The effective\_elements specify the objects that have an Effectivity assigned to it.

### 8.10.1.19 Class Effectivity\_relationship

An Effectivity\_relationship is a relationship between two Effectivity objects.

NOTE: Sometimes the effectivity is not dependent on particular dates but on the effectivity of other items. In this case the dates are not instantiated and there is an Effectivity\_relationship to the reference Effectivity.

#### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- relation\_type: string[1]  
The relation\_type specifies the meaning of the relationship.

### **Compositions**

- description: String\_select[0..1]  
The description specifies additional information about the Effectivity\_relationship.

### **Associations**

- related: Effectivity[1]  
The related specifies the second of the two Effectivity objects related by the Effectivity\_relationship.  
NOTE: The semantics of this attribute are defined by the attribute 'relation\_type.'

### **8.10.1.20 Class Final\_solution**

A Final\_solution is the specification of a set of additional sensual characteristics that can be applied to an Item\_instance that represents a neutral part in order to finalize its definition.

#### **Base Class**

- Alternative\_solution

#### **Attributes**

- final\_status : String [1]  
The final\_status specifies the level of completion between the neutral part and the final part.

#### **Compositions**

- none

#### **Associations**

- final\_specification : Final\_definition\_select [1..\*]  
The final\_specification specifies the means of finalization that is applied to the neutral part and which may be objects of type Descriptive\_specification, Physical\_instance, or Design\_discipline\_item\_definition.

### **8.10.1.21 Class Instance\_placement**

An Instance\_placement is the information pertaining to the placement of a Single\_instance, which is defined in its own Cartesian\_coordinate\_space, in the coordinate space of a reference Product\_component.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

### **Associations**

- reference\_product\_component : Product\_component [1]  
The reference\_product\_component specifies the Product\_component that specifies indirectly the reference coordinate space. A Model\_property\_association shall be assigned to the reference\_product\_component to define this reference coordinate space.
- placement : Transformation\_select [1]  
The placement specifies the Geometric\_model\_relationship\_with\_transformation or the Template\_instance that defines the position of the 'placed\_instance' relatively to the 'reference\_product\_component.' In the case of Template\_instance, the scale shall be omitted or set to 1.0.

#### **8.10.1.22 Class Item\_function\_association**

An Item\_function\_association is a mechanism to relate a Product\_function and a Design\_discipline\_item\_definition.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- association\_type : String [1]  
The association\_type specifies the kind of association.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Item\_function\_association.

### **Associations**

- associated\_function : Product\_function [1]  
The associated\_function specifies the associated Product\_function.

#### **8.10.1.23 Class Lot\_configuration**

A Lot\_configuration is a Manufacturing\_configuration that applies to a given production batch of the product that is related with the object referred to as 'is\_solution\_for.'

### **Base Class**

- Manufacturing\_configuration (ABS)

### **Attributes**

- lot\_id : String [1]  
The lot\_id specifies the identification of the batch for which the Lot\_configuration applies.
- lot\_size : Value\_with\_unit (ABS) [1]  
The lot\_size specifies the size of the batch for which the Lot\_configuration applies.

### **Compositions**

- none

### **Associations**

- none

#### **8.10.1.24 Class Manufacturing\_configuration (ABS)**

A Manufacturing\_configuration is the association of a Product\_design with an Item\_instance.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- concerned\_organization : Organization [0..\*]  
The concerned\_organization specifies the Organization in which the Manufacturing\_configuration is valid. The case where the concerned\_organization is an empty set means that the Manufacturing\_configuration regards any organization that may consider the 'configured\_element.'
- is\_solution\_for : Product\_design [1]  
The is\_solution\_for specifies the design for which an Item\_instance is configured.

#### **8.10.1.25 Class Physical\_instance**

A Physical\_instance is the denomination of a physically realized object. A Physical\_instance may be identified by a serial number. A lot id may be provided additionally to the serial number.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- serial\_number : String [0..1]  
The serial\_number is an identifier that distinguishes one Physical\_instance from another.
- lot\_id : String [0..1]  
The lot\_id specifies the identifier of the lot the Physical\_instance is part of.
- inventory\_number : String [0..1]  
The inventory\_number specifies an alphanumerical string to identify an item in the detailed list of articles, such as goods and chattels, found in the possession of a person or enterprise.

### **Compositions**

- physical\_instance\_test\_result : Physical\_instance\_test\_result [0..\*]  
The physical\_instance\_test\_result specifies the physical\_instance\_test\_result for which this Physical\_instance was the subject of the test activity.

- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Physical_instance`.
- `physical_assembly_relationship` : `Physical_assembly_relationship` [0..\*]  
The `physical_assembly_relationship` specifies the `physical_assembly_relationship` for which this `Physical_instance` serves as the assembly in the physical structure.
- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Physical_instance`.
- `alias_identification` : `Alias_identification` [0..\*]  
The `Alias_identification` specifies the `Alias_identification` that is applied to this `Physical_instance`.
- `simple_property_value` : `Simple_property_value (ABS)` [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- `is_realization_of` : `Physical_instance_definition_select` [0..1]  
The `is_realization_of` specifies the `Product_identification` or the `Design_discipline_item_definition` that collects the information defining the `Physical_instance`.

### **8.10.1.26 Class Physical\_instance\_test\_result**

A `Physical_instance_test_result` is a mechanism to associate a `Physical_instance` with measurements made on this `Physical_instance`.

### **Base Class**

- `PLM_object (ABS)`

### **Attributes**

- `id` : `String` [1]  
The `id` specifies the identifier of the `Physical_instance_test_result`.

### **Compositions**

- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Physical_instance_test_result`.
- `Document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Physical_instance_test_result`.

### **Associations**

- `test_result` : `Property_value_representation` [0..\*]  
The `test_result` specifies the characteristics that were determined by the performed test.
- `test_activity` : `Test_activity_select` [0..1]  
The `test_activity` specifies the `Activity` or the `Process_operation_occurrence` that has led to the test result.

### **8.10.1.27 Class Product\_class**

A `Product_class` is the identification of a set of similar products to be offered to the market. `Product_class` objects that are related to each other by a `Product_class_relationship` do not inherit any characteristics from each other.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- id : String [1]  
The id specifies the identifier of the Product\_class that shall be unique.
- level\_type : String [0..1]  
The level\_type specifies the level or category of this Product\_class in a hierarchical structure of Product\_class objects. The level\_type shall only be used if and only if the level\_type is specified in the context of the unit of functionality 'specification\_control' (UoF S7).
- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of a Product\_class.

### **Compositions**

- product\_identification : Product\_identification [0..\*]  
The product\_identification specifies the product\_identification of the product that belongs to this Product\_class.
- description : String\_select [0..1]  
The description specifies additional information about the Product\_class.
- name : String\_select [0..1]  
The name specifies the word or group of words by which the Product\_class is referred to.
- class\_structure\_relationship : Class\_structure\_relationship [0..\*]  
The class\_structure\_relationship specifies the class\_structure\_relationship that relates this Product\_class.
- class\_specification\_association : Class\_specification\_association [0..\*]  
The class\_specification\_association specifies the class\_specification\_association that is valid for this Product\_class.
- class\_inclusion\_association : Class\_inclusion\_association [0..\*]  
The class\_inclusion\_association specifies the class\_inclusion\_association that is valid for this Product\_class.
- class\_condition\_association : Class\_condition\_association [0..\*]  
The class\_condition\_association specifies the class\_condition\_association that is valid for this Product\_class.
- class\_category\_association : Class\_category\_association [0..\*]  
The class\_category\_association specifies the class\_category\_association that is valid for this Product\_class.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Product\_class.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Product\_class.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- none

#### **8.10.1.28 Class Product\_class\_relationship**

A Product\_class\_relationship is a relationship between two Product\_class objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- relation\_type : string[1]  
The relation\_type specifies the meaning of the relationship.  
NOTE: The relationship does not imply inheritance of any kind between the application objects that are related.

### **Compositions**

- description : String\_select[0..1]  
The description specifies additional information about the Product\_class\_relationship.

### **Associations**

- related : Product\_class[1]  
The related specifies the second of the two Product\_class objects related by the Product\_class\_relationship.  
NOTE: The semantics of this attribute are defined by the attribute relation\_type.

## **8.10.1.29 Class Product\_component**

A Product\_component is an element in a conceptual product structure.

### **Base Class**

- Complex\_product (ABS)

### **Attributes**

- instance\_required : Boolean [1]  
The instance\_required specifies if the existence of a corresponding Item\_instance is required for the various Alternative\_solution objects of that Product\_component. A value of 'true' indicates that a corresponding Item\_instance is required.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Product\_component.
- name : String\_select [0..1]  
The name specifies the word or group of words by which the Product\_component is referred to.
- configuration : Configuration [0..\*]  
The configuration specifies the configuration that controls this Product\_component for its valid usage.
- component\_placement : Component\_placement [0..\*]  
The component\_placement specifies the component\_placement that is positioned with respect to this Product\_component.

### **Associations**

- is\_relevant\_for : Application\_context [0..\*]  
The is\_relevant\_for specifies the Application\_context objects in which the Product\_component has to be considered.

- `is_influenced_by` : `Class_category_association` [0..\*]  
The `is_influenced_by` specifies the `Specification_category` objects that impact the design of a solution for the `Product_component` in the context of the `Product_class` objects that are referred to by the `Class_category_association` objects.

### 8.10.1.30 Class `Product_design`

A `Product_design` is a mechanism to associate an `Item_version` with its corresponding `Product_identification`.

#### **Base Class**

- `PLM_object` (ABS)

#### **Attributes**

- none

#### **Compositions**

- none

#### **Associations**

- `product` : `Product_identification` [1]  
The `product` specifies the `Product_identification` that represents the requirements.

### 8.10.1.31 Class `Product_function`

A `Product_function` is a behavior or an action expected from a product.

#### **Base Class**

- `Complex_product` (ABS)

#### **Attributes**

- none

#### **Compositions**

- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Product_function`.
- `name` : `String_select` [0..1]  
The `name` specifies the word or group of words by which the `Product_function` is referred to.
- `configuration` : `Configuration` [0..\*]  
The `configuration` specifies the configuration that controls this `Product_function` for its valid usage.

#### **Associations**

- `is_relevant_for` : `Application_context` [0..\*]  
The `is_relevant_for` specifies the `Application_context` objects in which the `Product_function` has to be considered.



### 8.10.1.32 Class Product\_identification

A Product\_identification identifies a manufacturable object, or expected as so. A Product\_identification is defined with respect to the Product\_class it is a member of.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of a Product\_identification.
- id : String [1]  
The id specifies the identifier of the Product\_identification.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Product\_identification.
- name : String\_select [0..1]  
The name specifies the word or group of words by which the Product\_identification is referred to.
- item\_instance : Item\_instance (ABS) [0..\*]  
The item\_instance specifies the item\_instance for which this Product\_identification serves as a definition.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Product\_identification.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

#### **Associations**

- none

### 8.10.1.33 Class Product\_specification

A Product\_specification is a Product\_identification for which one or more additional Specification objects enhance the characterization provided for the associated Product\_class.

#### **Base Class**

- Product\_identification

#### **Attributes**

- none

#### **Compositions**

- none

### **Associations**

- `defining_specification` : `Specification` [1..\*]  
The `defining_specification` specifies the set of `Specification` objects necessary to discriminate the `Product_specification` within its `Product_class`.

#### **8.10.1.34 Class `Product_structure_relationship`**

A `Product_structure_relationship` is an association between a `Complex_product` and a `Product_constituent`, in which the `Product_constituent` is a functional, logical, or physical component or a realization of the `Complex_product`.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `relation_type` : `String` [1]  
The `relation_type` specifies the meaning of the relationship. Where applicable the following values shall be used:

'decomposition'	The related <code>Product_constituent</code> is one of potentially more components of the relating <code>Complex_product</code> . This relation type shall only be used for <code>Complex_product</code> and <code>Product_constituent</code> of the same type.
'functionality'	The related <code>Product_constituent</code> is an element of the functional structure of the relating <code>Complex_product</code> . This relation type shall only be used with a <code>Complex_product</code> of type <code>Alternative_solution</code> or <code>Product_component</code> and with a <code>Product_constituent</code> of type <code>Product_function</code> .
'occurrence'	The related <code>Product_constituent</code> is an occurrence defined by the relating <code>Complex_product</code> . This relation type shall only be used if related <code>Product_constituent</code> is of type <code>Product_component</code> .
'realization'	The related <code>Product_constituent</code> is a means for fulfilling, either partially or fully, the requirements identified with the relating <code>Complex_product</code> . This relation type shall be used only when the <code>Complex_product</code> and the <code>Product_constituent</code> are of different types.
'specialization'	The related <code>Product_constituent</code> fulfils the requirements of the relating <code>Complex_product</code> in a more specific way than defined for the relating <code>Complex_product</code> . This relation type shall only be used for <code>Product_constituent</code> and <code>Complex_product</code> of the same type.

### **Compositions**

- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Product_structure_relationship`.
- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Product_structure_relationship`.
- `simple_property_value` : `Simple_property_value` (ABS) [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- related : Product\_constituent\_select [1]  
The related specifies the Product\_constituent that is a functional, logical, or physical component or a realization of the relating Complex\_product.

#### **8.10.1.35 Class Retention\_period**

A Retention\_period is the definition of a period of time that product data needs to be maintained due to organizational policy or legal requirements.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- none

### **Compositions**

- retention\_purpose : String\_select[0..1]  
The retention\_purpose specifies the rationale behind the Retention\_period.

### **Associations**

- start\_definition : Event\_or\_date\_select[1]  
The start\_definition specifies the point in time at which the Retention\_period starts.
- earliest\_end\_definition : Period\_or\_date\_select[[1]  
The earliest\_end\_definition specifies the point in time after which any object the Retention\_period is applied to may be deleted or destroyed. In this context deletion or destruction applies to all subordinate objects that are not referenced by other objects.
- is\_applied\_to: General\_organizational\_data\_select[1]  
The is\_applied\_to specifies the objects whose existence is controlled by the Retention\_period.  
NOTE: The master document is the one for which earliest\_end\_definition and latest\_end\_definition are the same.
- latest\_end\_definition: Period\_or\_date\_select[1]  
The latest\_end\_definition specifies the point in time before which any objects that the Retention\_period is applied to shall be deleted or destroyed. In this context deletion or destruction applies to all subordinate objects that are not used by other objects.

#### **8.10.1.36 Class Serial\_configuration**

A Serial\_configuration is a Manufacturing\_configuration that applies onwards from a given serial number of the product that is considered within the object referred to as 'is\_solution\_for.'

### **Base Class**

- Manufacturing\_configuration (ABS)

### **Attributes**

- serial\_start\_number : String [1]  
The serial\_start\_number specifies the serial number of that instance of the product that is the first instance for which the Serial\_configuration applies.
- serial\_end\_number : String [0..1]  
The serial\_end\_number specifies the serial number of that instance of the product that is the last instance for which the Serial\_configuration applies.

### **Compositions**

- none

### **Associations**

- none

### **8.10.1.37 Class Specification**

A Specification is a characteristic of a product. A Specification discriminates one product from other members of the same Product\_class. A Specification refers to a Specification\_category that completes the semantics of the Specification.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- id : String [1]  
The id specifies the identifier of the Specification that shall be unique within the scope of a Specification\_category.
- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of a Specification.
- package : Boolean [1]  
The 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 Specification\_inclusion per Product\_class considered that refers to this Specification as 'if\_condition.' The Specification objects that are members of the package shall be specified as included\_specification.

### **Compositions**

- specification\_inclusion : Specification\_inclusion [0..\*]  
The specification\_inclusion specifies the specification\_inclusion for which this Specification serves as the condition for the inclusion.
- description : String\_select [0..1]  
The description specifies additional information about the Specification.
- name : String\_select [0..1]  
The name specifies the word or group of words by which the Specification is referred to.
- alias\_identification : Alias\_identification [0..\*]  
The Alias\_identification specifies the Alias\_identification that is applied to this Specification.

- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Specification`.

### **Associations**

- `category` : `Specification_category` [1]  
The `category` specifies the `Specification_category` that completes the semantics of the `Specification`.

### **8.10.1.38 Class Specification\_category**

A `Specification_category` is the definition of a set of `Specification` objects serving the same purpose.

#### **Base Class**

- `PLM_root_object` (ABS)

#### **Attributes**

- `implicit_exclusive_condition` : `Boolean` [1]  
The `implicit_exclusive_condition` specifies whether the `Specification` objects within the `Specification_category` 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.
- `id` : `String` [1]  
The `id` specifies the identifier of the `Specification_category` that shall be unique.

#### **Compositions**

- `specification_category_hierarchy` : `Specification_category_hierarchy` [0..\*]  
The `specification_category_hierarchy` specifies the `specification_category_hierarchy` for which this `Specification_category` is the higher level.
- `description` : `String_select` [1]  
The `description` specifies information about the `Specification_category`.
- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Specification_category`.
- `alias_identification` : `Alias_identification` [0..\*]  
The `Alias_identification` specifies the `Alias_identification` that is applied to this `Specification_category`.

#### **Associations**

- none

### **8.10.1.39 Class Specification\_category\_hierarchy**

A `Specification_category_hierarchy` is used to build up hierarchical structures of `Specification_category` objects.

#### **Base Class**

- `PLM_object` (ABS)

#### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- sub\_category : Specification\_category [1]  
The sub\_category is the lower level of Specification\_category in Specification\_category\_hierarchy.

#### **8.10.1.40 Class Specification\_expression**

A Specification\_expression is a combination of Specification objects formed by Boolean operations.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- operation : String [1]  
The operation specifies the kind of Boolean operation. Four kinds of operations are permitted:

'and'	All of the identified Specification objects shall be used.
'or':	A subset or all of the identified Specification objects shall be used.
'oneof'	Exactly one of the identified Specification objects shall be used.
'not'	The identified Specification shall not be used.

- id : String [0..1]  
The id specifies the identifier of the Specification\_expression.

### **Compositions**

- specification\_inclusion : Specification\_inclusion [0..\*]  
The specification\_inclusion specifies the specification\_inclusion for which this Specification\_expression serves as the condition for the inclusion.
- description : String\_select [0..1]  
The description specifies additional information about the Specification\_expression.

### **Associations**

- operand : Specification\_operand\_select [1..\*]  
The operand specifies the operands of the Boolean operation that are either Specification objects or other Specification\_expression objects.

#### **8.10.1.41 Class Specification\_inclusion**

A Specification\_inclusion is the representation of the statement that specifies that the application of a Specification or of a Specification\_expression implies the inclusion of an additional Specification or Specification\_expression.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- id : String [0..1]  
The id specifies the identifier of the Specification\_inclusion.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Specification\_inclusion.

### **Associations**

- included\_specification : Specification\_operand\_select [1]  
The included\_specification specifies the Specification or the Specification\_expression objects that are to be included. The included\_specification shall not reference a Specification\_expression with an operation of type 'or' or 'oneof,' except for negating expressions, i.e., as participants in an expression preceded by a 'not' operator. Expressions of operator 'not' shall not be nested within each other.

### **8.10.1.42 Class Supplier\_solution**

A Supplier\_solution is an alternative solution provided by a particular supplier.

#### **Base Class**

- Alternative\_solution

#### **Attributes**

- probability\_rate : String [0..1]  
The probability\_rate specifies the share that is assigned to the supplier in the context of the base element.

#### **Compositions**

- none

#### **Associations**

- supplier : Organization [1]  
The supplier specifies the Organization that acts as supplier for the Supplier\_solution.

### **8.10.1.43 Class Technical\_solution**

A Technical\_solution is an alternative solution where the functional requirements are fulfilled in a certain technical way.

#### **Base Class**

- Alternative\_solution

#### **Attributes**

- none

#### **Compositions**

- description : String\_select [1]  
The description specifies additional information about the Technical\_solution.

## ***Associations***

- none

## **8.10.2 Interfaces**

### **8.10.2.1 Interface Complex\_product\_select**

This empty interface is realized by the following classes:

- Product\_function
- Product\_component
- Alternative\_solution

### **8.10.2.2 Interface Configured\_specification\_select**

This empty interface is realized by the following classes:

- Class\_specification\_association
- Class\_condition\_association

### **8.10.2.3 Interface Configured\_item\_select**

## ***Compositions***

- configuration : Configuration [0..\*]

## ***Implemented By***

- Process\_operation\_occurrence
- Product\_function
- Product\_component
- Alternative\_solution
- Process\_plan
- Item\_instance

### **8.10.2.4 Interface Effective\_element\_select**

This empty interface is realized by the following classes:

- Classification\_system
- Specification\_inclusion
- Specification\_expression
- Specification\_category
- Specification
- Product\_structure\_relationship
- Product\_identification
- Product\_class



- Design\_constraint
- Configuration
- Security\_classification
- Complex\_product\_relationship
- Complex\_product (ABS)
- Class\_structure\_relationship
- Class\_specification\_association
- Class\_inclusion\_association
- Class\_condition\_association
- Class\_category\_association
- Document\_version
- Document\_representation (ABS)
- Document\_file (ABS)
- Document
- Item\_version
- Item\_definition\_relationship (ABS)
- Item
- Item\_instance\_relationship (ABS)
- Item\_instance (ABS)
- Item\_definition\_instance\_relationship (ABS)
- Assembly\_substitute\_relationship
- Process\_plan
- Process\_operation\_resource\_assignment
- Process\_operation\_occurrence\_relationship
- Process\_operation\_occurrence
- Process\_operation\_definition\_relationship
- Process\_operation\_definition
- Property\_value\_association (ABS)
- Simple\_property\_association
- Property (ABS)
- Material
- Geometric\_model

#### **8.10.2.5 Interface Final\_definition\_select**

This empty interface is realized by the following classes:

- Physical\_instance
- Descriptive\_specification
- Design\_discipline\_item\_definition

#### **8.10.2.6 Interface Instance\_definition\_select**

##### ***Compositions***

- item\_instance : Item\_instance [0..\*]

##### ***Implemented By***

- Product\_identification
- Design\_discipline\_item\_definition

#### **8.10.2.7 Interface Physical\_instance\_definition\_select**

This empty interface is realized by the following classes:

- Product\_identification
- Design\_discipline\_item\_definition

#### **8.10.2.8 Interface Product\_function\_component\_select**

This empty interface is realized by the following classes:

- Product\_function
- Product\_component

#### **8.10.2.9 Interface Specification\_operand\_select**

##### ***Compositions***

- specification\_inclusion : Specification\_inclusion [0..\*]

##### ***Implemented By***

- Specification
- Specification\_expression

#### **8.10.2.10 Interface Test\_activity\_select**

This empty interface is realized by the following classes:

- Activity
- Process\_operation\_occurrence

## 8.11 Package Change\_and\_work\_management

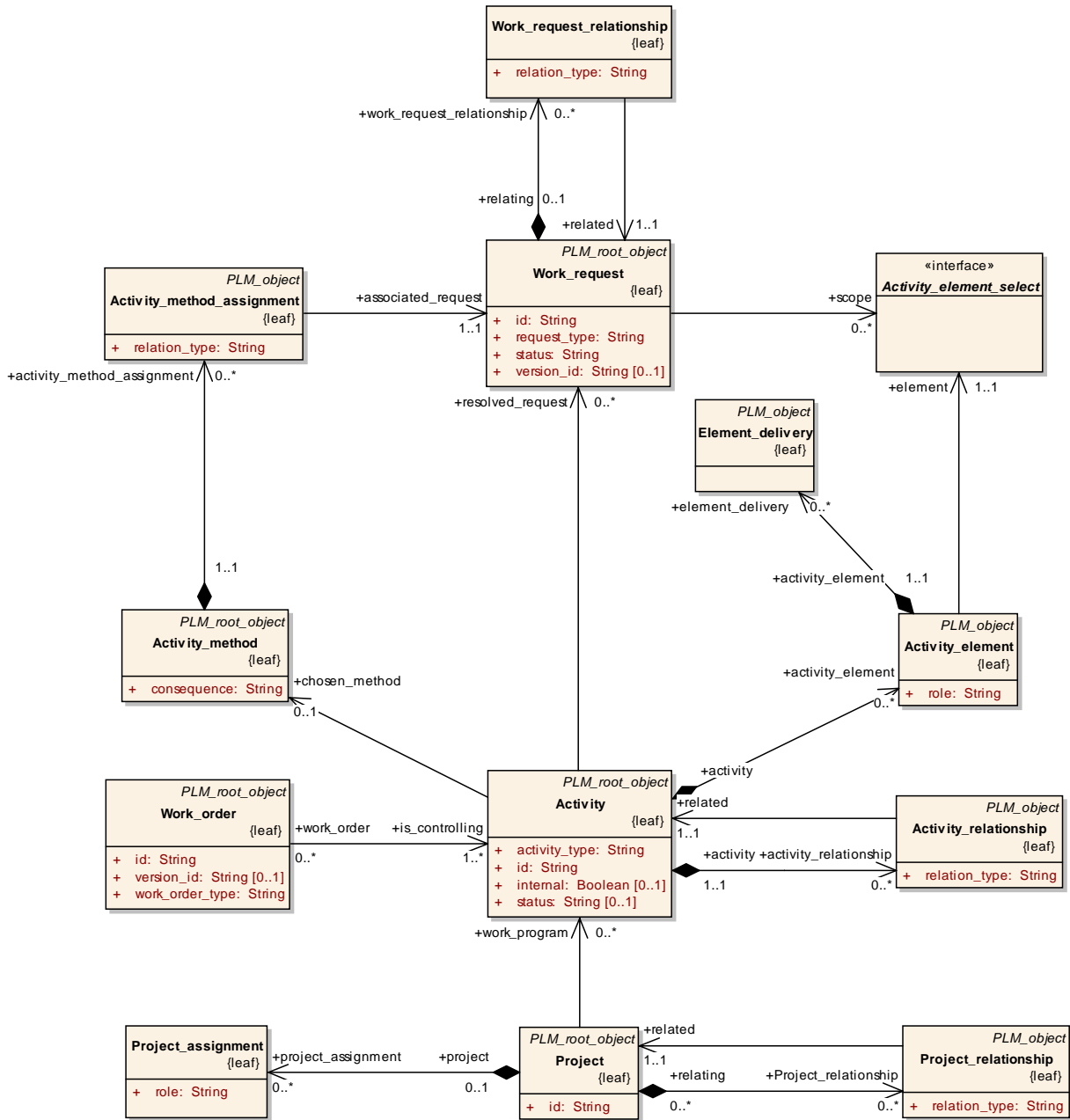


Figure 8.27 - Change management

### 8.11.1 Classes

#### 8.11.1.1 Class Activity

An Activity is the fact of achieving or accomplishing an action.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- activity\_type : String [1]

The activity\_type specifies the purpose of the Activity. Where applicable the following values shall be used:

'amendment'	An Activity to add information to product data.
'analysis'	An Activity to determine the behaviour of an element under certain physical circumstances.
'cancellation'	An Activity to delete an element from the bill of material or to cancel the whole bill of material.
'delivery change'	An Activity to change the delivery schedule of an element.
'design change'	An Activity to change the design of an item or an assembly; this might include changes to the geometry or to properties of the object.
'design'	An Activity concerning the development of a design of an item.
'mock-up creation'	An Activity to create an experimental model or replica of an item.
'prototype building'	An Activity to manufacture a preliminary version of an item.
'rectification'	An Activity to correct the data, documentation or structure associated with an item.
'restructuring'	An Activity to create a new structure or position within a bill of material without changing the data associated with the items in the bill of material.
'spare part creation'	An Activity to design a spare part or to classify an item as a spare part.
'stop notice':	An Activity to stop the manufacturing process of an item.
'testing'	An Activity to test an item.
'work definition'	An Activity to manage several sub-activities related to this Activity by an Activity_relationship with a 'relation_type' of value 'decomposition.'

- id : String [1]

The id specifies the identifier of the Activity.

- status : String [0..1]

The status specifies the level of completion of the Activity.

- internal : Boolean [0..1]

The internal specifies whether the activity is carried out within the organization that initiated the activity. A value of 'true' indicates that the activity is carried out within this particular organization.

### **Compositions**

- activity\_relationship : Activity\_relationship [0..\*]

The Activity\_relationship specifies the Activity\_relationship that relates the first of the two Activity objects.

- `activity_element` : `Activity_element` [0..\*]  
The `Activity_element` specifies the `Activity_element` that belongs to this `Activity`.
- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Activity`.
- `document_assignment` : `Document_assignment` [0..\*]  
The `document_assignment` specifies the object that provides information for this `Activity`.
- `simple_property_value` : `Simple_property_value (ABS)` [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- `chosen_method` : `Activity_method` [0..1]  
The `chosen_method` specifies the `Activity_method` used to carry out the `Activity`.
- `actual_start_date` : `Date_time` [0..1]  
The `actual_start_date` specifies the date when the `Activity` actually started.
- `planned_start_date` : `Event_or_date_select` [0..1]  
The `planned_start_date` specifies the date when the `Activity` is or was supposed to be started.
- `planned_end_date` : `Period_or_date_select` [0..1]  
The `planned_end_date` specifies the date when the `Activity` is or was supposed to be finished.
- `actual_end_date` : `Date_time` [0..1]  
The `actual_end_date` specifies the date when the `Activity` actually finished.
- `requestor` : `Date_and_person_organization` [0..1]  
The `requestor` specifies the `Person` or `Organization` that requested the `Activity` and the date the request was submitted.
- `supplying_organization` : `Organization` [0..\*]  
The `supplying_organization` specifies the set of `Organization` objects that carry out the work.
- `concerned_organization` : `Organization` [0..\*]  
The `concerned_organization` specifies the set of `Organization` objects that are affected by the result of the `Activity`.
- `resolved_request` : `Work_request` [0..\*]  
The `resolved_request` specifies the set of `Work_request` objects that are resolved by the `Activity`.

#### **8.11.1.2 Class `Activity_element`**

An `Activity_element` is an item of work that is part of an `Activity`.

#### **Base Class**

- `PLM_object (ABS)`

### **Attributes**

- role : String [1]  
The role specifies the function that is performed by the Activity\_element in the context of the concerned Activity. Where applicable the following values shall be used:

'control'                    The referenced element is an object that has immediate influence on the Activity performed.

'input'                      The referenced element serves as initial data for the Activity.

'output'                     The referenced element is a result of the Activity.

### **Compositions**

- element\_delivery : Element\_delivery [0..\*]  
The Element\_delivery specifies the Element\_delivery that this Activity\_element is subject to.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Activity\_element.

### **Associations**

- element : Activity\_element\_select [1]  
The element specifies the piece of product data that is under work.

#### **8.11.1.3 Class Activity\_method**

An Activity\_method is a procedure that may be used to solve a request.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- consequence : String [0..1]  
The consequence specifies the expected positive or negative effects of the application of a particular Activity\_method.

### **Compositions**

- activity\_method\_assignment : Activity\_method\_assignment [0..\*]  
The activity\_method\_assignment specifies the activity\_method\_assignment for which this activity\_method is recommended or shall not be chosen.
- name : String\_select [1]  
The name specifies the word or group of words by which the Activity\_method is referred to.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Activity\_method.

**Associations**

- description : String\_select [1]  
The description specifies additional information that defines the Activity\_method in terms of either the nature of the Activity\_method or in terms of the specific procedure steps required to implement it.

**8.11.1.4 Class Activity\_method\_assignment**

An Activity\_method\_assignment is an object that associates an Activity\_method with a Work\_request. The associated Activity\_method serves as a recommended or non-recommended method to resolve the tasks specified in the Work\_request.

**Base Class**

- PLM\_object (ABS)

**Attributes**

- relation\_type : String [1]  
The relation\_type specifies whether the specified Activity\_method may be used or not. Where applicable the following values shall be used:

'non recommended method'	The specified Activity_method shall not be used in order to accomplish the specified Work_request.
'recommended method'	The specified Activity_method may be used in order to accomplish the specified Work_request.

**Compositions**

- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

**Associations**

- associated\_request : Work\_request [1]  
The associated\_request identifies the Work\_request that the recommended or non-recommended method applies to.

**8.11.1.5 Class Activity\_relationship**

An Activity\_relationship is a relationship between two Activity objects.

**Base Class**

- PLM\_object (ABS)

**Attributes**

- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship. Where applicable the following values shall be used:

'alternative':	The application object defines a relationship where the related Activity may be used alternatively instead of the relating Activity.
'decomposition'	The application object defines a relationship where the related Activity is one of potentially more sub-activities into which the relating Activity is broken down.
'derivation'	The application object defines a relationship where the related Activity is derived from the relating Activity.
'exclusiveness'	The application object defines a relationship where the relating and the related Activity shall not have any overlap in time of execution.
'precedence'	The application object defines a relationship where the related Activity has higher priority than the relating Activity.
'sequence'	The application object defines a relationship where the relating Activity shall be completed before the related Activity starts.
'simultaneity'	The application object defines a relationship that establishes that both the relating and related Activity are considered as occurring during the same time period or shall be performed together in order to ensure consistency and enhance efficiency.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Activity\_relationship.

### **Associations**

- related : Activity [1]  
The related specifies the second of the two Activity objects related by an Activity\_relationship.

### **8.11.1.6 Class Change**

A Change is a mechanism to collect the Model\_change objects and the Property\_change objects that describe the differences between the two objects referenced by the specified relationship object.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- none

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Change.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Change.



### **Associations**

- none

#### **8.11.1.7 Class Element\_delivery**

An Element\_delivery is the specification of the expected delivery of an Activity\_element.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- quantity : Value\_with\_unit (ABS) [1]  
The quantity specifies the number of objects referred by the Activity\_element to be delivered.
- destination : Organization [1]  
The destination specifies the Organization the Activity\_element is to be delivered to.

#### **8.11.1.8 Class Project**

A Project is an identified program of work.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- id : String [1]  
The id specifies the identifier of the Project.

### **Compositions**

- Project\_relationship : Project\_relationship [0..\*]  
The Project\_relationship specifies the Project\_relationship that relates the first of the two Project objects.
- description : String\_select [0..1]  
The description specifies additional information about the Project.
- name : String\_select [1]  
The name specifies the word or group of words by which the Project is referred to.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Project.

### **Associations**

- `planned_end_date` : `Period_or_date_select` [0..1]  
The `planned_end_date` specifies either the date when the Project is or was supposed to be finished or the planned duration of the Project.
- `work_program` : `Activity` [0..\*]  
The `work_program` specifies the Activity objects that are carried out within the Project.
- `planned_start_date` : `Event_or_date_select` [0..1]  
The `planned_start_date` specifies the date when the Project is or was supposed to be started.
- `actual_end_date` : `Date_time` [0..1]  
The `actual_end_date` specifies the date when the Project was actually finished.
- `actual_start_date` : `Date_time` [0..1]  
The `actual_start_date` specifies the date when the Project was actually started.
- `is_applied_to` : `Project_information_select` [0..\*]  
The `is_applied_to` specifies the set of objects that the work carried out by a Project applies to.

#### **8.11.1.9 Class Project\_relationship**

A `Project_relationship` is a relationship between two Project objects.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `relation_type` : `String` [1]  
The `relation_type` specifies the meaning of the relationship. Where applicable the following values shall be used:

'decomposition'	The application object defines a relationship where the related Project is one of potentially more components into which the relating Project is broken down.
'dependency'	The related Project is dependent upon the relating Project.
'sequence'	The application object defines a relationship where the relating Project shall be completed before the related Project starts.
'succession'	The related Project is the successor of the relating Project.

### **Compositions**

- `description` : `String_select` [0..1]  
The `description` specifies additional information about the `Project_relationship`.

### **Associations**

- `related` : `Project` [1]  
The `related` specifies the second of the two Project objects related by a `Project_relationship`.

### 8.11.1.10 Class Work\_order

A Work\_order is the authorization for one or more Activity objects to be performed.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [1]  
The id specifies the identifier of the Work\_order.
- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of a Work\_order.
- work\_order\_type : String [1]  
The work\_order\_type specifies the kind of the Work\_order. Where applicable the following values shall be used:

'design deviation permit'	An authorization for a deviation from the approved design data.
'design release'	An authorization for the design of a product or of an item or to create a bill of material.
'management resolution'	An authorization by a committee, such as the board of directors, to design or change an item.
'manufacturing release'	An authorization for the manufacturing process of a product or of an item.
'production deviation permit'	An authorization for a deviation from the approved manufacturing process.

#### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Work\_order.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Work\_order.

#### **Associations**

- is\_controlling : Activity [1..\*]  
The is\_controlling specifies the Activity objects that are controlled by this particular Work\_order.

### 8.11.1.11 Class Work\_request

A Work\_request is the solicitation for some work to be done.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [1]  
The id specifies the identifier of the Work\_request.

- request\_type : String [1]  
The request\_type specifies the intention of the Work\_request. Where applicable the following values shall be used:

'change of standard'	A request to translate a change to a standard into action.
'cost reduction'	A request aimed at reducing the engineering and manufacturing costs of an item
'customer rejection'	A request resulting from a rejection by a customer.
'customer request'	A request for an activity that is necessary to solve the request of a customer.
'durability improvement'	A request aimed at extending the life time of an item.
'government regulation'	A request resulting from legal requirements.
'procurement alignment'	A request to adjust the manufacturing process of different items.
'production alignment'	A request to adjust the manufacturing process of different items.
'production relief':	A request aimed at achieving a simpler assembly and production process.
'production requirement'	A request for an activity that is necessary from a production point of view.
'quality improvement'	A request aimed at increasing the quality of an item.
'security reason'	A request for an activity that is necessary from a security point of view.
'standardization'	A request to unify variants of an item.
'supplier request'	A request for an activity necessary to solve the request of a supplier.
'technical improvement'	A request aimed at improving the technical aspects of an item.
'tool improvement'	A request aimed at increasing the useful life of a tool.

- status : String [1]  
The status specifies the stage of the Work\_request. Where applicable the following values shall be used:

'in work'	The request is being developed.
'issued'	The request has been completed and reviewed, and immediate action takes place.
'proposed'	The request has been completed and is awaiting review and authorization.
'resolved'	The request is resolved; the actions as defined by the request have been completed and no further work is required.

- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of a Work\_request.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Work\_request.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Work\_request.

## **Associations**

- notified\_person : Date\_and\_person\_organization [1..\*]  
The notified\_person specifies the personnel that shall be informed about the Work\_request and the date when the personnel or organization shall be informed.
- scope : Activity\_element\_select [0..\*]  
The scope specifies the objects that are subject to the Work\_request.
- requestor : Date\_and\_person\_organization [1]  
The requestor specifies the person or organization who issued the Work\_request and the date when this person or organization issued the Work\_request.

## **8.11.2 Interfaces**

### **8.11.2.1 Interface Activity\_element\_select**

This empty interface is realized by the following classes:

- Activity\_method
- Specification\_inclusion
- Specification\_expression
- Specification\_category
- Specification
- Product\_structure\_relationship
- Product\_identification
- Product\_class
- Physical\_instance
- Manufacturing\_configuration (ABS)
- Design\_constraint
- Configuration
- Complex\_product (ABS)
- Class\_structure\_relationship
- Class\_specification\_association
- Class\_inclusion\_association
- Class\_condition\_association
- Class\_category\_association
- Document\_version
- Document\_representation (ABS)
- Document\_file (ABS)
- Geometric\_model
- Document
- Item\_version
- Item\_definition\_relationship (ABS)

- Item
- Design\_discipline\_item\_definition
- Physical\_assembly\_relationship
- Item\_instance\_relationship (ABS)
- Item\_instance (ABS)
- Item\_definition\_instance\_relationship (ABS)
- Assembly\_substitute\_relationship
- Alternate\_item\_relationship
- General\_classification
- Project
- Process\_plan
- Process\_operation\_occurrence
- Process\_operation\_definition
- Property\_value\_association (ABS)
- Simple\_property\_association
- Property (ABS)
- Material
- Geometric\_model

#### **8.11.2.2 Interface Change\_relationship\_select**

##### ***Compositions***

- change : Change [0..\*]

##### ***Implemented By***

- Process\_operation\_occurrence\_relationship
- Process\_plan\_relationship
- Shape\_element\_relationship
- Design\_constraint\_relationship
- Replaced\_definition\_relationship
- Replaced\_usage\_relationship
- Complex\_product\_relationship
- Item\_version\_relationship

#### **8.11.2.3 Interface Project\_information\_select**

This empty interface is realized by the following Classes:

- Product\_identification
- Product\_class
- Physical\_instance

- Complex\_product (ABS)
- Document\_version
- Document
- Item\_version
- Item\_instance
- Item
- Activity

## 8.12 Package Process\_planning

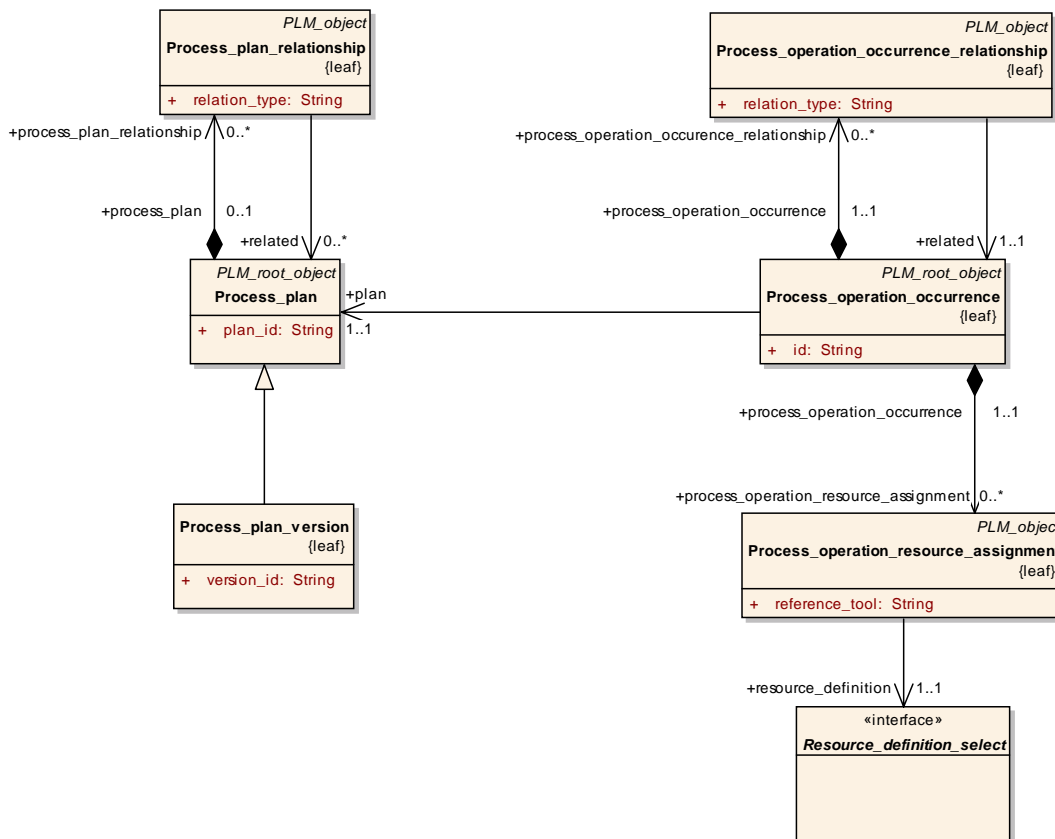


Figure 8.28 - Process planning

### 8.12.1 Classes

#### 8.12.1.1 Class Mated\_item\_association

A Mated\_item\_association is a relationship between a Mating\_definition and an Item\_instance used within the Mating\_definition. A Mated\_item\_association is a type of Item\_definition\_instance\_relationship.

### **Base Class**

- Item\_definition\_instance\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- mated\_item\_relationship : mated\_item\_relationship[0..\*]

### **Associations**

- placement : Transformation\_select[0..1]  
The placement specifies the Geometric\_model\_relationship\_with\_transformation or the Template\_instance that conveys the Transformation information. This Transformation is used to locate and to orient the constituent in the coordinate space of the Mating\_definition. In the case of a Template\_instance, the scale factor shall be omitted or set to 1.0.

## **8.12.1.2 Class Mated\_item\_relationship**

A Mated\_item\_relationship is a relationship between two Mated\_item\_association objects.

This relationship specifies additional information about the mating of two particular items that go into a Mating\_definition. The two Mated\_item\_association objects that are referenced by the Mated\_item\_relationship shall refer to the same Mating\_definition.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- mated\_shape : Shape\_element\_relationship[0..1]  
The mated\_shape specifies the Shape\_element\_relationship that relates the two Shape\_element objects that form the area of mating contact prior to mating.
- mating\_material : Quantified\_instance[0..1]  
The mating\_material specifies the set of Quantified\_instance objects used as material for the mating.
- related : Mated\_item\_association[1]  
The related specifies the second of the two Mated\_item\_association objects related by the Mated\_item\_relationship.



### 8.12.1.3 Class Mating\_definition

A Mating\_definition is a view of an Item\_version, defining the physical connection of two or more Item\_instance objects. It includes technical information about the kind of connection. This information is independent from the hierarchical assembly structure.

A Mating\_definition is a type of Design\_discipline\_item\_definition.

#### **Base Class**

- Design\_discipline\_item\_definition

#### **Attributes**

- mating\_type: string[1]  
The mating\_type specifies the kind of mating, i.e., how the items shall be mated together.

#### **Compositions**

- mated\_items: mated\_item\_association [2..\*]

#### **Associations**

- none

### 8.12.1.4 Class Process\_operation\_definition

A Process\_operation\_definition is the specification of an activity that may be included in a Process\_plan. A Process\_operation\_definition characterizes a manufacturing or control operation.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- id : String [1]  
The id specifies the identifier of the Process\_operation\_definition that shall be unique within the scope of the associated Process\_plan\_version.
- process\_type : String [1]  
The process\_type specifies the type of the Process\_operation\_definition.
- version\_id : String [0..1]  
The version\_id specifies the identification of a particular version of a Process\_operation\_definition.

#### **Compositions**

- process\_operation\_definition\_relationship : Process\_operation\_definition\_relationship [0..\*]  
The process\_operation\_definition\_relationship specifies the process\_operation\_definition\_relationship that relates the first of the two Process\_operation\_definition objects.
- description : String\_select [0..1]  
The description specifies additional information about the Process\_operation\_definition.

- `name` : `String_select` [0..1]  
The name specifies the word or group of words by which the `Process_operation_definition` is referred to.
- `simple_property_value` : `Simple_property_value` (ABS) [0..\*]  
The `simple_property_value` specifies the assigned simple property values.

### **Associations**

- none

#### **8.12.1.5 Class `Process_operation_definition_relationship`**

A `Process_operation_definition_relationship` is a relationship between two `Process_operation_definition` objects.

### **Base Class**

- `PLM_object` (ABS)

### **Attributes**

- `relation_type` : `String` [1]  
The `relation_type` specifies the meaning of the relationship. Where applicable the following values shall be used:

'alternative'	The application object defines a relationship where the related <code>Process_operation_definition</code> may be used alternatively instead of the relating <code>Process_operation_definition</code> .
'substitution'	The application object defines a relationship where the related <code>Process_operation_definition</code> replaces the relating <code>Process_operation_definition</code> .
'version association'	The application object defines a relationship where the related <code>Process_operation_definition</code> is a version of the relating <code>Process_operation_definition</code> . In this case, only the related <code>Process_operation_definition</code> shall specify a <code>version_id</code> .
'version sequence'	The application object defines a relationship where the relating <code>Process_operation_definition</code> is the preceding version and the related <code>Process_operation_definition</code> is the following version. In this case, both <code>Process_operation_definition</code> objects shall specify a <code>version_id</code> .

### **Compositions**

- none

### **Associations**

- `related` : `Process_operation_definition` [1]  
The `related` specifies the second of the two objects related by the `Process_operation_definition_relationship`.

#### **8.12.1.6 Class `Process_operation_input_or_output`**

A `Process_operation_input_or_output` is the input or expected result of a `Process_operation_definition`.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- role : String [1]  
The role specifies whether the identified element plays the role of an input or an output for the operation.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Process\_operation\_input\_or\_output.

### **Associations**

- concerned\_shape : Shape\_element [0..\*]  
The concerned\_shape specifies the set of Shape\_element objects that are affected by the Process\_operation\_occurrence.
- placement : Transformation (ABS) [0..1]  
The placement specifies the geometrical Transformation between the local coordinate system of the element acting as Process\_operation\_input\_or\_output, and the reference coordinate system. The reference coordinate system is either the coordinate system of the reference tool, if present, for the concerned Process\_operation\_occurrence or, if no reference tool is present, the coordinate system of the Process\_operation\_occurrence itself.
- element : Process\_operation\_input\_or\_output\_select [1]  
The element specifies the element that plays the role of the input or the output for the operation.

### **8.12.1.7 Class Process\_operation\_occurrence**

A Process\_operation\_occurrence is the usage of a Process\_operation\_definition in a Process\_plan. This association states that the Process\_operation\_definition is part of the Process\_plan.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- id : String [1]  
The id specifies the identifier of the Process\_operation\_occurrence.

### **Compositions**

- process\_operation\_resource\_assignment : Process\_operation\_resource\_assignment [0..\*]  
The process\_operation\_resource\_assignment specifies the process\_operation\_resource\_assignment that is associated with this Process\_operation\_occurrence.
- process\_operation\_occurrence\_relationship : Process\_operation\_occurrence\_relationship [0..\*]  
The process\_operation\_occurrence\_relationship specifies the process\_operation\_occurrence\_relationship that relates the first of the two Process\_operation\_occurrence objects.
- process\_operation\_input\_or\_output : Process\_operation\_input\_or\_output [0..\*]  
The process\_operation\_input\_or\_output specifies the process\_operation\_input\_or\_output that is associated with this Process\_operation\_occurrence.

- configuration : Configuration [0..\*]  
The configuration specifies the configuration that controls this Process\_operation\_occurrence for its valid usage.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Process\_operation\_occurrence.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- operation\_definition : Process\_operation\_definition [1]  
The operation\_definition specifies the Process\_operation\_definition that defines the Process\_operation\_occurrence in a Process\_plan.
- is\_defined\_in : Cartesian\_coordinate\_space (ABS) [0..1]  
The is\_defined\_in specifies the Cartesian\_coordinate\_space of the Process\_operation\_occurrence for the case where none of the tools associated by Process\_operation\_input\_or\_output plays the role of a reference tool defining the reference coordinate space.
- plan : Process\_plan [1]  
The plan specifies the Process\_plan to which the Process\_operation\_occurrence is assigning a Process\_operation\_definition.

### **8.12.1.8 Class Process\_operation\_occurrence\_relationship**

A Process\_operation\_occurrence\_relationship is a relationship between two Process\_operation\_occurrence objects.

#### **Base Class**

- PLM\_object (ABS)

#### **Attributes**

- relation\_type : String [1]  
The relation\_type specifies the meaning of the relationship. Where applicable the following values shall be used:

'decomposition'	The application object defines a relationship where the related Process_operation_occurrence is one of the components of the relating Process_operation_occurrence.
'exclusiveness'	The application object defines a relationship where the relating and the related Process_operation_occurrence shall not have any overlap in time of execution.
'sequence'	The application object defines a relationship where the relating Process_operation_occurrence shall be completed before the related Process_operation_occurrence starts.
'simultaneity'	The application object defines a relationship where the relating and the related Process_operation_occurrence are considered as occurring during the same time period.
'substitution'	The application object defines a relationship where the related Process_operation_occurrence replaces of the relating Process_operation_occurrence.

### **Compositions**

- description : String\_select [0..1]  
The description specifies additional information about the Process\_operation\_occurrence\_relationship.
- change : Change [0..\*]  
The change specifies the change for which this object references a modified object and the corresponding original object.

### **Associations**

- cycle\_time : Duration [0..1]  
The cycle\_time specifies the interval of time within which both Process\_operation\_occurrence objects have to take place in order to be declared as simultaneous.
- waiting\_time : Property\_value (ABS) [0..1]  
The waiting\_time specifies the time that shall elapse, at least, between the completion of the relating Process\_operation\_occurrence and the start of the related Process\_operation\_occurrence. The referenced shall have a definition that is a Duration\_property.
- related : Process\_operation\_occurrence [1]  
The related specifies the second of the two Process\_operation\_occurrence objects related by a Process\_operation\_occurrence\_relationship.

#### **8.12.1.9 Class Process\_operation\_resource\_assignment**

A Process\_operation\_resource\_assignment is a mechanism to associate a resource with a Process\_operation\_occurrence.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- reference\_tool : Boolean [1]  
The reference\_tool specifies whether or not the resource identified by the Process\_operation\_resource\_assignment plays the role of the reference tool for the occurrence of an operation.

### **Compositions**

- reason : String\_select [0..1]  
The reason specifies the rationale behind the use of the resource for a particular Process\_operation\_occurrence.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

### **Associations**

- placement : Transformation (ABS) [0..1]  
The placement specifies the geometrical Transformation between the local coordinate system of the Process\_operation\_resource\_assignment and the reference coordinate system.
- resource\_definition : Resource\_definition\_select [1]  
The resource\_definition specifies the tool used to perform the operation.

### 8.12.1.10 Class Process\_plan

A Process\_plan is the manufacturing planning information, necessary to realize or produce a particular version of an Item.

#### **Base Class**

- PLM\_root\_object (ABS)

#### **Attributes**

- plan\_id : String [1]  
The plan\_id specifies the identifier of the Process\_plan that shall be unique within the scope of an organization.

#### **Compositions**

- process\_plan\_relationship : Process\_plan\_relationship [0..\*]  
The process\_plan\_relationship specifies the process\_plan\_relationship that relates the first of the two Process\_plan objects.
- description : String\_select [0..1]  
The description specifies additional information about the Process\_plan.
- name : String\_select [0..1]  
The name specifies the word or group of words by which the Process\_plan is referred to.
- configuration : Configuration [0..\*]  
The configuration specifies the configuration that controls this Process\_plan for its valid usage.
- document\_assignment : Document\_assignment [0..\*]  
The document\_assignment specifies the object that provides information for this Process\_plan.
- simple\_property\_value : Simple\_property\_value (ABS) [0..\*]  
The simple\_property\_value specifies the assigned simple property values.

#### **Associations**

- produced\_output : Item\_version [0..\*]  
The produced\_output specifies the set of Item\_version objects that are produced by the operations of the Process\_plan.

### 8.12.1.11 Class Process\_plan\_relationship

A Process\_plan\_relationship is the relationship between two Process\_plan objects.

#### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- `relation_type` : String [1]  
The `relation_type` specifies the meaning of the relationship. Where applicable the following values shall be used:

'alternative'	The application object defines a relationship where the related <code>Process_plan</code> may be used alternatively to the relating <code>Process_plan</code> .
'version association'	The application object defines a relationship where the related <code>Process_plan</code> is a version of the relating <code>Process_plan</code> . In this case, the related <code>Process_plan</code> shall be a <code>Process_plan_version</code> .
'version sequence'	The application object defines a relationship where the relating <code>Process_plan</code> is the preceding version and the related <code>Process_plan</code> is the following version. In this case, both <code>Process_plan</code> objects shall be of type <code>Process_plan_version</code> .

### **Compositions**

- `description` : String\_select [0..1]  
The description specifies additional information about the `Process_plan_relationship`.
- `change` : Change [0..\*]  
The change specifies the change for which this object references a modified object and the corresponding original object.

### **Associations**

- `related` : `Process_plan` [1]  
The related specifies the second of the two `Process_plan` objects related by a `Process_plan_relationship`.

#### **8.12.1.12 Class `Process_plan_version`**

A `Process_plan_version` is a particular version of a `Process_plan`.

### **Base Class**

- `Process_plan`

### **Attributes**

- `version_id` : String [1]  
The `version_id` specifies the identification of a particular version of a `Process_plan`.

### **Compositions**

- none

### **Associations**

- none

#### **8.12.1.13 Class `Process_property_association`**

A `Process_property_association` is a mechanism to assign a property value to process related objects.

### **Base Class**

- Property\_value\_association (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- described\_element : Process\_property\_select [1]  
The described\_element specifies the object that is described by the property value.

### **8.12.1.14 Class Process\_state**

A Process\_state is a view of an in-process-item definition of a particular version of an Item. It characterizes a state of the Item\_version that occurs before the state identified by the 'related\_item\_definition.' The identifier of a Process\_state shall be unique within the context of the Item\_version and of the Process\_plan\_version.

### **Base Class**

- Design\_discipline\_item\_definition

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- related\_item\_definition : Design\_discipline\_item\_definition [1]  
The related\_item\_definition specifies the Design\_discipline\_item\_definition that defines the final item that the in-process-item is a preliminary stage of.

### **8.12.1.15 Class Process\_state\_relationship**

A Process\_state\_relationship is a relationship between two Design\_discipline\_item\_definition objects where the relating Design\_discipline\_item\_definition is view of an in-process-item definition of a particular version of an Item. It characterizes a state of the Item\_version that occurs before the state identified by the related Design\_discipline\_item\_definition.

NOTE: Process\_state\_relationship may be used to characterize intermediate states of the Item\_version, if this Item\_version is one of the produced outputs of a Process\_plan.

A Process\_state\_relationship is a type of Item\_definition\_relationship.

### **Base Class**

- Item\_definition\_relationship (ABS)



### **Attributes**

- none

### **Compositions**

- description: String\_select[0..1]  
The description specifies additional information about the Process\_state\_relationship.

### **Associations**

- none

#### **8.12.1.16 Class Same\_time\_machining\_relationship**

A Same\_time\_machining\_relationship is a relationship between two Item\_instance objects that specifies that those two objects shall be manufactured together within the same Process\_operation\_occurrence.

A Same\_time\_machining\_relationship is a type of Item\_instance\_relationship.

### **Base Class**

- Item\_instance\_relationship (ABS)

### **Attributes**

- none

### **Compositions**

- description : String\_select[0..1]  
The description specifies additional information about the Same\_time\_machining\_relationship.

### **Associations**

- placement : Transformation\_select[0..1]  
The placement specifies the geometrical Transformation between the local coordinate system of the related Item\_instance with respect to the local coordinate system of the relating Item\_instance.  
NOTE: The transformation is needed to position the related Item\_instance with respect to the relating Item\_instance for the same time machining process. In the case where the placement is realized by a Template\_instance the scale factor of that Template\_instance shall be omitted or set to 1.

## **8.12.2 Interfaces**

### **8.12.2.1 Interface Process\_operation\_input\_or\_output\_select**

This empty interface is realized by the following classes:

- Design\_discipline\_item\_definition
- Item\_instance (ABS)
- Assembly\_component\_relationship
- Mated\_item\_relationship

### 8.12.2.2 Interface Process\_property\_select

This empty interface is realized by the following classes:

- Activity\_method\_assignment
- Activity
- Process\_plan
- Process\_operation\_resource\_assignment
- Process\_operation\_occurrence
- Process\_operation\_definition

### 8.12.2.3 Interface Resource\_definition\_select

This empty interface is realized by the following classes:

- Product\_component
- Physical\_instance
- Descriptive\_specification
- Design\_discipline\_item\_definition
- Item\_instance (ABS)

## 8.13 Package Multi\_language\_support

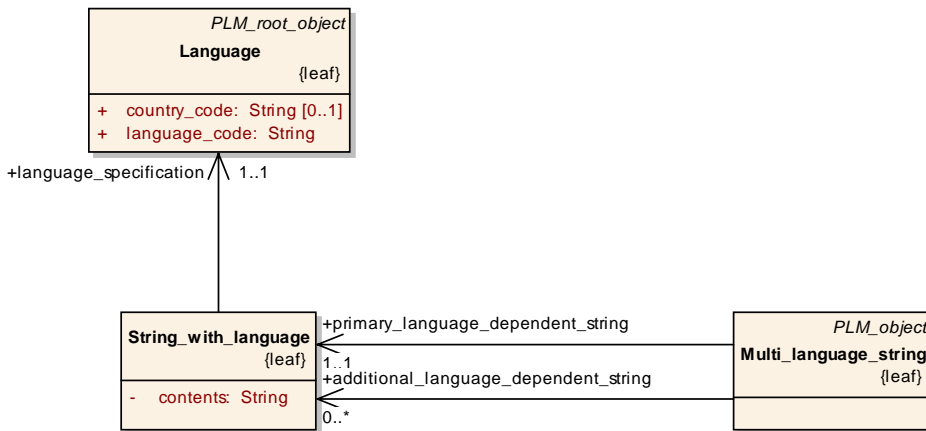


Figure 8.29 - Multi-language support

### 8.13.1 Classes

#### 8.13.1.1 Class Language

A Language is a specification of the language in which an information is given.

### **Base Class**

- PLM\_root\_object (ABS)

### **Attributes**

- language\_code : String [1]  
The language\_code specifies the language of the text information in the Alpha-3 bibliographic code specified in ISO 639-2.
- country\_code : String [0..1]  
The country\_code specifies the country, as addition to the language, according to the alpha-2 code specified in ISO 3166-1.

### **Compositions**

- none

### **Associations**

- none

## **8.13.1.2 Class Multi\_language\_string**

A Multi\_language\_string represents text information, expressed in one or more languages, that is associated with objects.

### **Base Class**

- PLM\_object (ABS)

### **Attributes**

- none

### **Compositions**

- none

### **Associations**

- primary\_language\_dependent\_string : String\_with\_language [1]  
The primary\_language\_dependent\_string specifies the String\_with\_language that represents the text information in the original language.
- additional\_language\_dependent\_string : String\_with\_language [0..\*]  
The additional\_language\_dependent\_string specifies the String\_with\_language objects that represent the text information in a particular language.

## **8.13.1.3 Class String\_with\_language**

A String\_with\_language represents text information in a specific language together with an identification of the language used.

### **Base Class**

- none

### **Attributes**

- contents : String [1]  
The contents is textual information stored in the language identified by the language attribute.

### **Compositions**

- none

### **Association**

- language\_specification : Language [1]  
The language\_specification specifies the Language in which the contents is given.

## **8.13.2 Interfaces**

### **8.13.2.1 Interface String\_select**

This empty interface is realized by the following class:

- Multi\_language\_string

## **8.13.3 Datatypes**

### **8.13.3.1 Datatype Default\_language\_string**



# 9 Computational Viewpoint (normative)

## 9.1 Overview

The computational viewpoint captures the functional aspects of the model described in Chapter 8 . There are many different use-cases for the platform independent data model. The main usage of STEP ISO 10303-214:214 [8] is the exchange of engineering data, but nowadays some companies think about using STEP as a company wide data model for all information exchange process.

To support a wide range of use cases the data model must be enriched by functional elements. Those elements should support an effective and easy to use interface for handling the data model.

The Computational Viewpoint provides the necessary life cycle functionality to create, read, update, and possibly to delete instances of the data model defined in the Informational Viewpoint. Especially, it defines a mechanism to query and traverse instances of the Informational Viewpoint. Therefore, the Computational Viewpoint is dependent on the Informational Viewpoint.

## 9.2 Base Model of the Computational Viewpoint

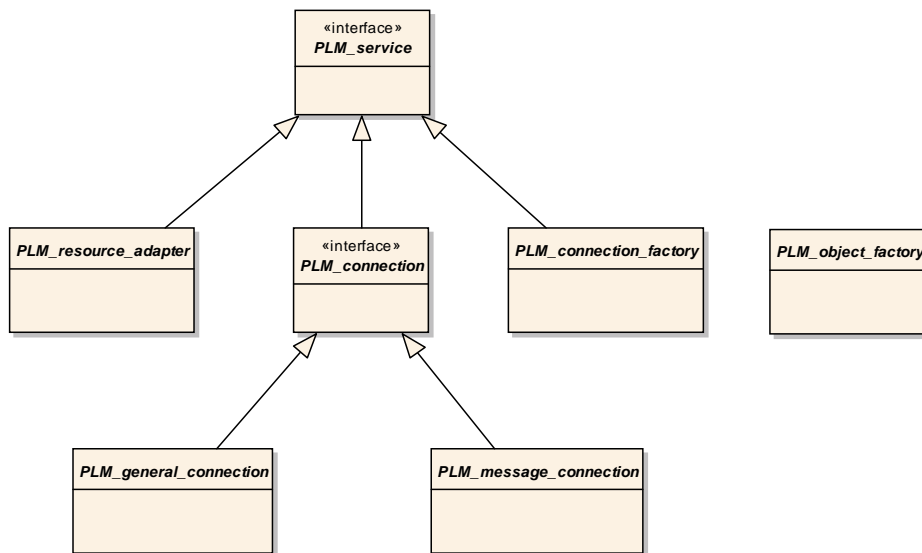
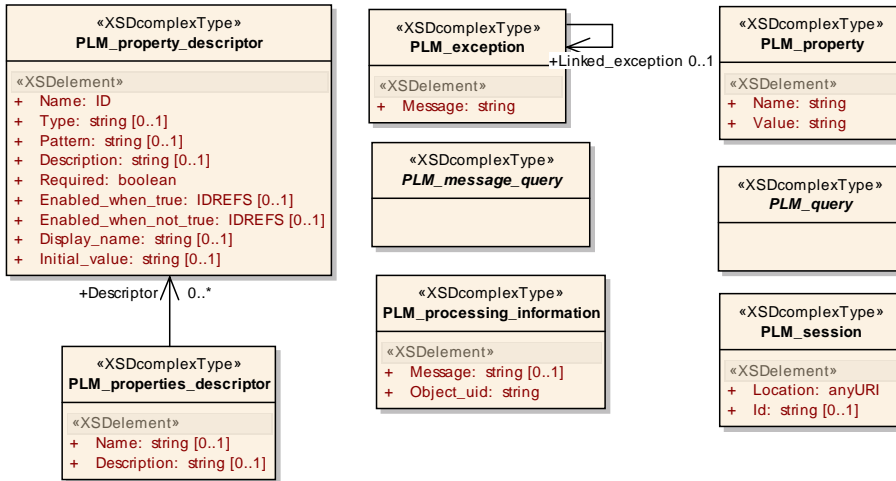


Figure 9.1 - Base Model of the Computational Viewpoint

The Computational Viewpoint has a similar functional model as a connector defined in the J2EE Connector Architecture specification. It uses five specific object types: PLM\_resource\_adapter, PLM\_object\_factory, PLM\_connection\_factory, PLM\_general\_connection and PLM\_message\_connection, the base interfaces PLM\_service and PLM\_connection, and the data types URL, UID, PLM\_query, PLM\_core\_container, PLM\_message, PLM\_context, PLM\_processing\_instruction, PLM\_properties\_descriptor and PLM\_property.

The types `PLM_core_container` and `UID` are defined in the Informational Viewpoint. The type `URL` is used to model URLs. All operations of all interfaces can throw `PLM_exception` objects.

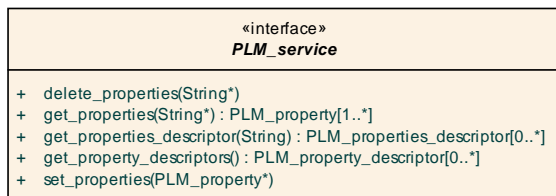


**Figure 9.2 - Using the Computational Viewpoint**

In Figure 9.2 a typical usage scenario is described that comprises the steps:

- referring to a resource adapter service,
- obtaining invocation property informations,
- retrieving a suitable connection\_factory,
- inspecting the general\_connection properties, and
- invoking a get on the factory instance.

### 9.2.1 PLM\_service Interface



**Figure 9.3 - The PLM\_service interface**

The `PLM_service` defines basic operations to set, get, delete, and get descriptions of properties of a service. It is the base definition for all services of the Computational Viewpoint.

#### **Set\_properties Operation**

`set_properties(in properties: PLM_property[]): void`

The operation `set_properties()` sets new values for service properties.

### ***Get\_properties Operation***

get\_properties(in names: String[]): PLM\_property []

The operation get\_properties() returns the properties, which names are given in the input parameter names.

### ***Delete\_properties Operation***

delete\_properties(in names: String[]): void

The operation set\_properties() sets new values for service properties.

### ***Get\_property\_descriptors Operation***

get\_property\_descriptors(): PLM\_property\_descriptors[]

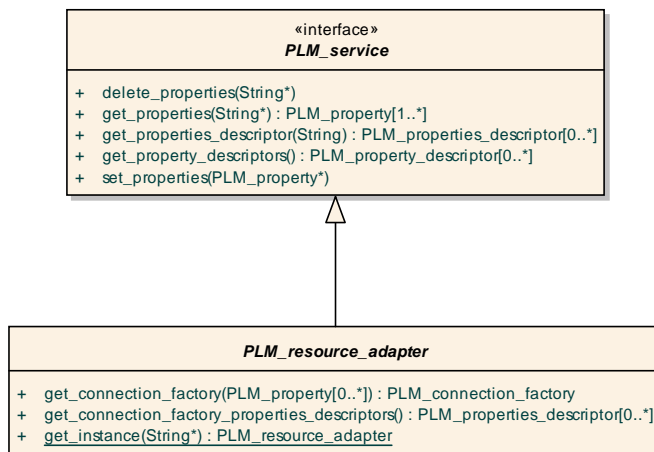
The operation get\_property\_descriptors() returns descriptors for the properties provided by a service.

### ***Get\_properties\_descriptors Operation***

get\_properties\_descriptors(operation\_name: String): PLM\_properties\_descriptor[]

The operation get\_properties\_descriptors() returns descriptors for all allowed property combination variants for a service operation that provides a set of PLM\_property elements as parameter.

## **9.2.2 PLM\_resource\_adapter Class**



**Figure 9.4 - The PLM\_resource\_adapter Class**

A PLM connector vendor must provide an implementation of the abstract **PLM\_resource\_adapter** class. A client may obtain an instance of a specific PLM resource adapter class by the static member function `get_instance()` with the class name of the specific PLM resource adapter as parameter.

By the operation `get_connection_factory()` the client can obtain a **PLM\_connection\_factory** object. The value of the parameter name is the name of the PLM connection factory. The list of all supported values for this parameter can be obtained by the operation `get_connection_factory_properties_descriptors()`. In the parameter properties the client can pass specific parameters. The values and semantics of the properties parameter will be defined in the Platform Specific Models. Examples for property



names are “java.naming.provider.url” and “java.naming.factory.initial” if the PLM connector implementation uses a JNDI name service.

### 9.2.3 PLM\_object\_factory Interface

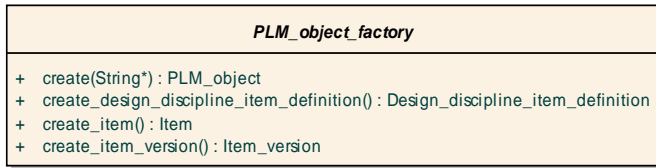


Figure 9.5 - The PLM\_object\_factory Interface (fragmentary)

The PLM\_object\_factory provides one specific create operation for each non abstract type that extends directly or indirectly PLM\_object. Additionally a generic create operation is provided. Allowed parameter values for the generic create operation are the names of those types for which a specific create operation in the PLM\_object\_factory exists. The result PLM\_objects from the create operations are local objects. The operation write() from the interface PLM\_general\_connection has to be used to transfer a local object to a PLM system (create a new object in the PLM system).

### 9.2.4 PLM\_connection\_factory Interface

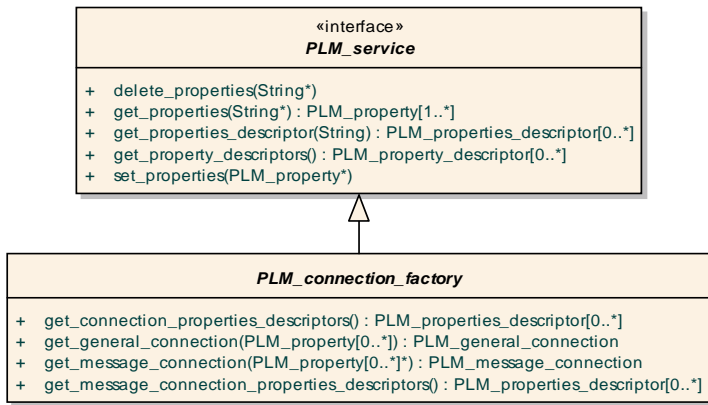
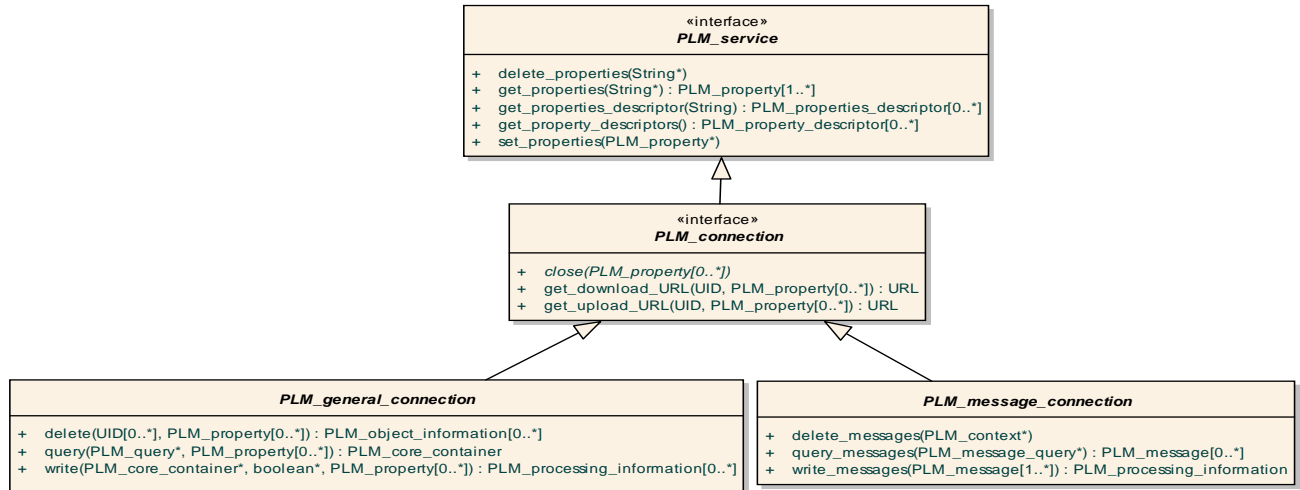


Figure 9.6 - The PLM\_connection\_factory

The interface PLM\_connection\_factory specializes the interface PLM\_service (see Figure 9.6) and provides the operation get\_general\_connection(), which returns a PLM\_general\_connection instance and the operation get\_message\_connection(), which returns a PLM\_message\_connection instance. By the parameter properties the client may pass specific information to the PLM\_connection\_factory. The actual properties are implementation specific and its descriptors can be obtained by the operation get\_properties\_descriptor() and get\_property\_descriptors().

## 9.2.5 PLM\_connection Interface



**Figure 9.7 - The PLM\_connection, PLM\_general\_connection and PLM\_message\_connection**

The **PLM\_connection** is derived from **PLM\_Service** (see Figure 9.7) and is the base interface for **PLM\_general\_connection** and **PLM\_message\_connection** and defines their common operations.

### **Get\_download\_URL Operation**

`get_download_URL(in file_uids: UID[0..*], properties: PLM_property[0..*]): URL[0..*]`

The `get_download_URL()` operation is assigned uid-attributes of **Digital\_file**-objects as the first parameter. As a return value, it delivers URLs to retrieve the content of a **Digital\_file** objects from the PLM system. The number of URLs shall be equal to the number of given `file_uids`. If semantically defined by `[0..*]`, the order of the URLs shall be the same as the order of the `in_file_uids`.

The `get_download_URL()` operation accepts a set of **PLM\_property** elements as additionally parameter. The list of all supported values for this parameter can be obtained by the operation `get_operation_properties_descriptors()`. The values and semantics of the properties parameter are implementation specific. One possible use of the properties parameter is to control the protocol of the returned URL, (e.g., “http,” “ftp,” “https”).

### **Get\_upload\_URL Operation**

`get_upload_URL(in file_uids: UID[0..*], properties: PLM_property[0..*]): URL[0..*]`

The `get_upload_URL()` operation expects uid-attributes of **Digital\_file**-objects as the first parameter. It returns URLs that are used to upload the new content of the **Digital\_file** objects to the PLM system. The number of URLs shall be equal to the number of given `file_uids`. If semantically defined by `[0..*]`, the order of the URLs shall be the same as the order of the `in_file_uids`.

The `get_upload_URL()` operation accepts a set of **PLM\_property** elements as additionally parameter. The list of all supported values for this parameter can be obtained by the operation `get_operation_properties_descriptors()`. The values and semantics of the properties parameter are implementation specific. One possible use of the properties parameter is to control the protocol of the returned URL, (e.g., “http,” “ftp,” “https”).

The value of the properties apply to all files. If different values apply to different files, single calls to `get_download_URL` with a single file shall be done.

### ***Close Operation***

`close(properties: PLM_property[0..*]): void`

The `close()` operation shuts down a connection to a PLM system. After a successful call of the close operation, all subsequent calls to this connection may raise an exception.

The `close()` operation accepts a set of `PLM_property` elements as parameter. The list of all supported values for this parameter can be obtained by the operation `get_operation_properties_descriptors()`. The values and semantics of the properties parameter are implementation specific. One possible use of the properties parameter is to control session cleanup efforts on the PLM system.

## **9.2.6 PLM\_general\_connection**

The `PLM_general_connection` is derived from the `PLM_connection_interface` (see Figure 9.7) and is the central service of this specification for a communication based on the request/respond approach. Its purpose is to grant access to the PLM system. To pass PLM data, it uses instances of the class `PLM_container`. To define the semantics of the operations, it is assumed, that all PLM data in the PLM system is instantiated as a single instance of `PLM_container` and the implementation of the operations works on that instance.

### ***Query Operation***

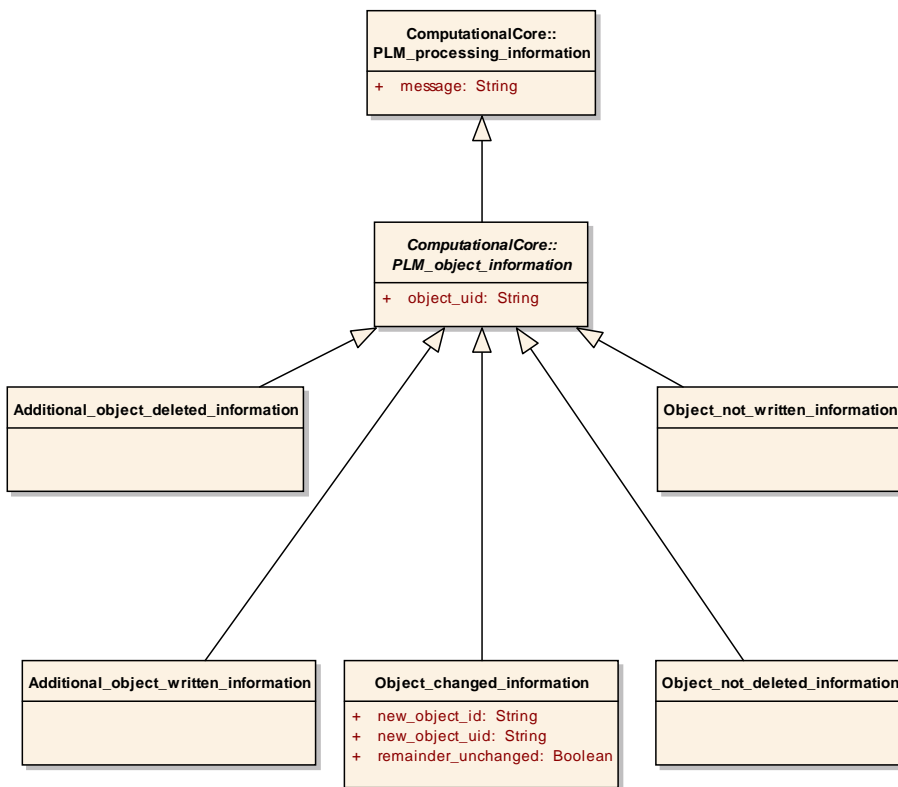
`query(in query: PLM_query, properties : PLM_property[0..*]): PLM_core_container`

The operation `query()` expects a `PLM_query` instance as its input parameter `query`. By applying this query to the data in a PLM system, a set of selected nodes is generated. As a result of the query, a `PLM_core_container` instance is returned containing all selected nodes of the query and all nodes required to fulfill the minimum multiplicity constraints of the relationships of the selected nodes. The query operation accepts a set of `PLM_property` objects as additional parameter. The allowed values and the semantic of this parameter are implementation specific and can be obtained by the operation `get_operation_properties_descriptors()`.

### ***Write Operation***

`write(data: PLM_core_container, fill_result_list: Boolean, properties: PLM_property[0..*]):  
PLM_processing_information[0..*]`

The operation `write()` expects a `PLM_core_container` instance as an input parameter. The PLM system uses the `uid-Attributes` of the single nodes in the `PLM_core_container` instance to identify which nodes already exist in the PLM System and which nodes have to be created. The operation has a return value of `PLM_processing_information` objects. In this return value information on manipulated objects is given. If the client ignores this information, the parameter `fill_result_list` shall be set to `FALSE`. By creating a new node, it is for a PLM system in general not feasible to use the attributes of the parameter data set. The operation adds one `Object_changed_information` for each changed object. If the `uid-Attribute`, the `id-Attribute` (e.g., `id`, `name`, `document_id`, `file_id`) or any other attribute has changed the `new_object_uid`, the `new_object_id`, or the `remainder_unchanged` attributes of the `Object_changed_information` are set accordingly. The result list is also used to inform the client, if not all objects of the data parameter were inserted in the PLM System. This information is added to the result list as `Object_not_inserted_information` instances. It is allowed to an implementation of the operation `write()` to add extra `PLM_objects` such as `creator` or `creation time` objects to the PLM system. If a write operation adds additional `PLM_objects` to the PLM system, this information has to be added to the result list as `Additional_objects_written_information` instances.



**Figure 9.8 - PLM\_processing\_information types of the result list of the write, import\_data and delete operations**

All elements of the data set are transferred to the PLM system. Should one element already exist, all attribute values of the existing entity in the PLM system are replaced by the attributes values of the entity in the parameter. The relationships of an existing entity are not replaced by the relationships of the corresponding entity in the parameter. Instead, the relationships of the entity of the parameter not already existing are created.

The write operation accepts a set of PLM\_property objects as additional parameter. The allowed values and the semantic of this parameter are implementation specific and can be obtained by the operation `get_operation_properties_descriptors()`. For example if the PLM system shall transform, filter, or extent the input data prior writing to its data base this behavior could be controlled by the properties parameter.

### **Delete Operation**

`delete(in uids: UID[0..*]): PLM_processing_information[0..*]`

The operation `delete()` expects a list of UID elements as input parameter. All objects with the given uids are deleted from the PLM system by this delete operation. If objects could not be deleted for some reasons, for each of those objects an `Object_not_deleted_message` instance that describes the reason has to be returned in the result list. Additionally, all nodes are deleted, which no longer fulfill the minimum multiplicity constraints of their type. For each additionally deleted object an `Additional_object_deleted_message` instance has to be added to the result list.

The operation `delete` accepts a set of PLM\_property objects as an additional parameter. The allowed values and the semantic of this parameter are implementation specific and can be obtained by the operation `get_operation_properties_descriptors()`. For

instance if the PLM system provides a recursive entity deletion capability, this functionality can be made available by a specific property parameter to the clients.

### 9.2.7 PLM\_message\_connection

The PLM\_message\_connection is derived from the PLM\_connection\_interface (see Figure 9.7) and is the central service of this specification for a communication based on the message exchange approach. It provides operations to query messages from a service, to write messages to a service, and to delete messages from a service.

#### Query\_messages Operation

Query\_messages(in query: PLM\_message\_query): PLM\_message[]

The query\_messages() operation allows a client to receive messages from a service. It expects a PLM\_message\_query as the parameter. For the message exchange based approach of client server communication, workflows must be specified that define when a client can or must send a new message to the service and when it can expect to receive a message from the service.

#### Write\_messages Operation

Write\_messages(in messages: PLM\_message[]): PLM\_processing\_information[]

The write\_messages() operation allows a client to send a message to a service. It expects one or more PLM\_message-object(s) as the parameter. The processing of a message in the service depends on the type of the message. For the message exchange based approach of client server communication workflows must be specified, which define when a client can or must send a new message to the service and when it can expect to receive a message from the service.

#### Delete\_messages Operation

Delete\_messages(in contexts: PLM\_context[]): PLM\_processing\_information[]

The delete\_messages() operation allows a client to delete explicitly messages in message queue on a service. It expects one or more PLM\_context-object(s) as the parameter to identify the messages to be deleted. If a client can or must explicitly delete a message from the message queue depends on the message type and on the workflow in which the message is used.

### 9.2.8 PLM\_property\_descriptor and PLM\_properties\_descriptor

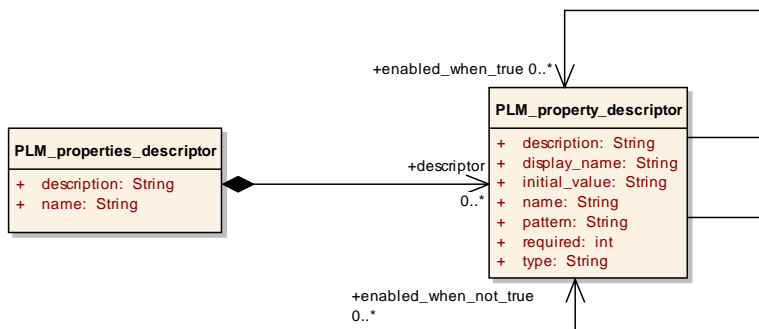


Figure 9.9 - PLM\_properties\_descriptor and PLM\_property\_descriptor

Some of the operations defined in the computational model use parameters of type `PLM_property`. The supported values of those parameters are implementation specific. Each operation with a parameter of type `PLM_property` has a corresponding operation that a client can use to obtain descriptions of the actual supported variants of values of the properties parameter. One supported variant of values is described by an instance of type `PLM_properties_descriptor`. A `PLM_properties_descriptor` has an attribute name that contains the name of the variant, an attribute description, which contains a description of the variant and a list of `PLM_property_descriptor`. Each element of the `PLM_properties_descriptor` list describes one `PLM_property` of the variant.

A `PLM_property_descriptor` describes one `PLM_property` instance. The attribute name of the `PLM_property_descriptor` defines the value of attribute name of the `PLM_property` instance. The attribute type describes the type of the `PLM_property` instance. The attribute pattern defines a pattern that must match valid values of attribute value of the `PLM_property`. The attribute description of the `PLM_property_descriptor` contains a description of the described `PLM_property` instance. The attribute required defines whether the described `PLM_property` instance must be present or if it is optional. The references `enabled_when_true` and `enabled_when_not_true` can select other `PLM_property_descriptor` instances. The selected instance must have the type Boolean and must be contained by the same `PLM_properties_descriptor` instance. If a `PLM_property_descriptor` has `enabled_when_true`- or `enabled_when_not_true` references, its attribute required must not have a value of TRUE.

A described `PLM_property` value can only be used in a properties parameter list for an operation

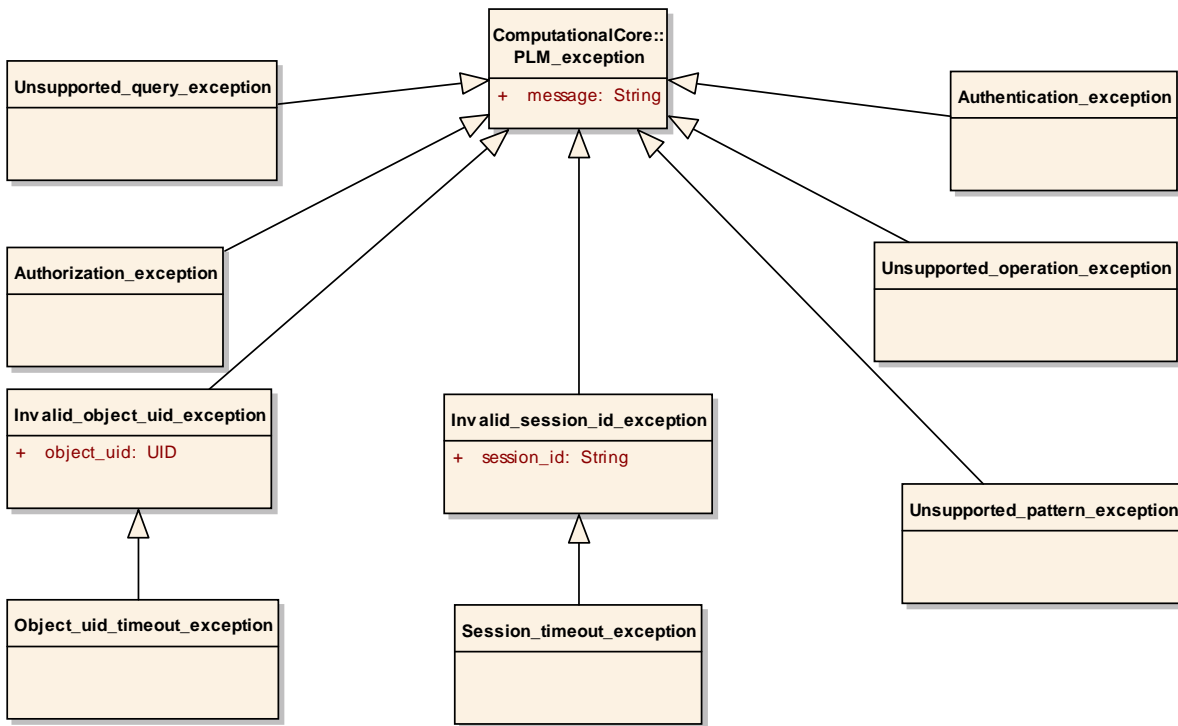
- if
  - all `PLM_property` values described by the `PLM_property_descriptors` referenced by `enabled_when_true` are also in the properties parameter list and have the value TRUE, and
  - no `PLM_property_value` described by a `PLM_property_descriptor` referenced by `enabled_when_not_true` is in the properties parameter list and has the value TRUE,
- then
  - the value of the attribute `display_name` can be used as display name of the described `PLM_property` in user interfaces.

## 9.2.9 PLM\_message\_query

The `PLM_message_query` is the abstract base class for all types that can be used as parameter for the `query_messages` operation.

### 9.2.10 PLM\_exception classes

All operations of the interfaces of the Computational Viewpoint can raise exceptions derived from `PLM_exception`. The following subtypes of `PLM_exception` the following exceptions are defined in this specification: `Authentication_exception`, `Authorization_exception`, `Session_timeout_exception`, `Invalid_session_id_exception`, `Unsupported_query_exception`, `Unsupported_operation_exception`, `Object_uid_timeout_exception`, and `Invalid_object_uid_exception`.



**Figure 9.10 - PLM\_exception and its subtypes**

### 9.2.10.1 Authentication\_exception

The Authentication\_exception is thrown by the operation get\_connection of the interface PLM\_connection\_factory when the authentication of the client fails. The authentication mechanism is implementation specific.

### 9.2.10.2 Authorization\_exception

The Authorization\_exception is thrown by arbitrary operations if the client has not the right to perform the requested operation with the given parameters.

### 9.2.10.3 Invalid\_object\_uid\_exception

The Invalid\_object\_uid\_exception is thrown by arbitrary operations of the interface PLM\_connection when a UID value of a server object is used in one parameter of the operation the associated object of which no longer exists or had never existed on the server. The UID value is returned in the attribute object\_uid of the exception.

### 9.2.10.4 Invalid\_session\_id\_exception

The Invalid\_session\_id\_exception is thrown by arbitrary operations of the interfaces PLM\_general\_connection and PLM\_message\_connection if a session identifier is used for that operation that is unknown to the service implementation. The transfer of session identifiers has to be defined by the platform specific models.

### 9.2.10.5 Object\_uid\_timeout\_exception

It is allowed to an implementation to end the validity time of an object UID before a session is closed. The Object\_uid\_timeout\_exception must be thrown by an operation of the interface PLM\_connection if such an object UID is used by a client in a parameter.

### 9.2.10.6 Session\_timeout\_exception

The Session\_timeout\_exception is thrown by arbitrary operations of the interface PLM\_connection when the session time has expired.

### 9.2.10.7 Unsupported\_query\_exception

The Unsupported\_query\_exception is thrown by the query and export\_data operation of the interface PLM\_general\_connection and by the query\_messages operation of the interface PLM\_message\_connection if a PLM\_query or PLM\_message\_query value is used as parameter, that is not supported by the service implementation.

### 9.2.10.8 Unsupported\_pattern\_exception

The Unsupported\_pattern\_exception is thrown by the query() and export\_data() operations of the interface PLM\_general\_connection and by the query\_messages operation of the interface PLM\_message\_connection if in the parameter a pattern value is used that is not supported by the service implementation.

### 9.2.10.9 Unsupported\_operation\_exception

The Unsupported\_operation\_exception is thrown by arbitrary operations of arbitrary interfaces when the requested operation is not supported by a service implementation.

## 9.2.11 PLM\_query Type

The type Query is an abstract base type. It is used as parameter in the query and export\_data operation of the PLM\_general\_connection. The type PLM\_query is specialized in “Queries Conformance Points” Section 9.3 .

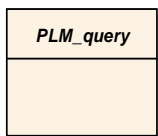


Figure 9.11 - PLM\_query Type

When a PLM\_query instance is applied to the set of PLM\_objects of a server it selects a subset of these PLM\_objects. The way of selecting this initial result set is specific to each specialization of the PLM\_query type. The initial result set of the PLM\_query instance has to be extended by further PLM\_objects of the server until the minimal result set is selected, which contains all initially selected PLM\_objects and fulfills all occurrence constraints of all selected PLM\_objects. This specification defines the following rules how to extend an initial result set to fulfill the multiplicity constraints:

- If a selected PLM\_object instance is a component in a composition, the result set has to be extended by the composite instance.



- If a selected PLM\_object instance is a composite in a composition and the multiplicity of the component in the composition is one, the result set has to be extended by the component instance.
- If a selected PLM\_object instance has a reference and the multiplicity of the referenced objects is one, the result set has to be extended by the referenced instance.
- If a selected PLM\_object instance is a composite in a composition or has a reference and the minimum multiplicity of the component respectively referenced objects is not zero and the maximum multiplicity is greater than one, the result set is not extended by selecting further PLM\_object instances. Either there are enough PLM\_object instances selected in the result set that play the component role in the composition respectively the referenced role in the directed association to fulfill the minimum multiplicity constraint or the result set is extended by NIL objects, which are used as components respectively referenced objects. NIL objects are special instances of types derived from PLM\_object that can be used as helper instances to fulfill multiplicity constraints in PLM\_object sets. The creation and distinction of NIL objects has to be defined in platform specific models.

### 9.3 Utilities Queries Conformance Point

The Utilities Queries Conformance Point provides specializations of the PLM\_query interface with the possibility of concatenated, recursive, and batch queries. All queries defined in the Utilities Queries Conformance Point are directly or indirectly derived from the abstract base type Utility\_query.

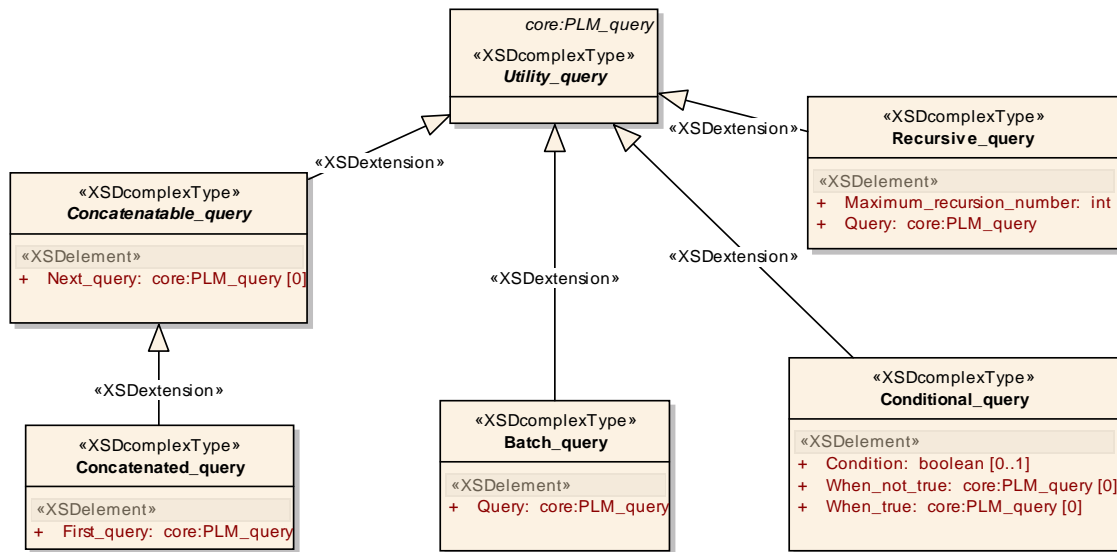


Figure 9.12 - The Utility queries class diagram

#### 9.3.1 Concatenatable\_query

The Concatenatable\_query is a base interface for query types that provides the possibility to concatenate other queries. The concatenation of queries is realized by an association that links a Concatenatable\_query object with a next PLM\_query object(s). The role name of the linked next PLM\_query object(s) is next\_query. If a query is extended by another query to a concatenated query, the result of the concatenated query is defined as the union of the results of the two single queries. The start nodes of the second query are limited to the nodes that the PLM\_general\_connection would return as a result of the first query alone.

This limitation concerns only the start nodes but not the result of the second query. In the second query all links from the result nodes of the first query to arbitrary nodes in the PLM system can be evaluated and added to the result of the second query.

### **9.3.2 Concatenated\_query**

The Concatenated\_query allows using PLM\_query instances as first query in a concatenation of queries that do not implement the Concatenatable interface.

### **9.3.3 Recursive\_query**

The Recursive\_query provides the possibility to execute other PLM\_query instances recursively. In general, executing queries against a tree of PLM objects as defined by the Informational viewpoint would require a recursive tree traversal. This recursion query instance is controlled by Recursive\_query that contains the PLM\_query instance as its attribute query. The recursion is controlled by the attribute maximum\_recursion\_number of the containing Recursive\_query. If this attribute is not set or has the value 0, the contained query is nonrecursive executed. If the attribute has a positive value n, the contained query instance has n recursions. A recursion of a PLM\_query instance has the same semantics as the concatenation of n equal PLM\_query instances. A maximum\_recursion\_number with a negative value means an infinitive recursion.

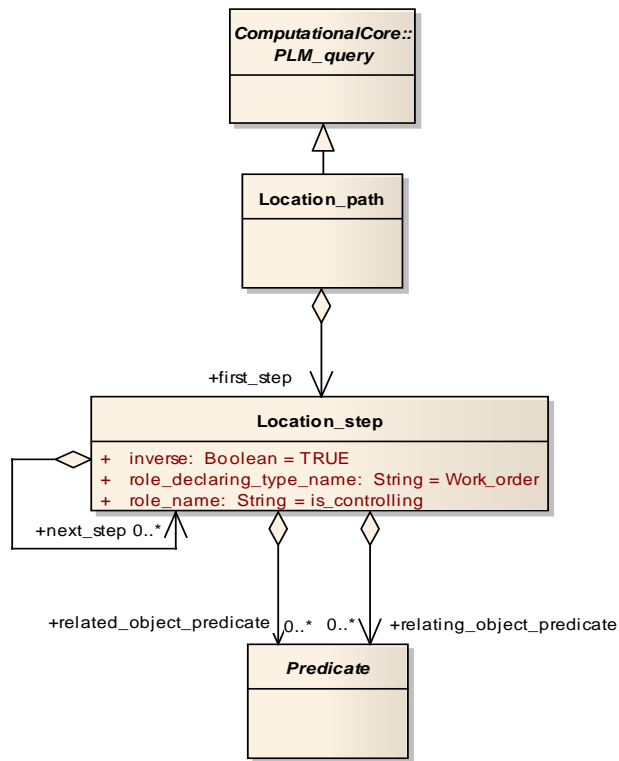
### **9.3.4 Batch\_query**

The type Batch\_query can combine other query instances to a batch job. A Batch\_query instance is evaluated by evaluating all contained PLM\_query instances of the Batch\_query instance independently and create one result from all objects selected by the contained queries.

### **9.3.5 Conditional\_query**

The type Conditional\_query enables the execution of a query in dependency of a condition. If the attribute condition is true, the query referenced by when\_true is executed, otherwise the query referenced by when\_not\_true is executed.

## 9.4 Generic Queries Conformance Point



**Figure 9.13 - The class diagram of the Generic Queries Conformance Point**

The Generic Queries Conformance Point defines a toolset of classes that can be used to query arbitrary data from a PLM system. This toolset consists of the types `Location_path`, `Location_step`, `Predicate`, and specializations of `Predicate`. The behavior of concatenation, batch, recursive, and condition processing is captured in specializations of this query type that is defined in a newly introduced Utilities Queries Conformance Point.

The `PLM_container` instance models PLM data as a set of direct or indirect contained nodes (instances of `PLM_Object`). The nodes are related by relationships. The relationship types of a node are composition or directed association. They are described in Chapter 8 for each node type.

To define a subset of the nodes of a `PLM_container` instance an instance of the abstract type `Query` has to be used. The type `Location_path` is the specialization of the `Query` type for the Generic Conformance Point. The `Location_path` is a new query tool that is designed to optimally implement the PLM Services needs. A `Location_path` consists of a tree of instances of `Location_step`.

The root node of the Tree is defined by the association `first_step` of the `Location_path`. By the association `next_step` of a `Location_step` instance the child nodes of this `Location_step` instance node in the tree are determined.

By applying a `Location_path` instance to a `PLM_container` instance each `Location_step` of the path in turn selects a set of nodes relative to the currently selected node-set.

The initially selected node-set is defined by all nodes that are directly or indirectly related to the PLM\_container instance. The resulting selected node-set of a Location\_path is the union of all selected node-sets of all Location\_steps of the Location\_path.

A location step consists of:

- a role name that specifies the nodes selected by the location step,
- the name of the type that declares the relationship with the role,
- a flag that indicates if the navigation direction is inverse in respect of the informational model,
- zero or more predicates that use arbitrary expressions further refining the set of start nodes selected by the location step,
- zero or more predicates that use arbitrary expressions further refining the set of nodes selected by the location step, and
- a list of location steps following directly the current location step.

The node-set selected by a location step is the node-set that results from generating an initial node-set from all nodes that are reached from the nodes in the current selected node-set by following the named relationship, and then filtering that node-set by each of the predicates in turn. If a Location\_step has more than one next\_step, these steps result in one different selected node-set for each step.

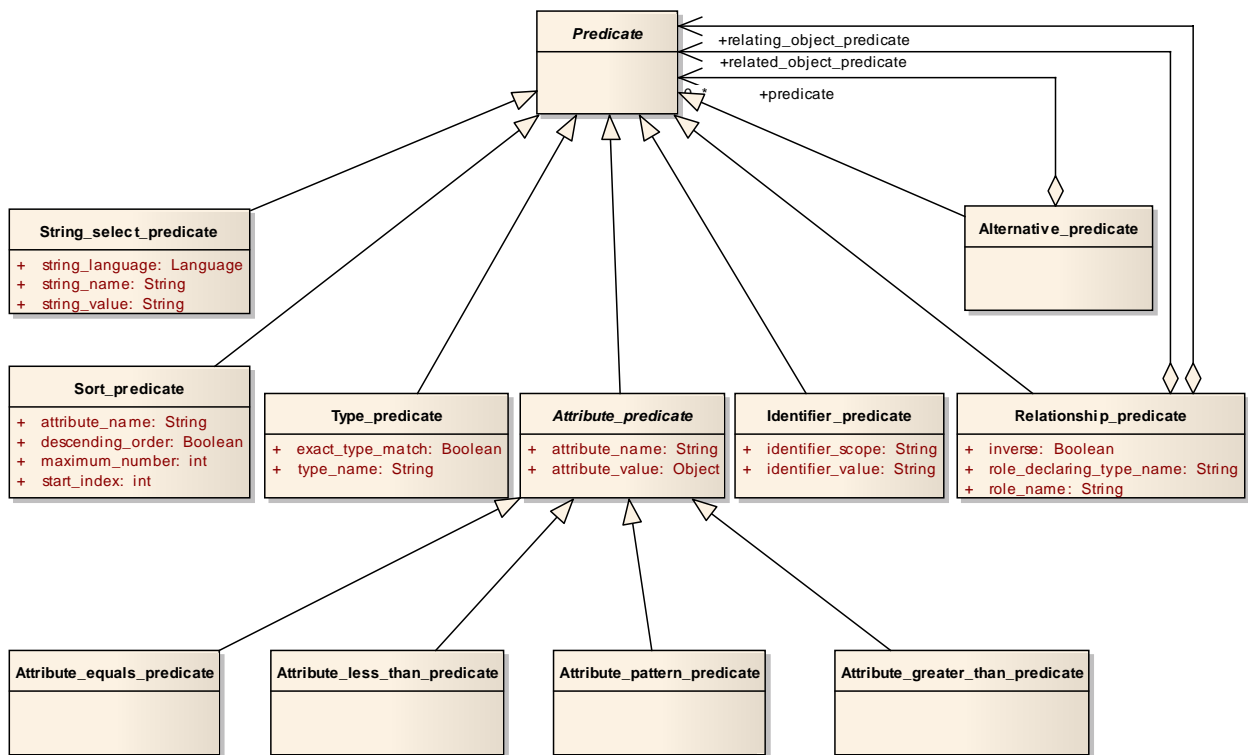


Figure 9.14 - Class diagram of predicates and attributes

Each non-abstract specialization of the abstract class Predicate defines a constraint for filtering object sets. Filtering means that the algorithm is applied to each object in the set and only the objects that fit the constraint remain in the set. The following non-abstract specializations of the class Predicate are defined in this specification:

### 9.4.1 Alternative\_predicate

An object fulfills an Alternative\_predicate constraint if it fulfills at least one of the Predicate instances referenced by the relationship predicate of the Alternative\_predicate.

### 9.4.2 Attribute\_equals\_predicate

An object fulfills an Attribute\_equals\_predicate if it has an attribute with the name given in the attribute attribute\_name of the Attribute\_equals\_predicate and if that attribute has a value that is equal to the value given by the attribute attribute\_value of the Attribute\_equals\_predicate.

### 9.4.3 Attribute\_greater\_than\_predicate

An object fulfills an Attribute\_greater\_than\_predicate if it has an attribute with the name given in the attribute attribute\_name of the Attribute\_greater\_than\_predicate and if that attribute has a value that is greater than the value given by the attribute attribute\_value of the Attribute\_greater\_than\_predicate.

### 9.4.4 Attribute\_less\_than\_predicate

An object fulfills an Attribute\_less\_than\_predicate if it has an attribute with the name given in the attribute attribute\_name of the Attribute\_less\_than\_predicate and if that attribute has a value that is less than the value given by the attribute attribute\_value of the Attribute\_less\_than\_predicate.

### 9.4.5 Attribute\_pattern\_predicate

An object fulfills an Attribute\_pattern\_predicate if it has an attribute with the name given in the attribute attribute\_name of the Attribute\_pattern\_predicate and if that attribute has a value that matches the pattern given by the attribute attribute\_value of the Attribute\_pattern\_predicate. This specification uses the pattern language defined in [XML Schema W3C Recommendation 28 October 2004].

### 9.4.6 Identifier\_predicate

All classes that have a composition of type Alias\_identification have also an attribute that corresponds with the attribute alias\_id of the related Alias\_identification. These corresponding attributes are identifying attributes and can be filtered by Identifier\_predicates.

There are three variants how an object can fulfill the constraints of an Identifier\_predicate.

- If the attribute identifier\_scope of the Identifier\_predicate is not set, an object fulfills the Identifier\_predicate if it has an identifier attribute and if that attribute has a value that matches the pattern given by the attribute identifier\_value of the Identifier\_predicate.
- If the attribute identifier\_scope of an Identifier\_predicate is set, an object fulfills the Identifier\_predicate if it has an Alias\_identification with a value for its attribute alias\_scope that is equal to the value of the attribute identifier\_scope and if the attribute alias\_id of the Alias\_identification has a value that matches the pattern given by the attribute identifier\_value of the Identifier\_predicate.
- If the attribute identifier\_scope of an Identifier\_predicate is set, an object fulfills the Identifier\_predicate if it has an identifier attribute and if that attribute has a value that matches the pattern given by the attribute identifier\_value of the Identifier\_predicate and if it is referenced by the relationship is\_applied\_to of a Person\_organization\_assignment\_instance and the attribute role of the Person\_organization\_assignment instance has

the value “id owner” and the Person\_organization\_assignment is referenced by the composition person\_organization\_assignment of an Organization instance and the attribute id of the Organization instance is equal to the value of the attribute identifier\_scope of the Identifier\_predicate.

This specification uses the pattern language defined in [XML Schema W3C Recommendation 28 October 2004].

#### 9.4.7 Relationship\_predicate

An object fulfills a Relationship\_predicate constraint if it fulfills the following partial constraints:

- The object fulfills all predicate instances referenced by the named relationship relating\_object\_predicate of the Relationship\_predicate.
- The object is related with another object that fulfills all Predicate instances referenced by the relationship predicate of the Relationship\_predicate.
- If the value of the attribute inverse of the Relationship\_predicate is set to false, and if the attribute role\_name is set, the role name of the other object in the relationship must be equal to the value of the attribute role\_name of the Relationship\_predicate.
- If the value of the attribute inverse of the Relationship\_predicate is set to true and if the attribute role\_name is set, the role name of this object in the relationship must be equal to the value of the attribute role\_name of the Relationship\_predicate.
- If the attribute role\_declaring\_type\_name is set, the relationship must be defined in a type that name is equal to the value of the attribute role\_declaring\_type\_name.

#### 9.4.8 String\_select\_predicate

An object fulfills a String\_select\_predicate if it has an attribute of type String\_select with the name given in the attribute string\_name of the String\_select\_predicate and if it fulfills one of the following constraints:

- If the attribute string\_language is not set and the attribute with the name given by the attribute string\_name must be a Default\_language\_string which value is equal to the value given by the attribute string\_value of the String\_select\_predicate.
- If the attribute string\_language is set and equals the default language of the server implementation the attribute with the name given by the attribute string\_name must be an instance of Default\_language\_string with a value that is equal to the value given by the attribute string\_value of the String\_select\_predicate or an instance of Multi\_language\_string with a primary\_language\_dependent\_string which value is equal to the value given by the attribute string\_value of the String\_select\_predicate.
- If the attribute string\_language is set and not equal to the default language of the server implementation, the attribute with the name given by the attribute string\_name must be an instance of Multi\_language\_string and have an additional\_language\_dependent\_string which value is equal to the value given by the attribute string\_value of the String\_select\_predicate.

#### 9.4.9 Type\_predicate

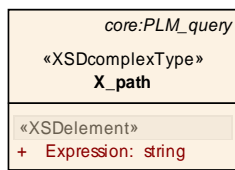
If the value of the attribute exact\_type\_match of a Type\_predicate is TRUE, an object fulfills that Type\_predicate constraint if it has the exact type specified in the attribute type\_name of the Type\_predicate.

If the value of the attribute `exact_type_match` of a `Type_predicate` is not `TRUE`, an object fulfills that `Type_predicate` constraint if it is an instance of the type specified in the attribute `type_name` of the `Type_predicate` or an instance of a derivation of that type.

### 9.4.10 Sort\_predicate

The `Sort_predicate` arranges objects of the initial node-set of a `Location_step` by the attribute specified by the attribute `attribute_name` of the `Sort_predicate`. The type of the attribute specified by `attribute_name` has to be a type for which a greater than and less than relation is defined. The default sort order is ascending order. If the attribute `Descending_order` of the `Sort_predicate` is set to true, the selected objects are sorted in descending order. If the attribute `Start_index` is set, all objects of the sorted node-set before the start index are filtered out. The default start index is 0. If the attribute `Maximum_number` is set, objects before and after the closed range `[Start_index, Start_index+Maximum_number-1]` are filtered out.

## 9.5 XPath Queries Conformance Point



**Figure 9.15 - The class diagram of the XPath Queries Conformance Point**

The XPath conformance point defines the type `X_path` as specialization of the type `Query`. The type `X_path` provides the possibility to use arbitrary XPath expressions conforming to the W3C XPath specification as queries. The Web Service PSM defined in this specification defines how a `PLM_container` instance has to be transformed to an XML-Document. An XPath expression selects nodes in this XML-Document. These nodes (or their parent nodes in the case of non XML element nodes) have equivalent instances in the PIM that are subtypes of `PLM_object`. These instances are the result set of an XPath expression at the PIM level.

## 9.6 Proxy Queries Conformance Point

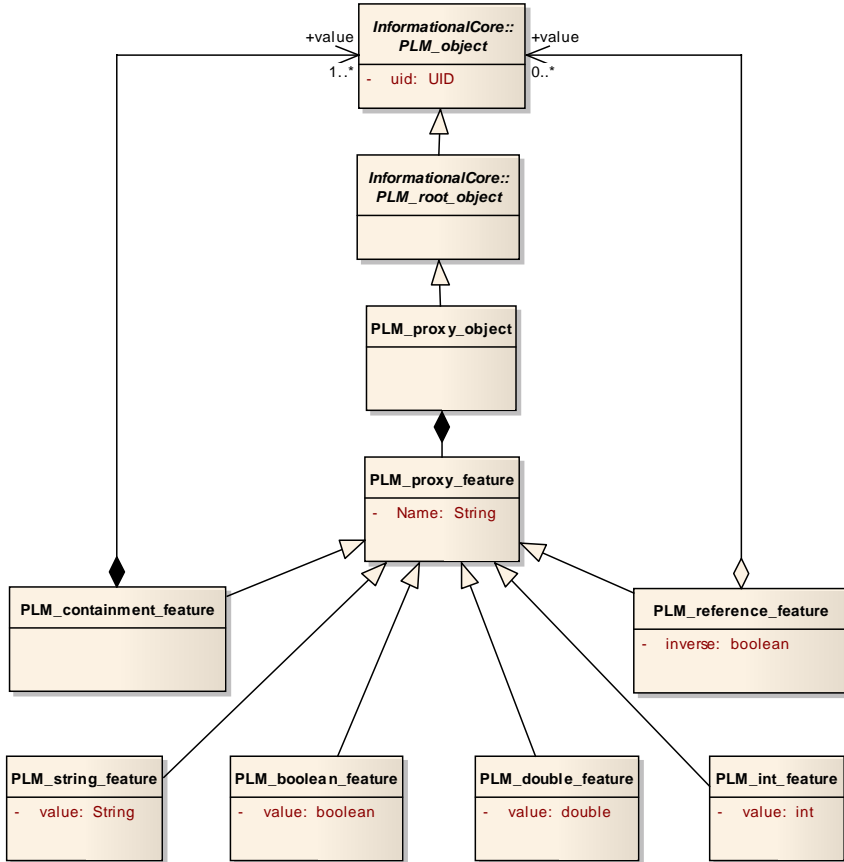


Figure 9.16 - The class diagram of the Proxy Queries Conformance Point

The Proxy Queries Conformance Point provides the possibility to avoid redundant data transfers in multiple query results and so to reduce the transferred data size. For that purpose a special container type is derived from the PLM\_container interface that can contain beside complete PLM\_objects so called proxy objects with reduced features. A client can ask proxy objects if it has stored the complete features of some objects in earlier requests.

### 9.6.1 PLM\_proxy\_object

The PLM\_proxy\_object is a proxy for an arbitrary object derived from PLM\_object and identifies the PLM object by its uid. If a client wants to receive a proxy object instead of a complete PLM object, it has to use the PLM\_proxy\_query instead of the Object\_by\_uid\_query. PLM\_proxy\_objects can also be used in the container parameter of the write operation of the PLM\_general\_connection interface to update single features of PLM\_objects.

#### Base Class

- PLM\_root\_object

#### Compositions

- PLM\_proxy\_feature: PLM\_proxy\_feature [0..\*]



## 9.6.2 PLM\_proxy\_feature

The PLM\_proxy\_feature is the abstract base class for reference and containment features.

### **Attributes**

- name: String

## 9.6.3 PLM\_containment\_feature

The PLM\_containment\_feature models a containment relation between a PLM\_proxy\_object (container) and one or more PLM\_objects (containeds).

### **Base Class**

- PLM\_proxy\_feature

### **Compositions**

- value: PLM\_object [1..\*]

## 9.6.4 PLM\_reference\_feature

The PLM\_reference\_feature models a reference between a PLM\_proxy\_object and one or more PLM\_objects. If the attribute inverse is TRUE, the PLM\_reference\_feature models references from other PLM objects to the PLM object represented by the parent object of the PLM\_reference\_feature; otherwise, it models references from the parent object of the PLM\_reference\_feature to other PLM objects.

### **Base Class**

- PLM\_proxy\_feature

### **References**

- value: PLM\_object [0..\*]

### **Attributes**

- inverse: Boolean [0..1]

## 9.6.5 PLM\_proxy\_container

The PLM\_proxy\_container extends the PLM\_container by the ability to contain PLM\_proxy\_objects.

### **Base Class**

- PLM\_container

### **Compositions**

- PLM\_propxy\_object: PLM\_proxy\_object [0..\*]

### 9.6.6 Proxy\_query

The Proxy\_query selects objects by its uid. The selected objects has to put into the result container not as copy but as PLM\_proxy\_objects. If next\_queries of the Proxy\_query traverses containments or references to other PLM\_objects, these containments or references have to be instantiated as PLM\_containment\_feature or PLM\_reference\_feature of the PLM\_proxy\_object. The client should provide not only the uid but also the opaque\_server\_data of the objects it is interested in if it has this data (e.g., from a former query result). The service implementation can use this information for an optimized processing of the PLM\_proxy\_query.

#### Base Class

- Concatenatable\_query

#### Attributes

- uid: UID
- opaque\_server\_data : Byte [0..\*]

## 9.7 Specific Queries Conformance Point

The Specific Queries Conformance Point defines a set of low level specialized queries. The semantics of each specialized query of this conformance point is defined by an equivalent Location\_path instance. The semantics of Location\_path is defined in the Generic Queries Conformance Point.

### 9.7.1 Common Queries as base types for specific queries

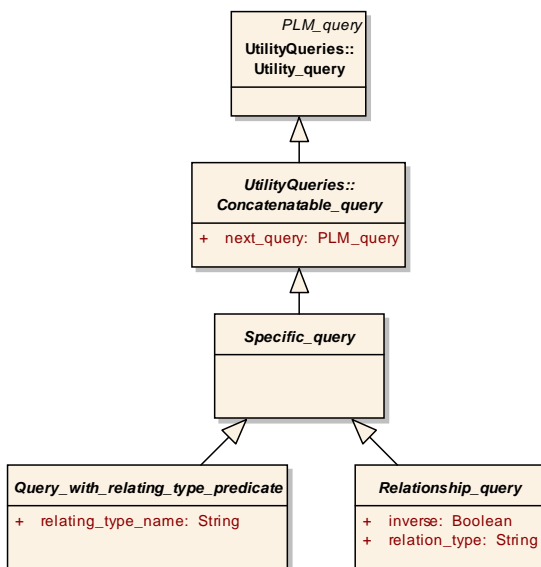


Figure 9.17 - Common base types in the specific queries conformance point

### 9.7.1.1 Class Specific\_query

All queries from the Specific Queries Point directly or indirectly derive from the Specific\_query, which implements the interface Concatenatable\_query from the Utilities Queries Conformance Point.

### 9.7.1.2 Class Query\_with\_relating\_type\_predicate

The abstract class Query\_with\_relating\_type\_predicate is used as base class for all queries that need an attribute relating\_type\_name.

### 9.7.1.3 Class Relationship\_query

The abstract class Relationship\_query is used as base class for all queries, which need an attribute relation\_type and an attribute inverse.

## 9.7.2 Activity\_query

The Activity\_query selects Activity objects.

### **Base**

- Specific\_query

### **Parameters**

- Activity\_type:string[0..1]  
filters Activity objects by their Activity\_type attribute
- Id : string[0..1]  
filters Activity objects by their Id attribute
- Id\_scope : string[0..1]  
filters Activity objects by the scope of their Id attribute
- Status : string[0..1]  
filters Activity objects by their Status attribute
- Resolved\_request\_id : string[0..1]  
filters Activity objects by the Id of their referenced Resolved\_request object
- Concerned\_organization\_id : string[0..1]  
filters Activity objects by the Id of their referenced Concerned\_organization objects
- Supplying\_organization\_id : string[0..1]  
filters Activity objects by the Id of their referenced Supplying\_organization objects
- Requestor\_id : string[0..1]  
filters Activity objects by the Id of their referenced Requestor object
- From\_requestor\_date : dateTime[0..1]  
filters Activity objects by the Id of their referenced Requestor date
- Till\_requestor\_date : dateTime[0..1]  
filters Activity objects by the Id of their referenced Requestor date
- From\_actual\_end\_date : dateTime[0..1]  
filters Activity objects by the Id of their referenced Actual\_end\_date

- **Till\_actual\_end\_date** : dateTime[0..1]  
filters Activity objects by the Id of their referenced Actual\_end\_date
- **From\_planned\_end\_date** : dateTime[0..1]  
filters Activity objects by the Id of their referenced Planned\_end\_date
- **Till\_planned\_end\_date** : dateTime[0..1]  
filters Activity objects by the Id of their referenced Planned\_end\_date
- **From\_actual\_start\_date** : dateTime[0..1]  
filters Activity objects by the Id of their referenced Actual\_start\_date
- **Till\_actual\_start\_date** : dateTime[0..1]  
filters Activity objects by the Id of their referenced Actual\_start\_date
- **From\_planned\_start\_date** : dateTime[0..1]  
filters Activity objects by the Id of their referenced Planned\_start\_date
- **Till\_planned\_start\_date** : dateTime[0..1]  
filters Activity objects by the Id of their referenced Planned\_start\_date
- **Internal** : boolean[0..1]  
filters Activity objects by the value of their Internal attribute

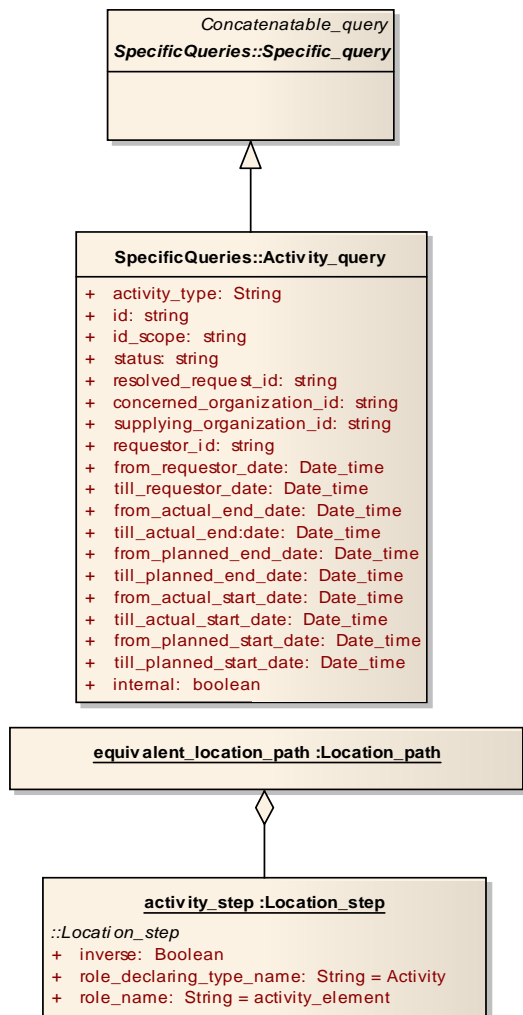


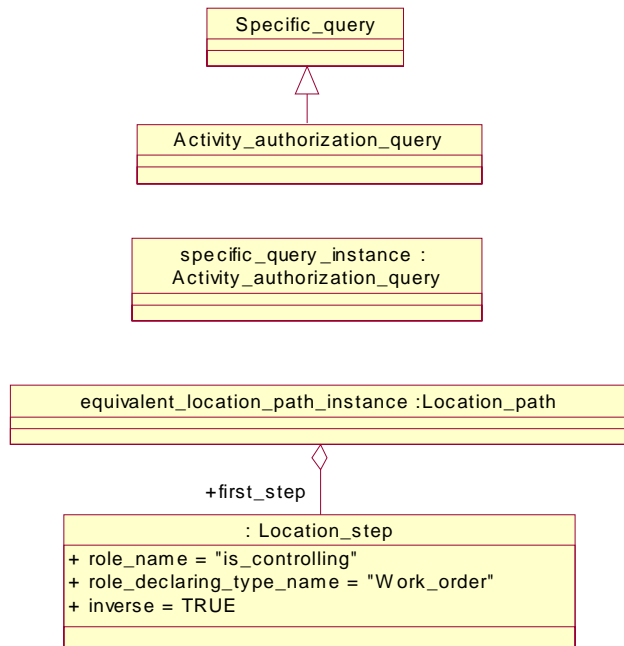
Figure 9.18 - Definition of the Activity\_query

### 9.7.3 Activity\_authorization\_query

The Activity\_authorization\_query traverses from Activity objects to Work\_order objects via the inverse Is\_controlling relationship.

#### Base Class

- Specific\_query



**Figure 9.19 - Definition, sample instance and equivalent Location\_path instance of the Activity\_authorization\_query**

### 9.7.4 Activity\_element\_query

The **Activity\_element\_query** traverses from **Activity** objects via **Activity\_element** objects to **Activity\_element\_select** objects.

#### **Base Class**

- **Specific\_query**

#### **Parameters**

- `role : String [0..1]`
- `element_type_name : String [0..1]`

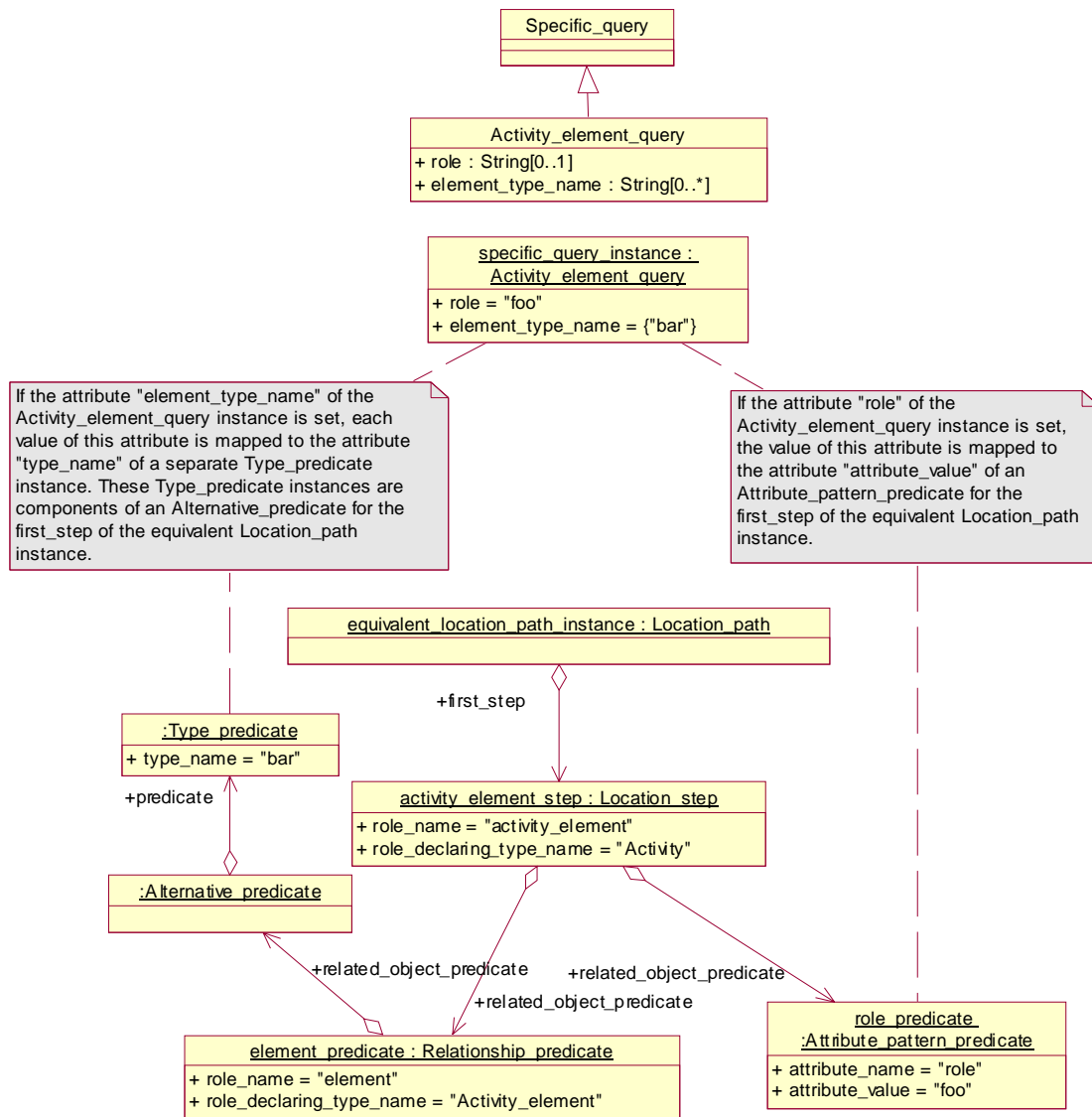


Figure 9.20 - Definition, sample instance and equivalent Location\_path instance of the Activity\_element\_query

### 9.7.5 Activity\_relationship\_query

The Activity\_relationship\_query traverses from Activity objects via Activity\_relationship objects to Activity objects.

#### Base Class

- Relationship\_query

#### Parameters

- relation\_type : String [0..1]

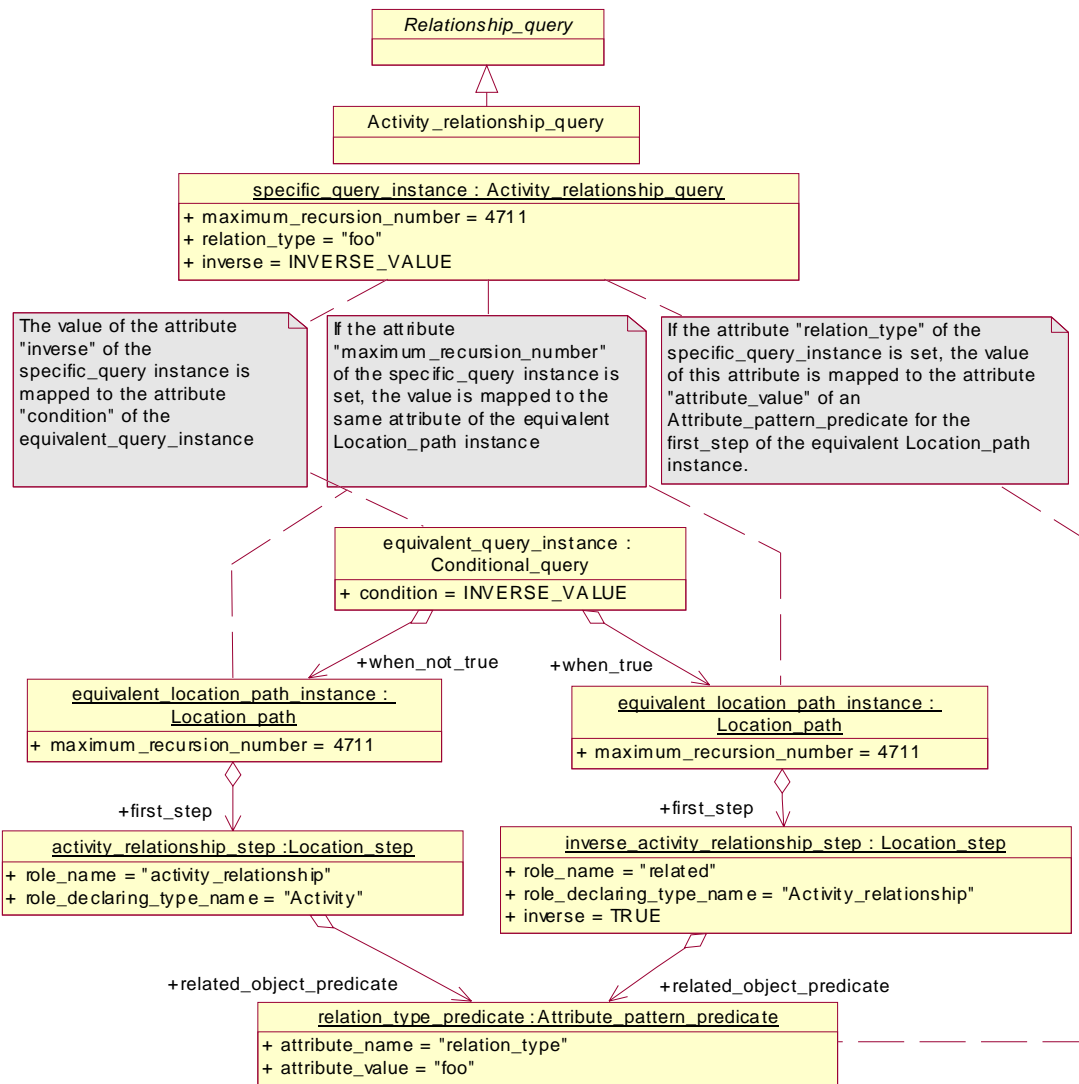


Figure 9.21 - Definition, sample instance and equivalent Location\_path instance of the Activity\_relationship\_query

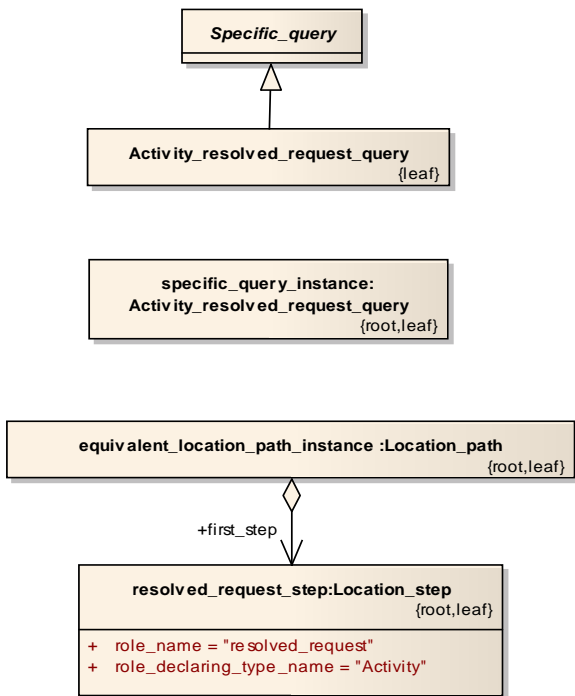
### 9.7.6 Activity\_resolved\_request\_query

The Activity\_resolved\_request\_query traverses from Activity objects to Work\_request objects via the Resolved\_request relationship.

#### Base Class

- Specific\_query





**Figure 9.22 - Definition, sample instance and equivalent Location\_path instance of the Activity\_resolved\_request\_query**

### 9.7.7 Alias\_identification\_query

The **Alias\_identification\_query** traverses alias information from instances that implements the interface **Alias\_select**.

#### **Base Class**

- **Query\_with\_releation\_type\_predicate**

#### **Parameters**

- **alias\_id** : String [0..1]
- **alias\_version\_id** : String [0..1]
- **alias\_scope** : String [0..1]

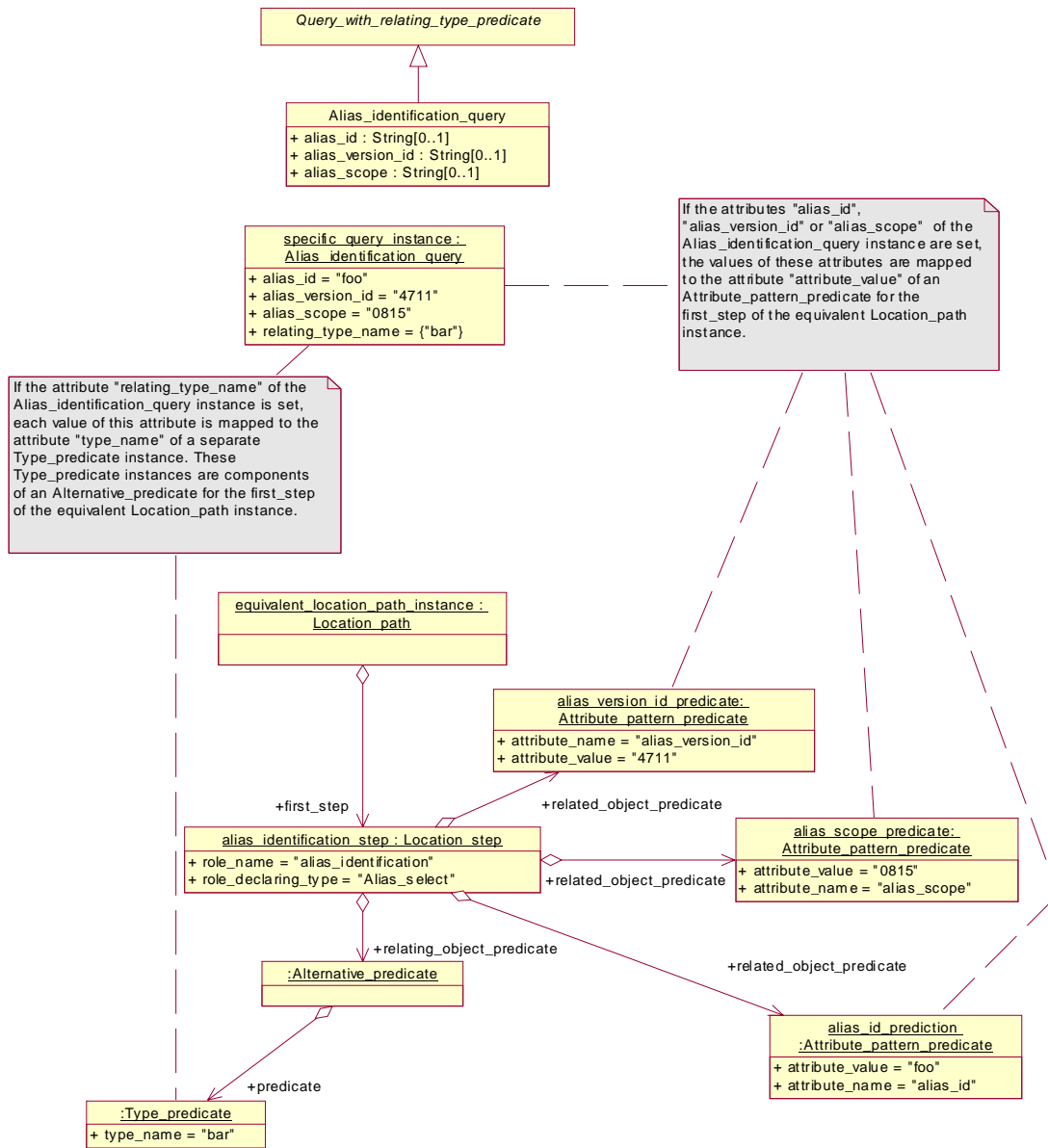


Figure 9.23 - Definition, sample instance and equivalent Location\_path instance of the Alias\_identification\_query

### 9.7.8 Alternative\_solution\_query

The Alternative\_solution\_query traverses from Complex\_product objects to Alternative\_solution objects.

#### Base Class

- Query\_with\_releation\_type\_predicate

**Parameters**

- relating\_type\_name : String [0..\*]

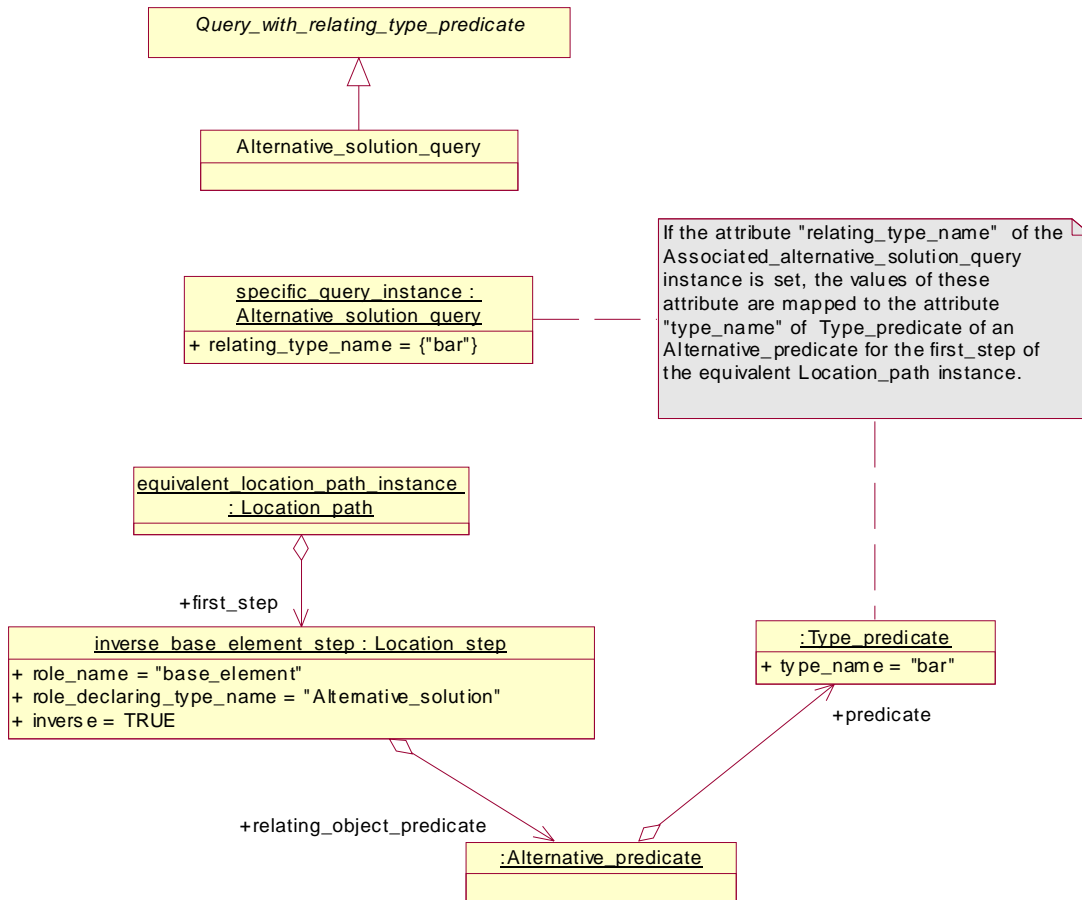


Figure 9.24 - Definition, sample instance and equivalent Location\_path instance of the Alternative\_solution\_query

**9.7.9 Application\_context\_query**

The Application\_context\_query selects Application\_context objects.

**Base Class**

- Specific\_query

**Parameters**

- application\_domain : String [0..1]
- life\_cycle\_stage : String [0..1]

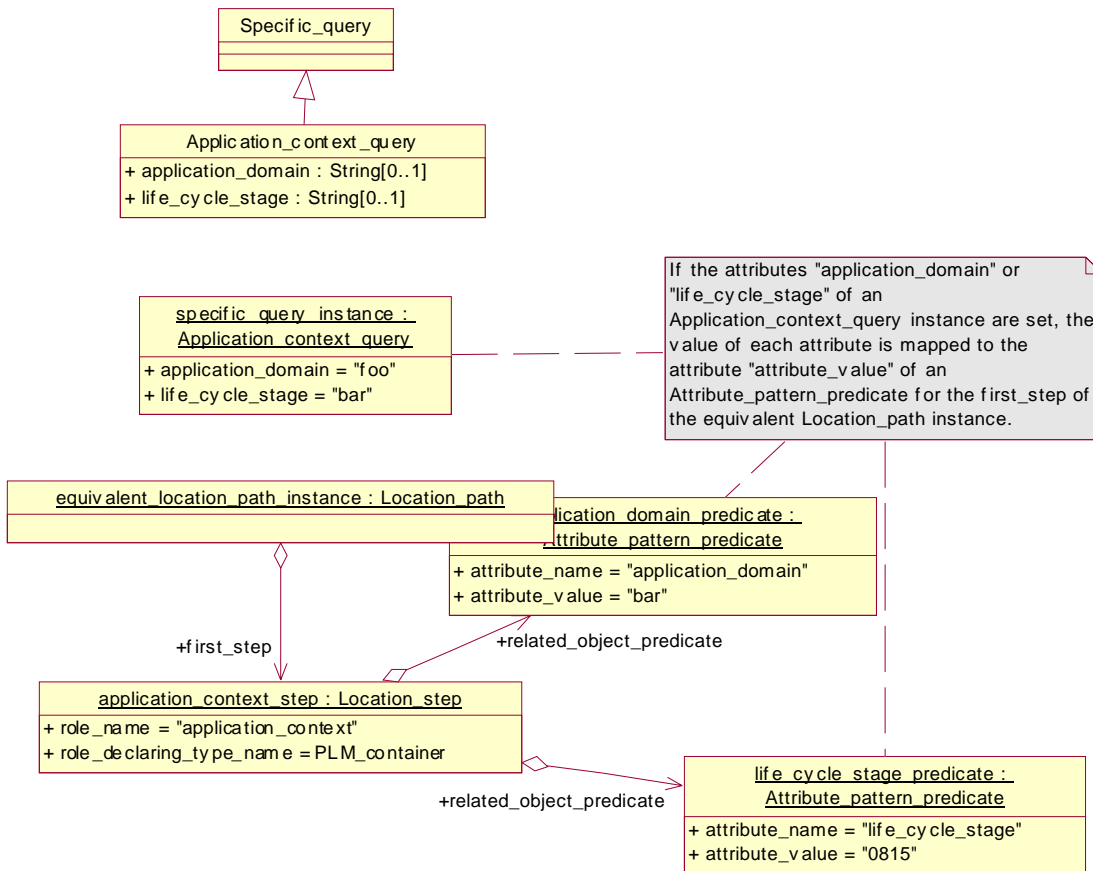


Figure 9.25 - Definition, sample instance and equivalent Location\_path instance of the Application\_context\_query

### 9.7.10 Approval\_relationship\_query

The Approval\_relationship\_query traverses from Approval objects via Approval\_relationship objects to Approval objects.

#### Base Class

- Relationship\_query

#### Parameters

- relation\_type : String [0..1]
- inverse : Boolean [0..1]

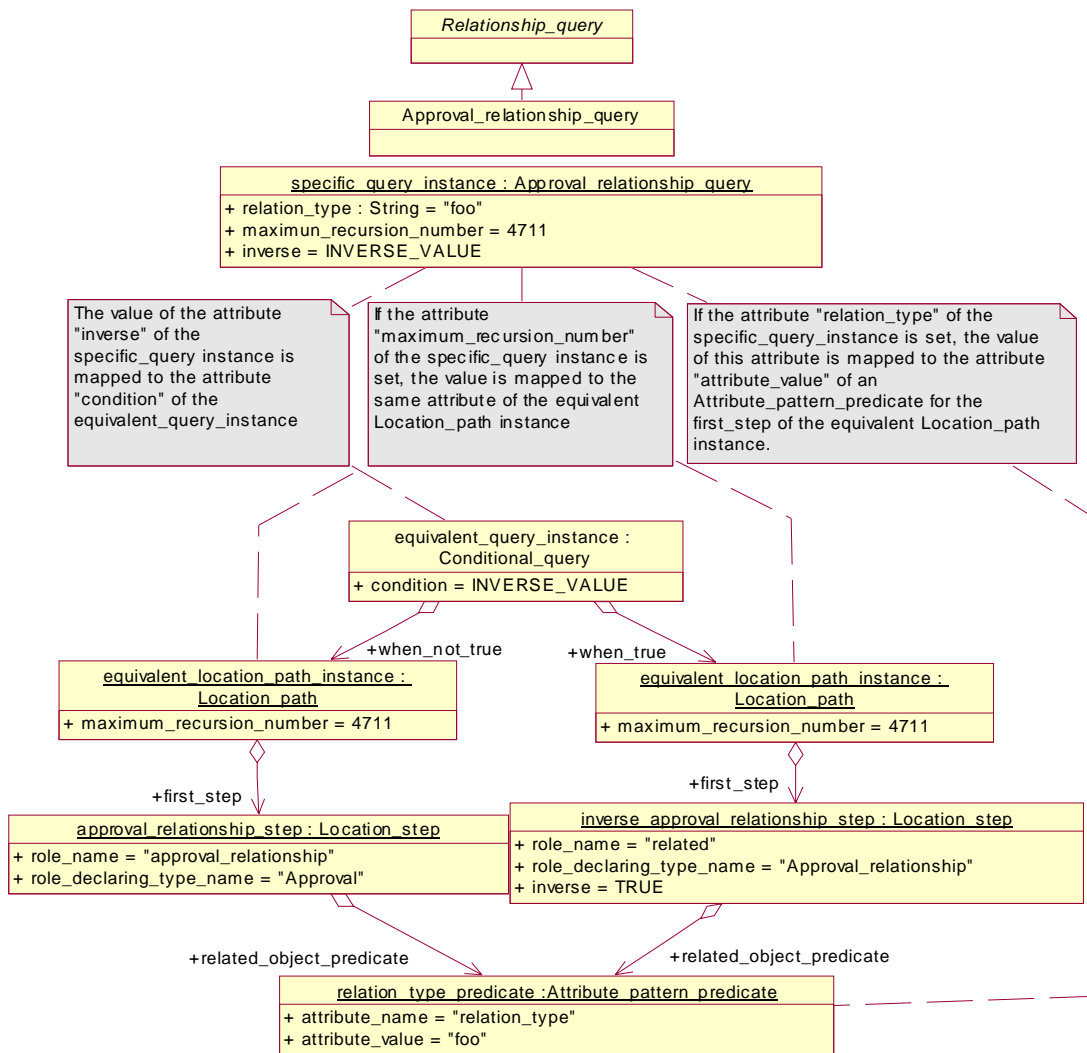


Figure 9.26 - Definition, sample instance and equivalent Location\_path instance of the Approval\_relationship\_query

### 9.7.11 Assembly\_component\_placement\_query

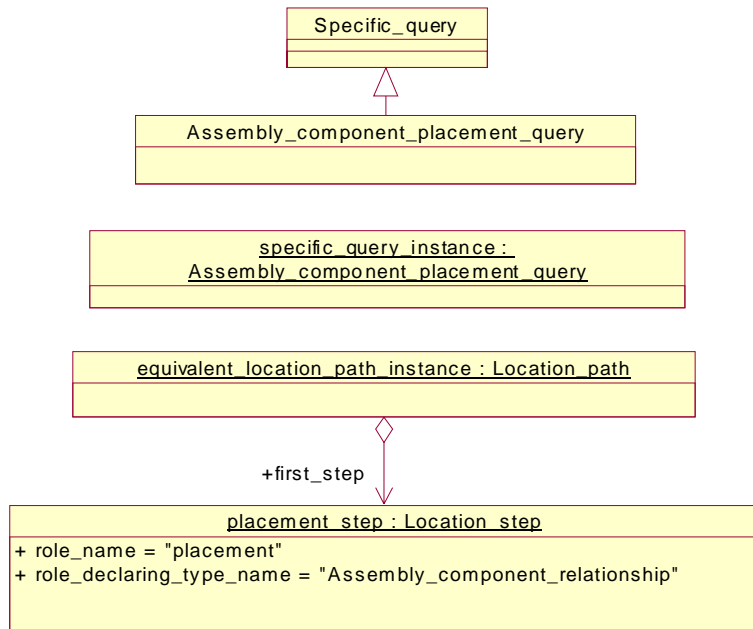
The Assembly\_component\_placement\_query traverses from Assembly\_component\_relationship objects to Transformation\_select objects.

#### Base Class

- Specific\_query

#### Parameters

- none



**Figure 9.27 - Definition, sample instance and equivalent Location\_path instance of the Assembly\_component\_placement\_query**

### 9.7.12 Associated\_activity\_query

The Associated\_activity\_query traverses from Activity\_element\_select objects via Activity\_element objects to Activity objects.

#### **Base Class**

- Query\_with\_relation\_type\_predicate

#### **Parameters**

- relation\_type : String [0..1]
- relating\_type\_name : String [0..1]

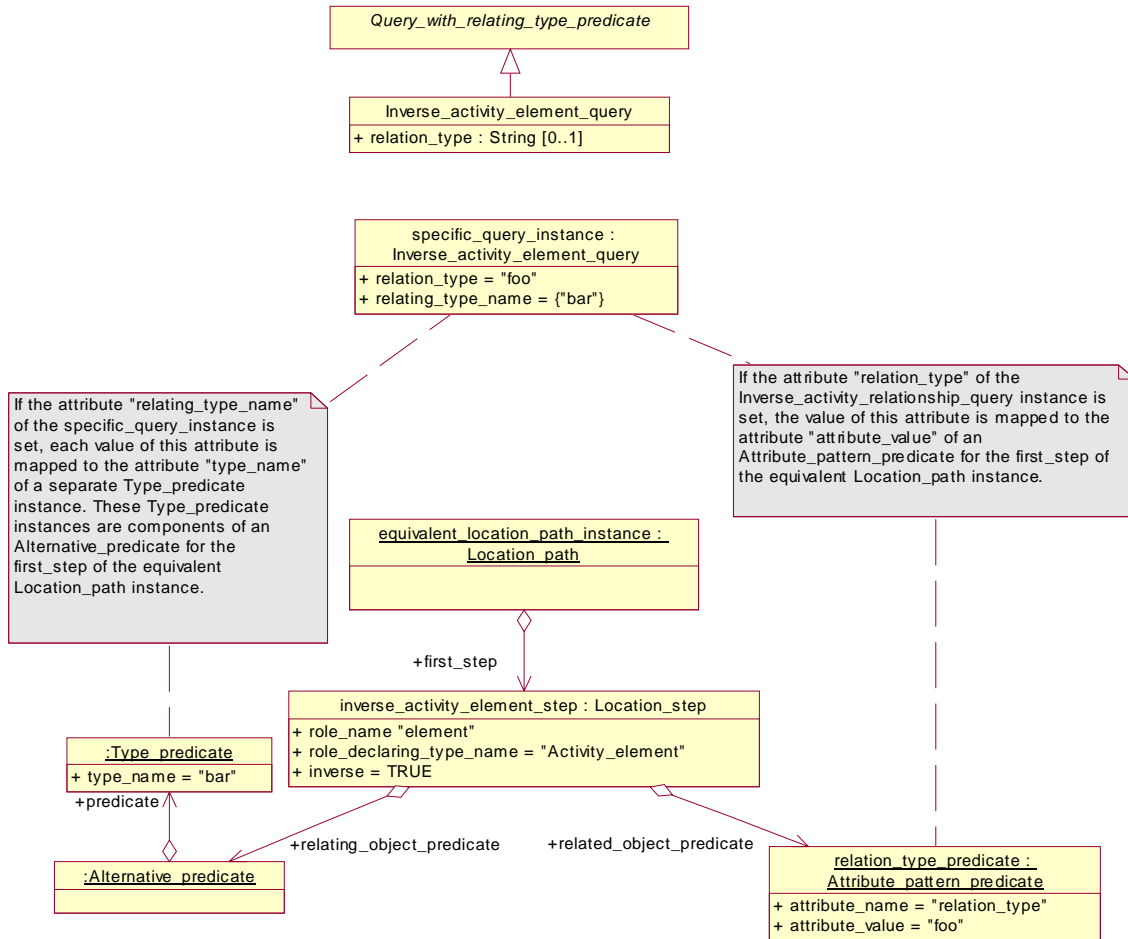


Figure 9.28 - Definition, sample instance and equivalent Location\_path instance of the Associated\_activity\_query

### 9.7.13 Associated\_approval\_query

The Associated\_approval\_query traverses from Approval\_element\_select objects to Approval objects.

#### Base Class

- Query\_with\_relation\_type\_predicate

#### Parameters

- level : String [0..1]
- relating\_type\_name : String [0..\*]

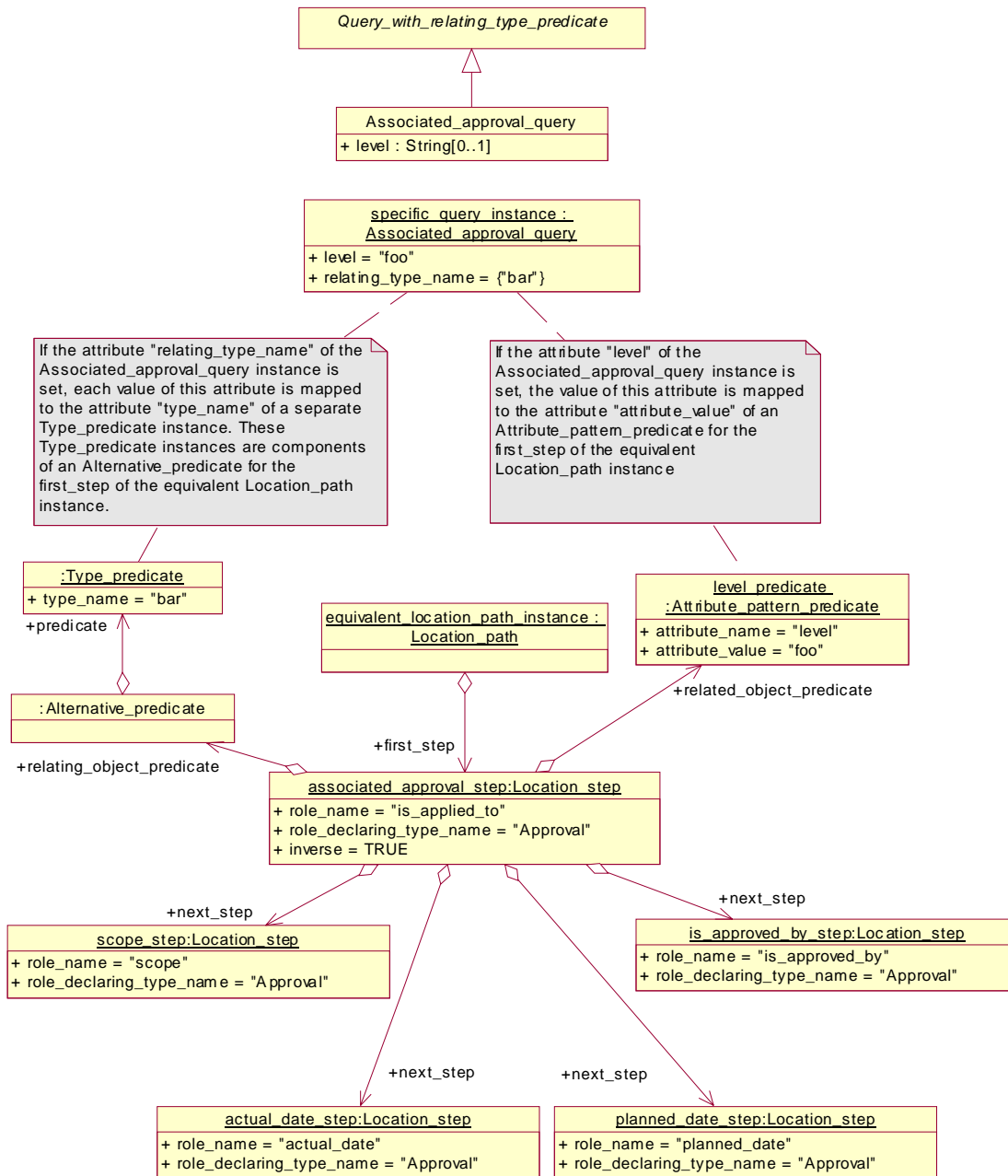


Figure 9.29 - Definition, sample instance and equivalent Location\_path instance of the Associated\_approval\_query

### 9.7.14 Associated\_classification\_query

The `Associated_classification_query` traverses from `Classified_element_select` objects via `Classification_association` objects to `General_classification` objects.



**Base Class**

- Query\_with\_relating\_type\_predicate

**Parameters**

- role : String [0..1]
- relating\_type\_name : String [0..\*]

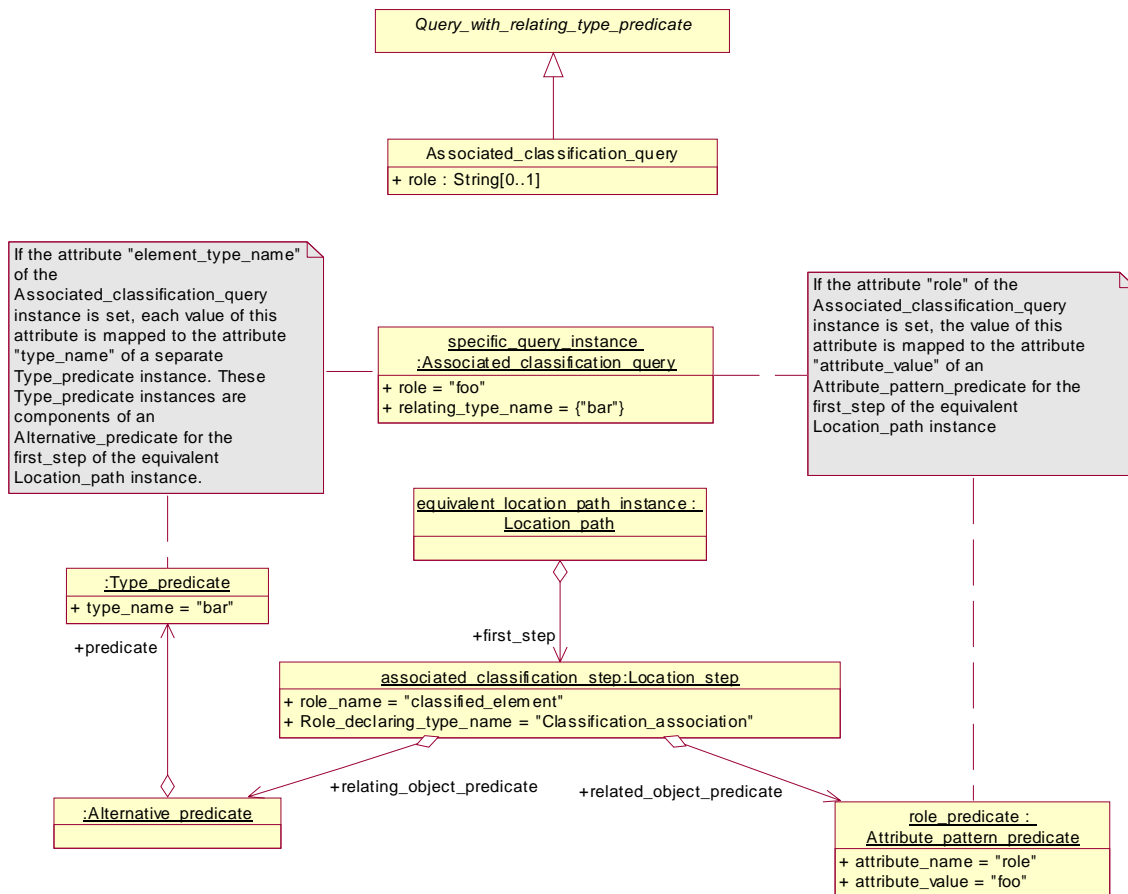


Figure 9.30 - Definition, sample instance and equivalent Location\_path instance of the Associated\_classification\_query

**9.7.15 Associated\_date\_organization\_query**

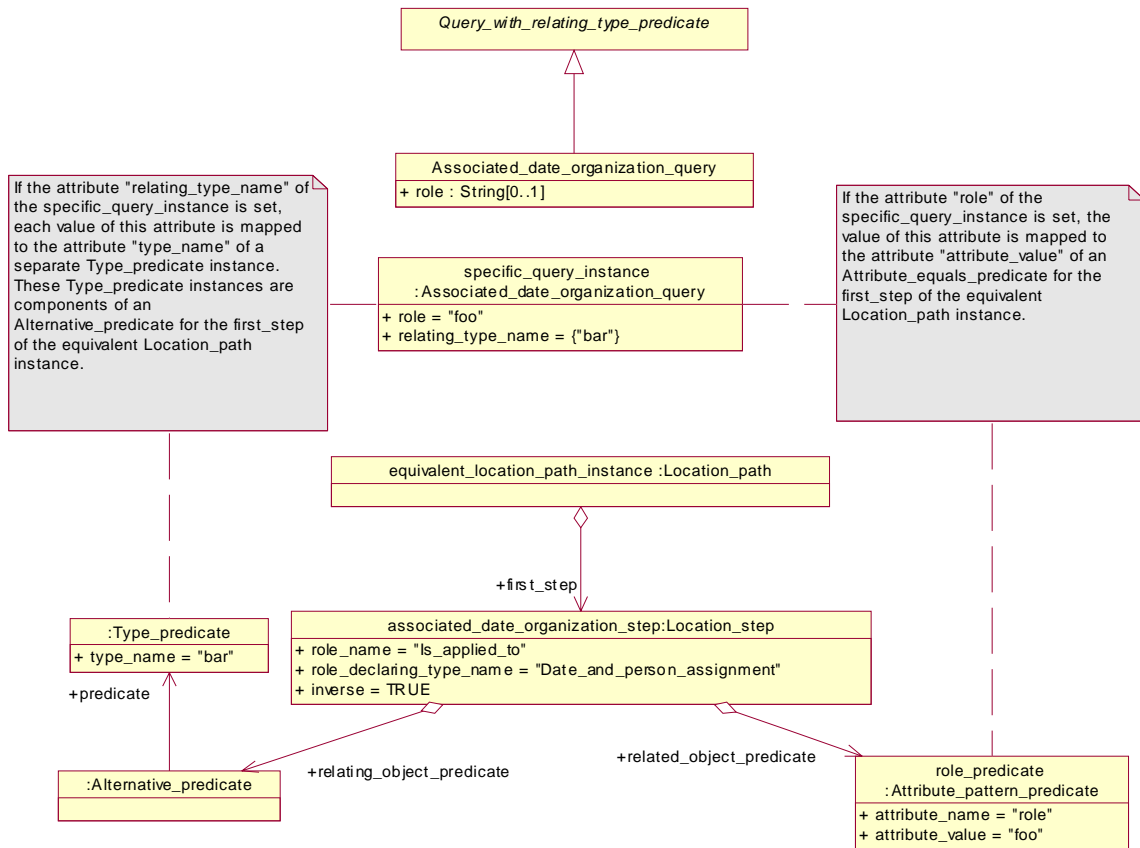
The `Associated_date_organization_query` traverses from `Date_time_person_organization_element_select` objects via `Date_and_person_assignment` objects to `Date_and_person_organization` objects.

**Base Class**

- Query\_with\_relating\_type\_predicate

**Parameters**

- role : String [0..1]
- relating\_type\_name : String [0..\*]



**Figure 9.31 - Definition, sample instance and equivalent Location\_path instance of the Associated\_date\_organization\_query**

**9.7.16 Associated\_date\_time\_query**

The Associated\_date\_time\_query traverses from Date\_time\_person\_organization\_element\_select objects via Date\_time\_assignment objects to Date\_time objects.

**Base Class**

- Query\_with\_relating\_type\_predicate

**Parameters**

- role : String [0..1]
- relating\_type\_name : String [0..\*]

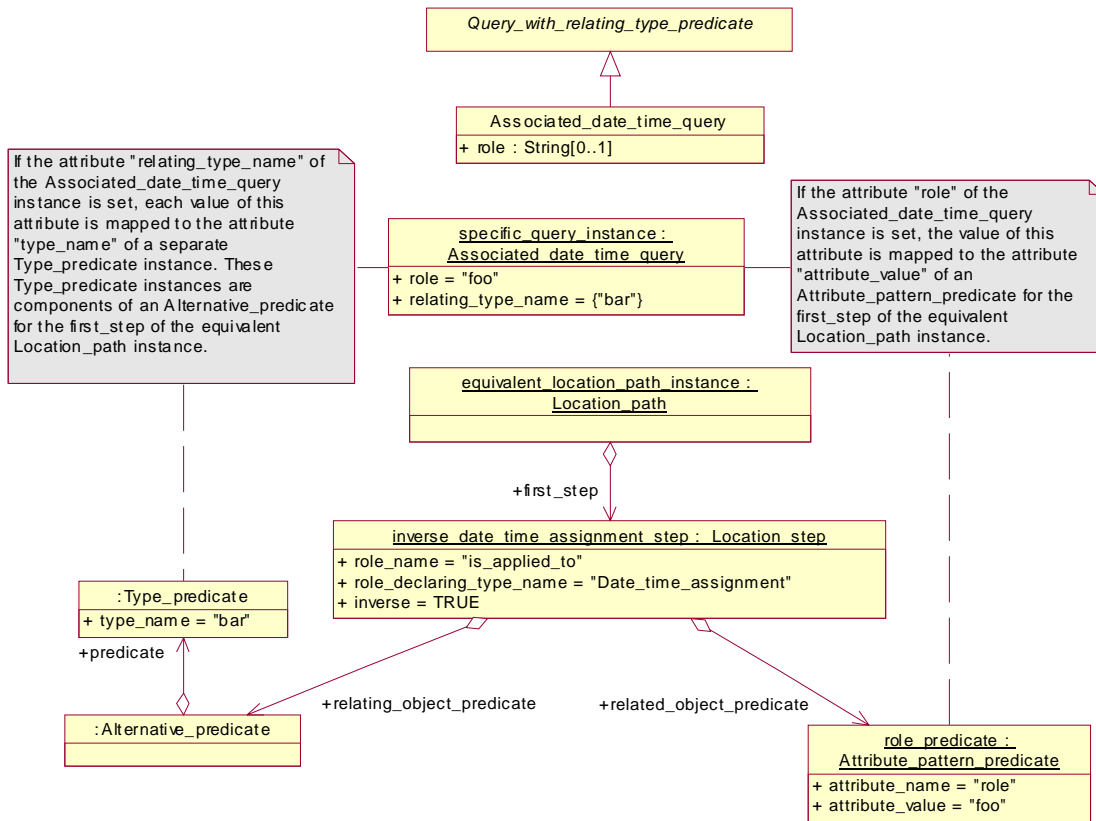


Figure 9.32 - Definition, sample instance and equivalent Location\_path instance of the Associated\_date\_time\_query

### 9.7.17 Associated\_document\_query

The Associated\_document\_query traverses from Documented\_element\_select objects via Document\_assignment objects to Assigned\_document\_select objects.

#### Base Class

- Query\_with\_relating\_type\_predicate

#### Parameters

- role : String [0..1]
- relating\_type\_name : String [0..\*]
- include\_file : Boolean [0..1]

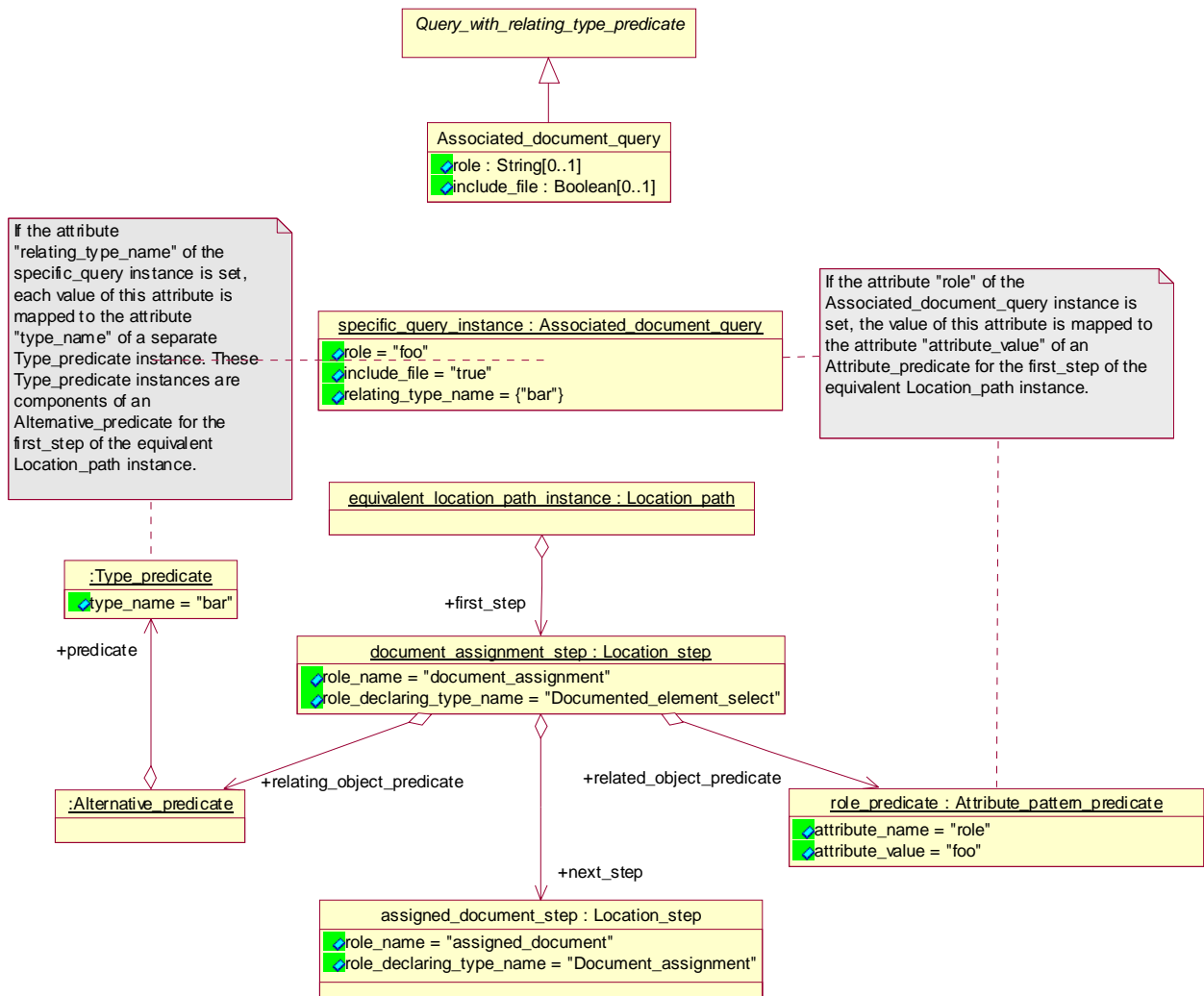


Figure 9.33 - Definition, sample instance and equivalent Location\_path instance of the Associated\_document\_query

### 9.7.18 Associated\_effectivity query

The Associated\_effectivity\_query traverses from Effective\_element\_select objects via Effectivity\_assignment objects to Effectivity objects.

#### Base Class

- Query\_with\_relating\_type\_predicate

#### Parameters

- role : String [0..1]
- relating\_type\_name : String [0..\*]

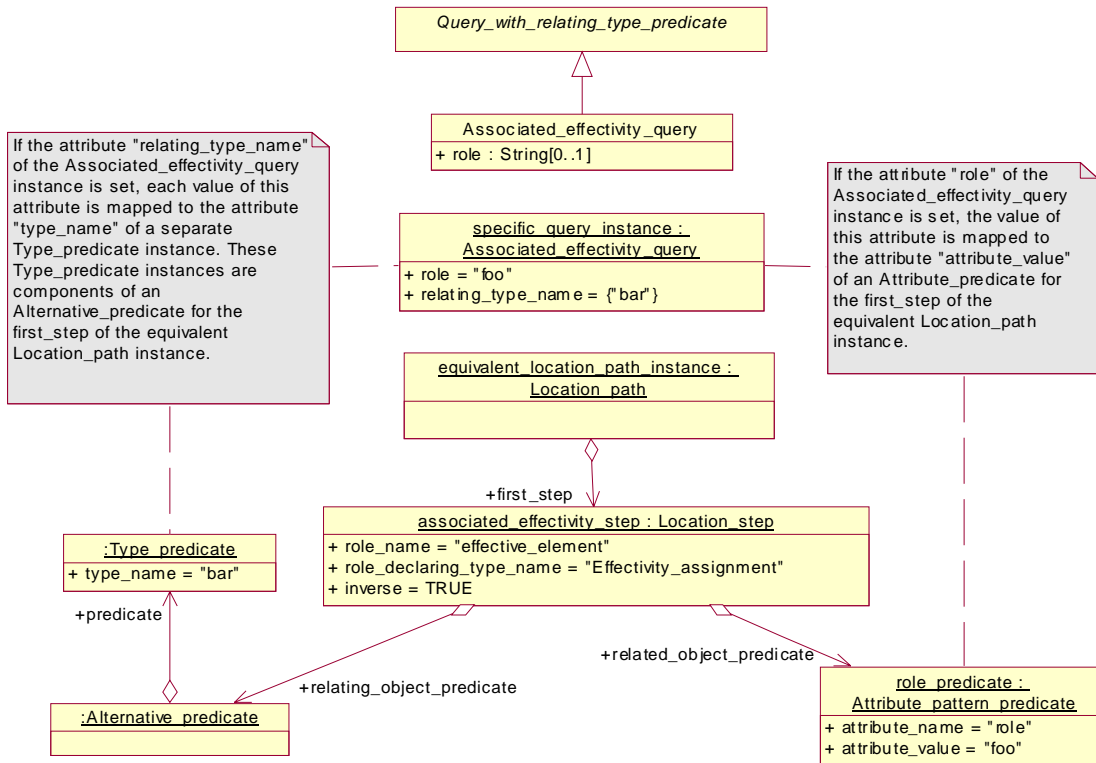


Figure 9.34 - Definition, sample instance and equivalent Location\_path instance of the Associated\_effectivity\_query

### 9.7.19 Associated\_file\_query

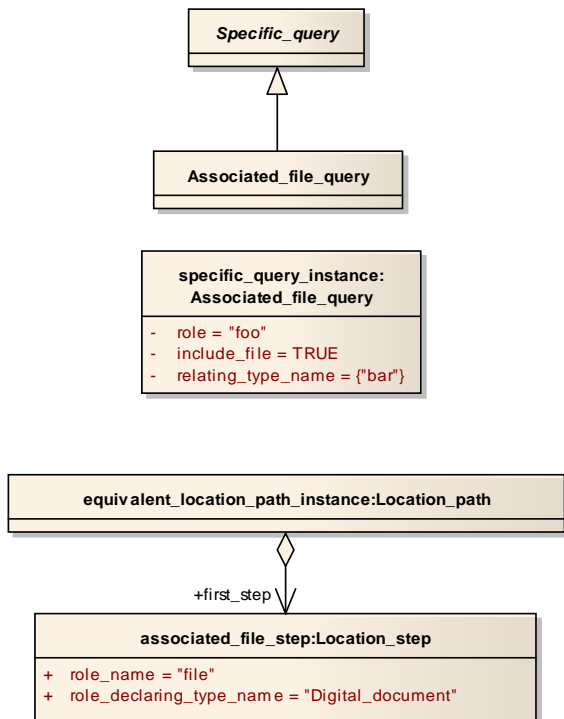
The Associated\_file\_query traverses the Digital\_file objects from Document\_representation objects.

#### Base Class

- Query\_with\_relating\_type\_predicate

#### Parameters

- role : String [0..1]
- relating\_type\_name : String [0..\*]
- include\_file : Boolean [0..1]



**Figure 9.35 - Definition, sample instance and equivalent Location\_path instance of the Associated\_file\_query**

### 9.7.20 Associated\_general\_classification\_query

The `Associated_general_classification_query` traverses from `Classified_element_select` objects to `General_classification` object via the `Classification_association` objects.

#### Base Class

- `Query_with_relating_type_predicate`

#### Parameters

- `Id: string [0..1]`
- `Id_scope: string [0..1]`
- `Version_id: string [0..1]`
- `Role: string [0..1]`

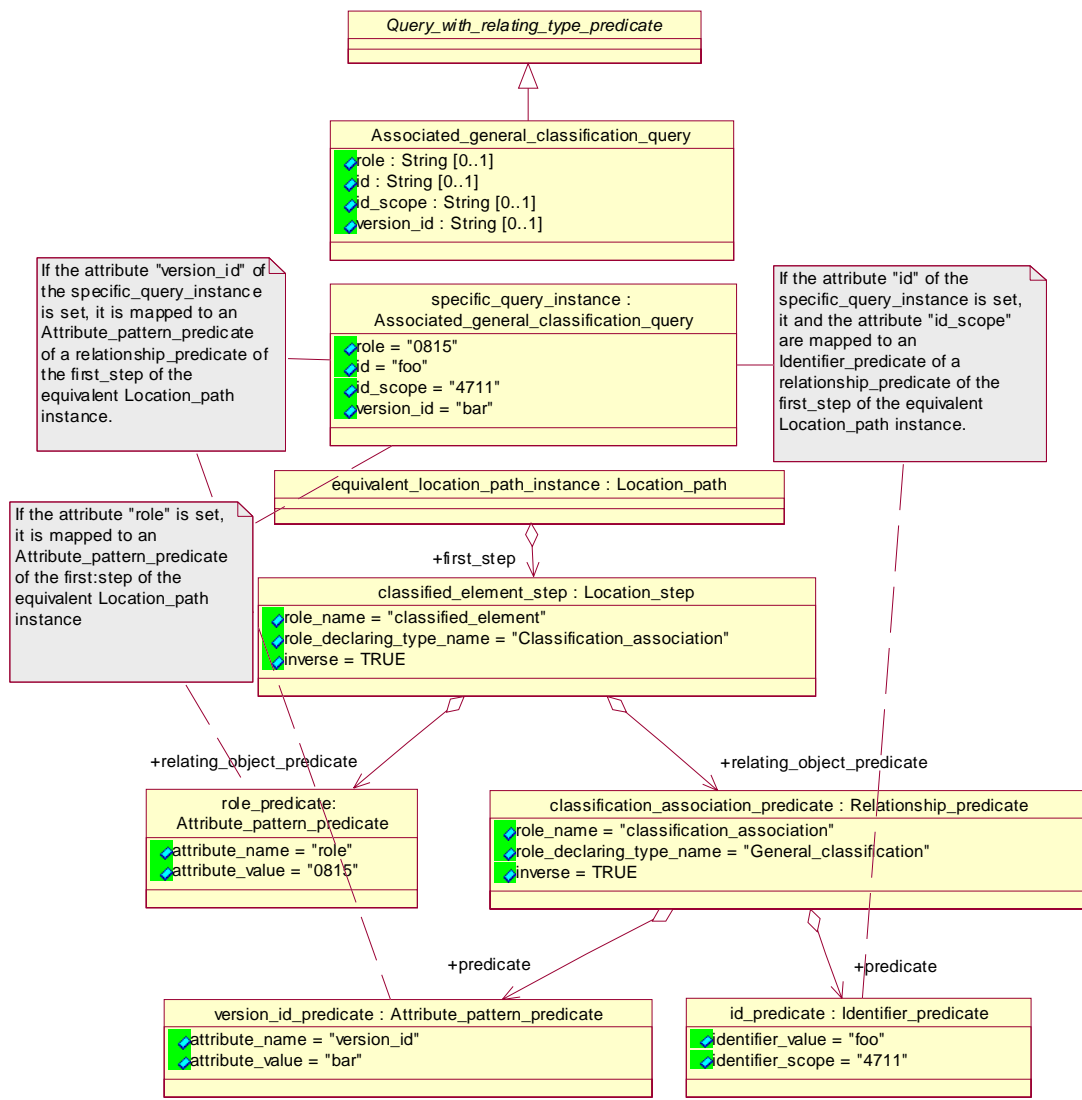


Figure 9.36 - Definition, sample instance and equivalent Location\_path instance of the Associated\_general\_classification\_query

### 9.7.21 Associated\_item\_property\_query

The Associated\_item\_property\_query traverses from Item\_property\_select objects via Property\_value\_association objects and Property\_value\_representation objects to the associated Property\_value objects.

#### Base Class

- Query\_with\_relating\_type\_predicate

## Parameters

- value\_name : String [0..1]
- relating\_type\_name : String [0..\*]

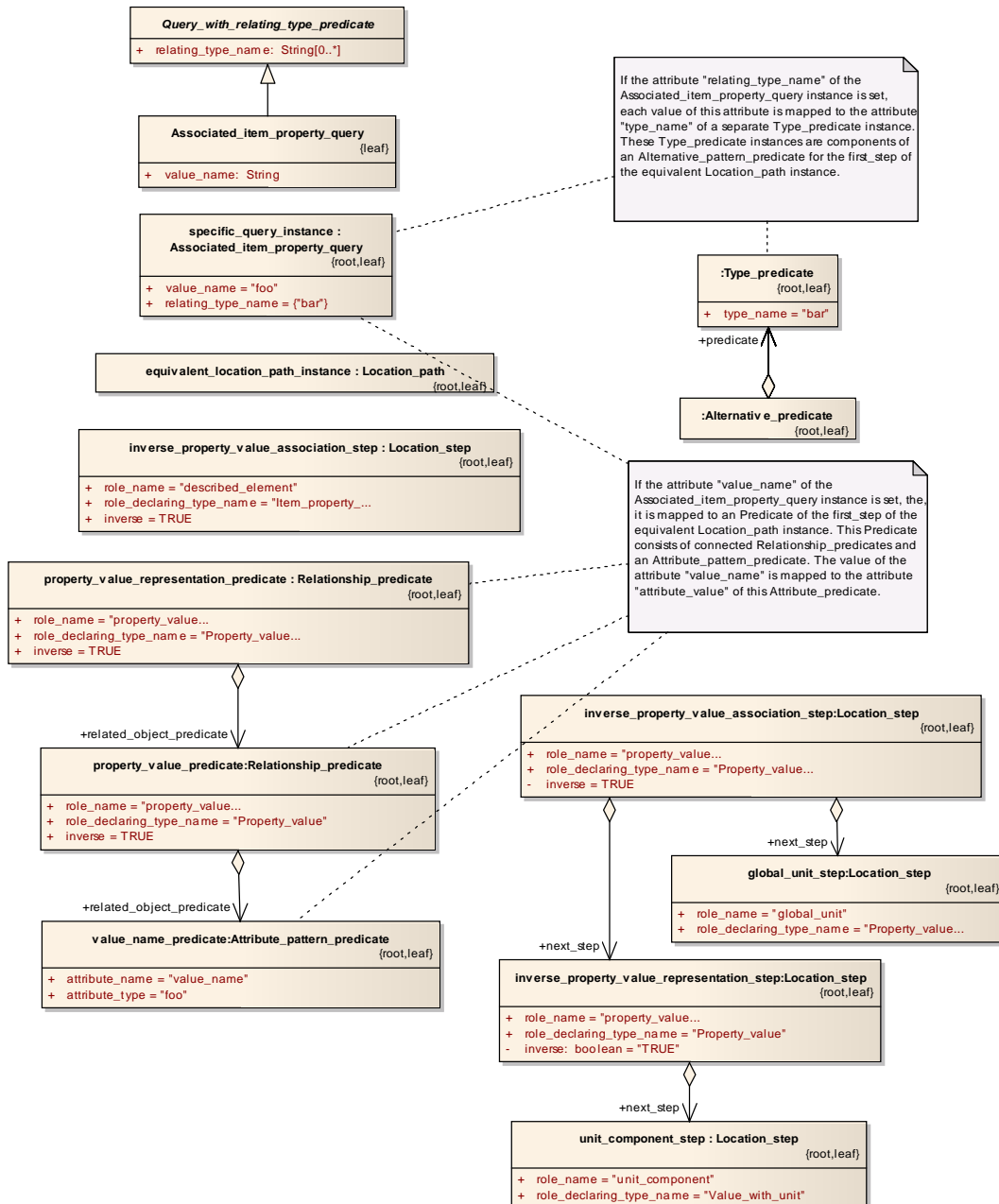


Figure 9.37 - Definition, sample instance and equivalent Location\_path instance of the Associated\_item\_property\_query



### 9.7.22 Associated\_person\_organization\_query

The Associated\_person\_organization\_query traverses from Date\_time\_person\_organization\_element\_select objects via Person\_organization\_assignment objects to Person\_organization\_select objects.

#### Base Class

- Query\_with\_relating\_type\_predicate

#### Parameters

- role : String [0..1]
- relating\_type\_names : String [0..\*]

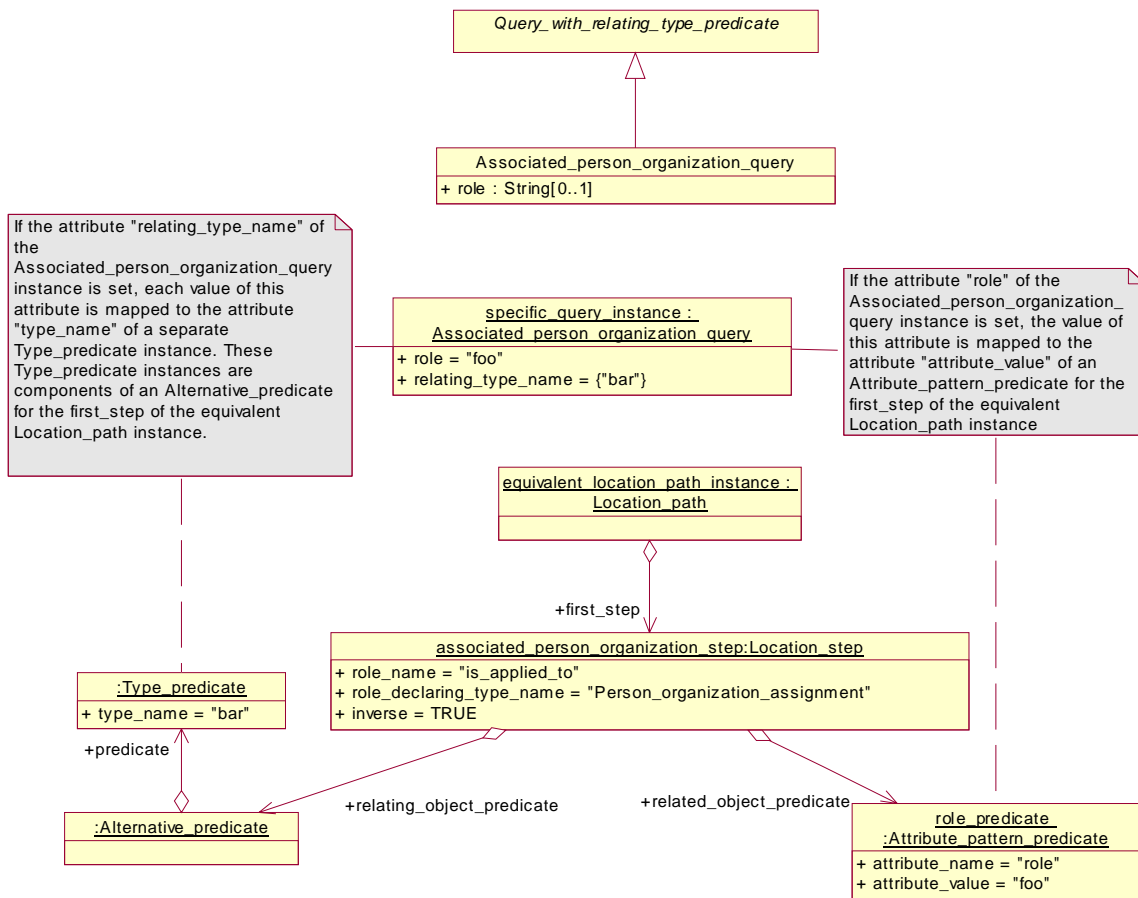


Figure 9.38 - Definition, sample instance and equivalent Location\_path instance of the Associated\_person\_organization\_query

### 9.7.23 Associated\_process\_property\_query

The Associated\_process\_property\_query traverses from Process\_property\_select objects via Property\_value\_association objects and Property\_value\_representation objects to the associated Property\_value objects.

#### **Base Class**

- Query\_with\_relating\_type\_predicate

#### **Parameters**

- value\_name : String [0..1]
- relating\_type\_name : String [0..\*]

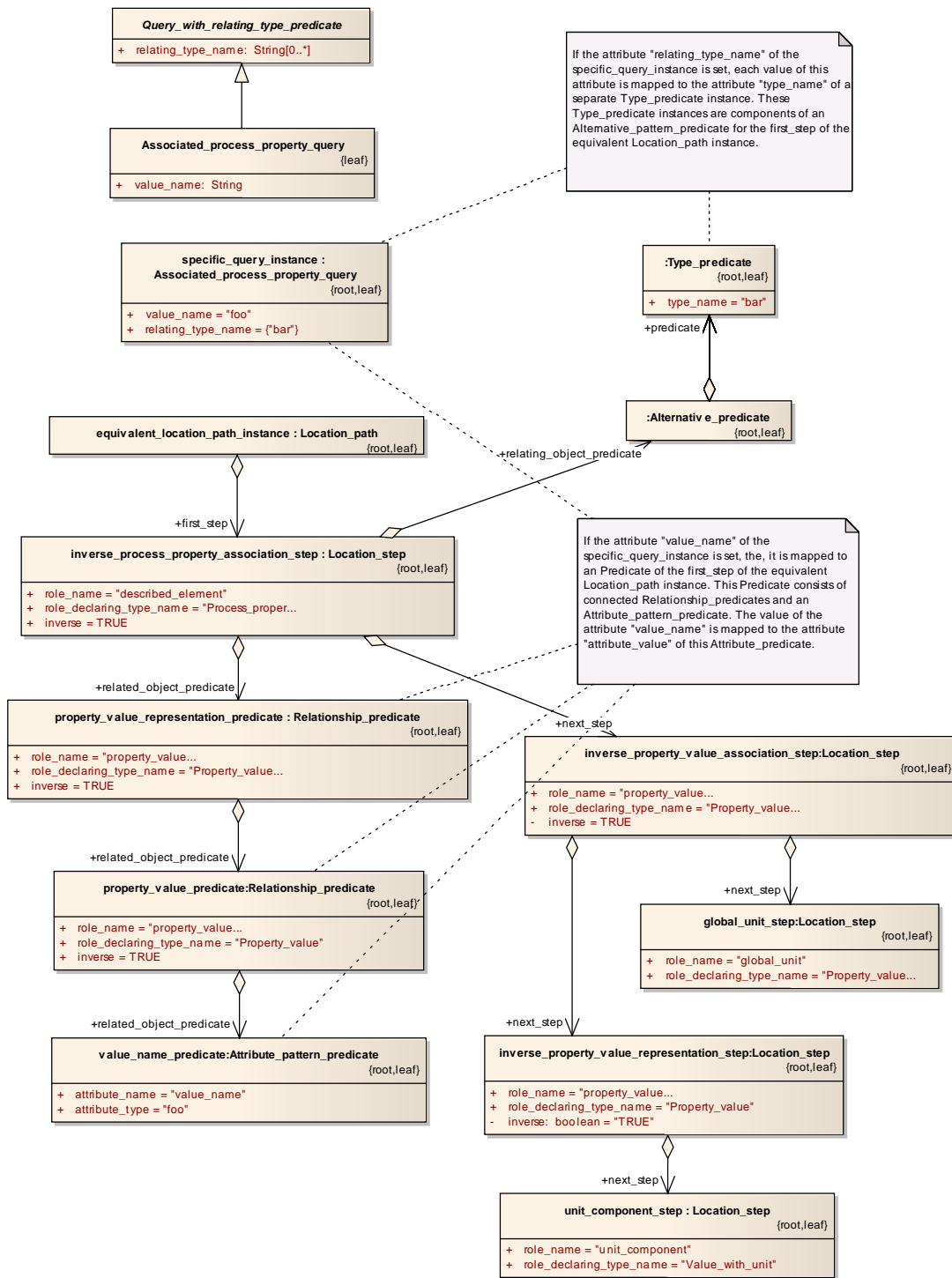


Figure 9.39 - Definition, sample instance and equivalent Location\_path instance of the Associated\_process\_property\_query

## 9.7.24 Associated\_project\_query

The Associated\_project\_query traverses from Project\_information\_select objects via Project\_assignment objects to Project objects.

### Base Class

- Query\_with\_relating\_type\_predicate

### Parameters

- role : String [0..1]
- relating\_type\_name : String [0..\*]

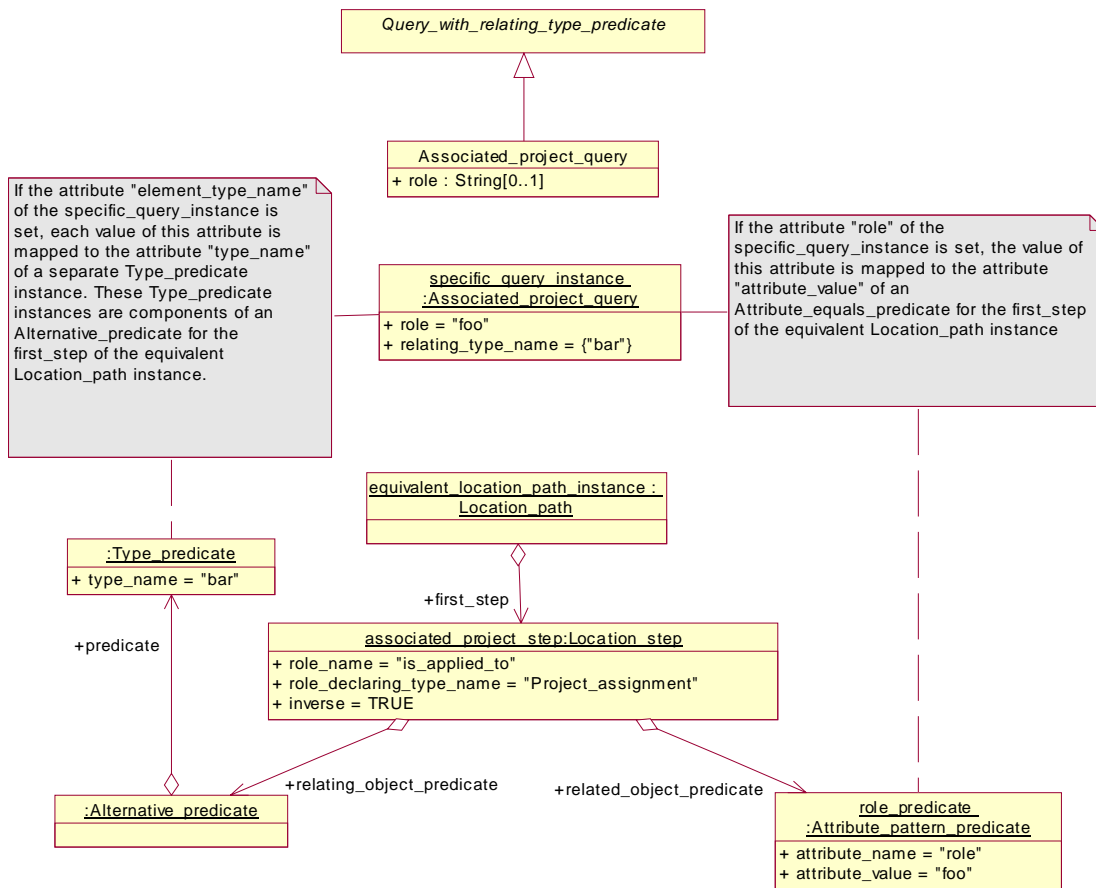


Figure 9.40 - Definition, sample instance and equivalent Location\_path instance of the Associated\_project\_query

### 9.7.25 Associated\_property\_query

The Associated\_property\_query traverses from Item\_property\_select and Process\_property\_select objects via Property\_value\_association objects and Property\_value\_representation objects to the associated Property\_value objects.

The Associated\_property\_query is defined as a Batch\_query of an Associated\_item\_property\_query and an Associated\_process\_property\_query.

#### Base Class

- Query\_with\_relating\_type\_predicate

#### Parameters

- value\_name : String [0..1]
- relating\_type\_name : String [0..\*]

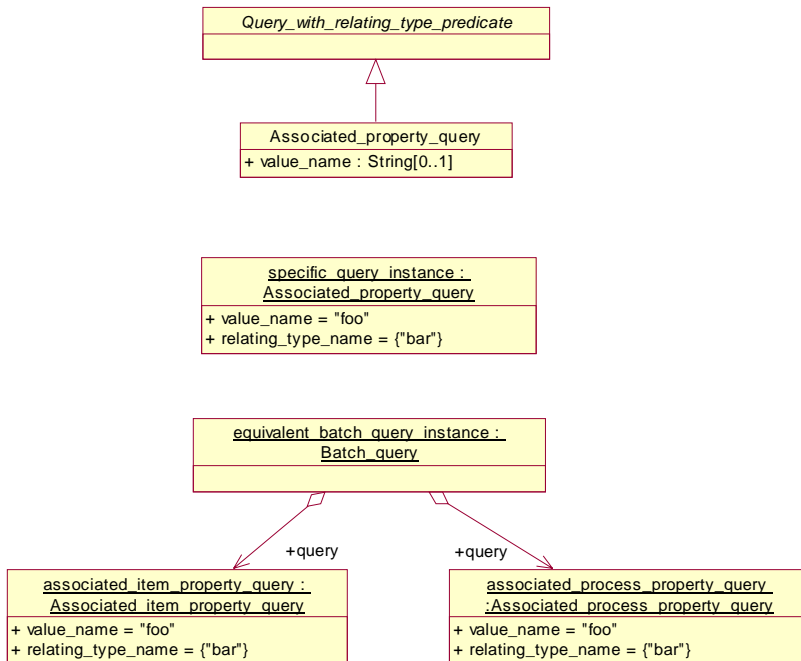


Figure 9.41 - Definition, sample instance and equivalent Location\_path instance of the Associated\_property\_query

### 9.7.26 Class\_structure\_query

The Class\_structure\_query traverses from Product\_class objects via Class\_structure\_relationship objects to Product\_function\_component\_select objects.

**Base Class**

- Specific\_query

**Parameters**

- relation\_type : String [0..1]

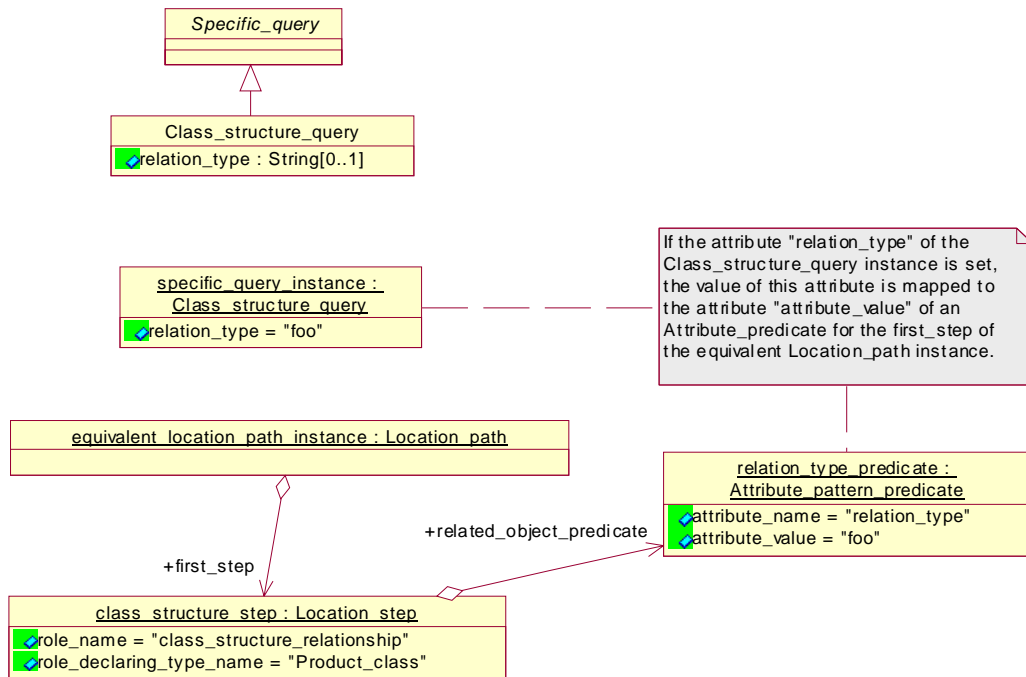


Figure 9.42 - Definition, sample instance and equivalent Location\_path instance of the Class\_structure\_query

**9.7.27 Complex\_product\_query**

The Complex\_product\_query selects Complex\_product objects by its id and version\_id attributes.

**Base Class**

- Specific\_query

**Parameters**

- id : String [0..1]
- id\_scope : String [0..1]
- version\_id : String [0..1]

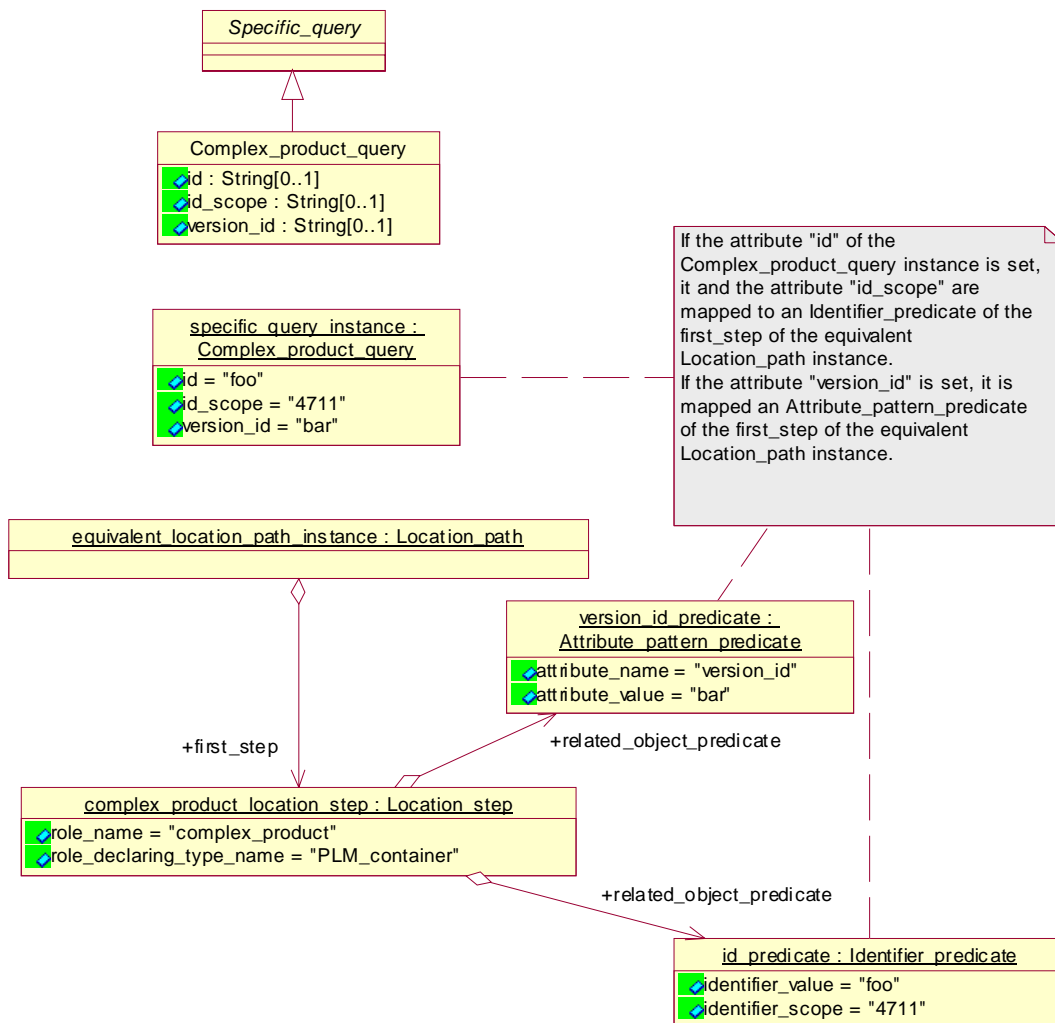


Figure 9.43 - Definition, sample instance and equivalent Location\_path instance of the Complex\_product\_query

### 9.7.28 Configuration\_query

The Configuration\_query traverses from Configured\_item\_select objects via Configuration objects to Configured\_specification\_select objects.

#### Base Class

- Query\_with\_relating\_type\_predicate

#### Parameters

- configuration\_type : String [0..1]
- inheritance\_type : String [0..1]
- relating\_type\_name : String [0..\*]

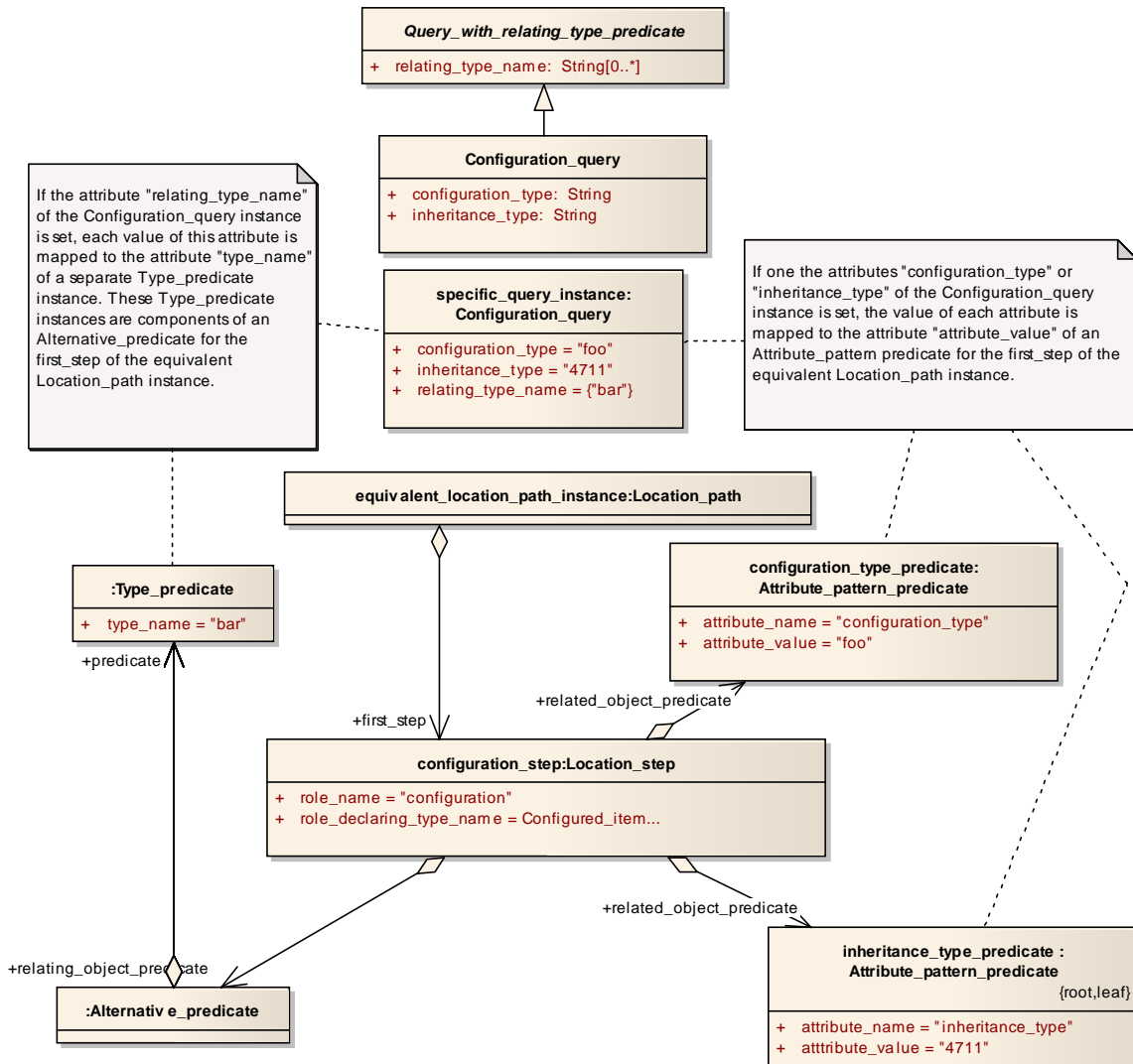


Figure 9.44 - Definition, sample instance and equivalent Location\_path instance of the Configuration\_query

### 9.7.29 Date\_organization\_assignment\_query

The Date\_organization\_assignment\_query traverses from Date\_and\_person\_organization objects via Date\_and\_person\_assignment objects to Date\_time\_person\_organization\_element\_select objects.

#### Parameters

- role : String [0..1]
- related\_type\_name : String [0..\*]



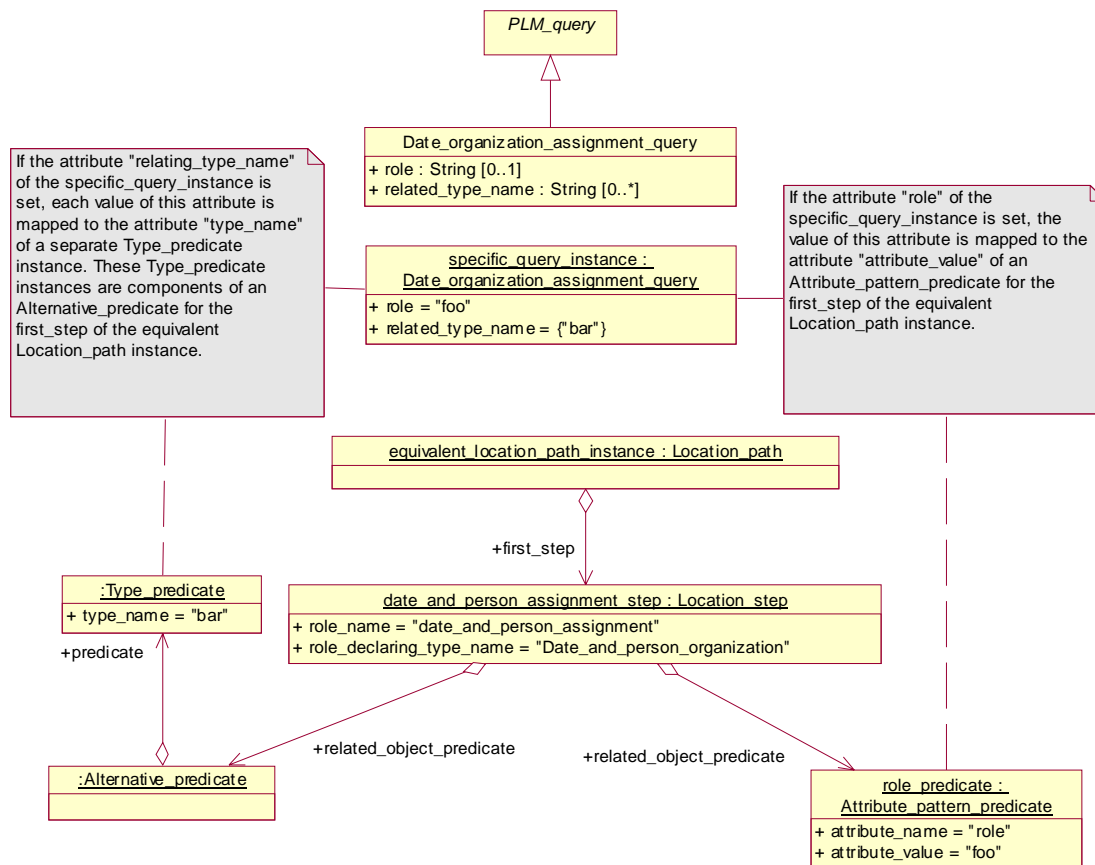


Figure 9.45 - Definition, sample instance and equivalent Location\_path instance of the Date\_organization\_assignment\_query

### 9.7.30 Design\_discipline\_item\_definition\_query

The Design\_discipline\_item\_definition\_query traverses from Item\_version objects to Design\_discipline\_item\_definition objects.

#### Parameters

- id : String [0..1]
- application\_domain : String [0..1]  
Traverse only Design\_discipline\_item\_definition objects, which relates via their initial\_context association to an Application\_context object with an application\_domain attribute of the given value.
- life\_cycle\_stage : String [0..1]  
Traverse only Design\_discipline\_item\_definition objects, which relates via their initial\_context association to an Application\_context object with an life\_cycle\_stage attribute of the given value.

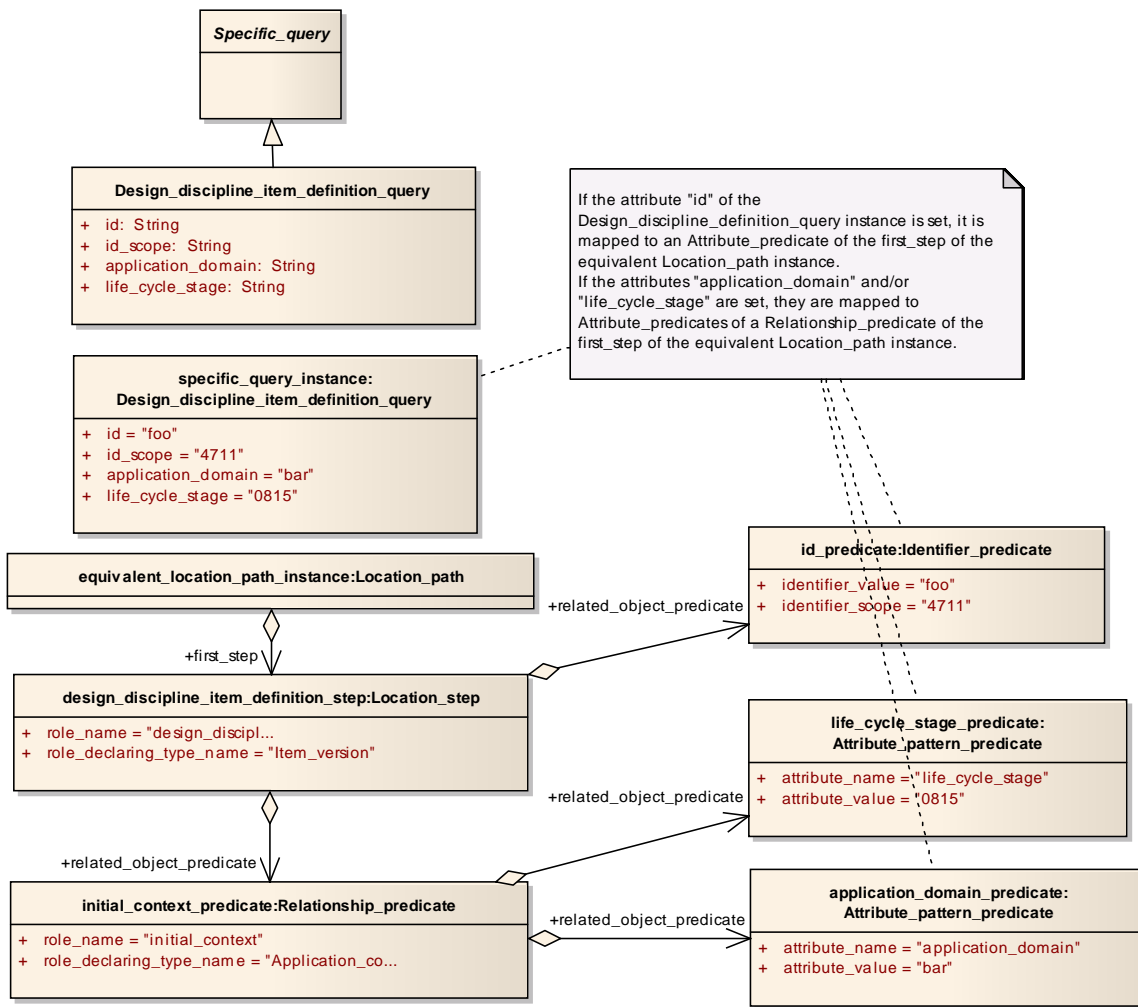


Figure 9.46 - Definition, sample instance and equivalent `Location_path` instance of the `Design_discipline_item_definition_query`

### 9.7.31 Document\_classification\_query

The `Document_classification_query` traverses from `Document` objects to `Specific_document_classification` objects.

#### Parameters

- `classification_name` : `String` [0..1]

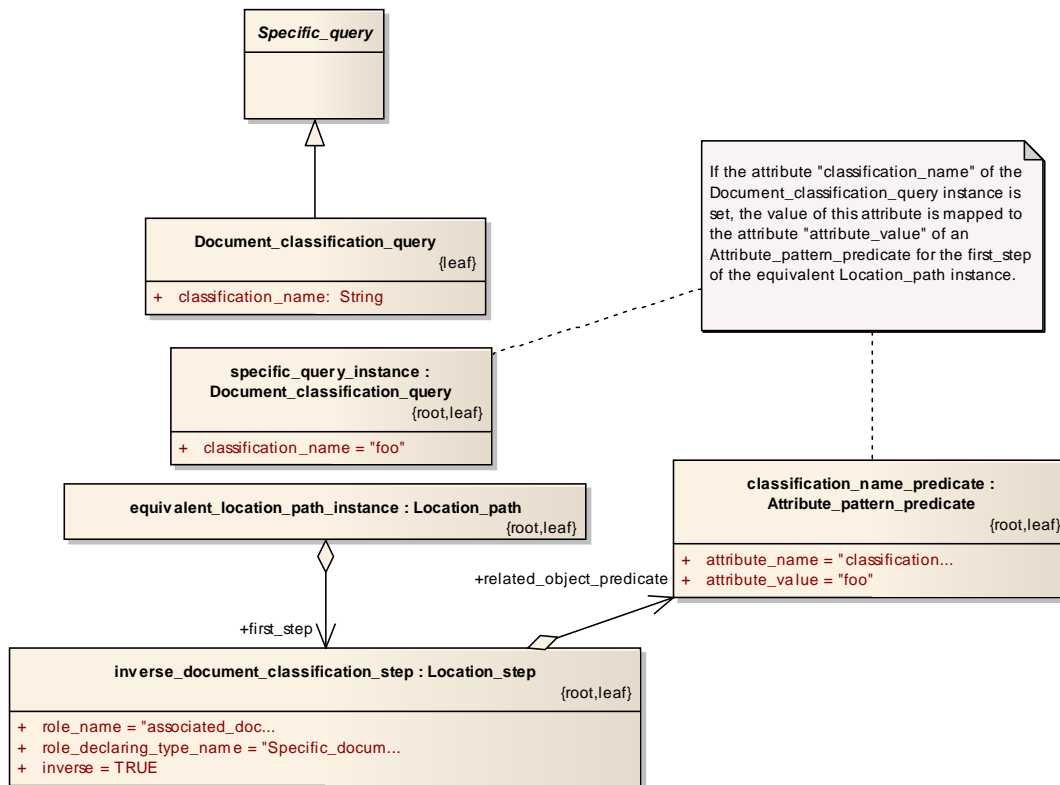


Figure 9.47 - Definition, sample instance and equivalent Location\_path instance of the Document\_classification\_query

### 9.7.32 Document\_classification\_hierarchy\_query

The `Document_classification_hierarchy_query` traverses from `Specific_document_classification` objects to `Specific_document_classification` objects via `Specific_document_classification_hierarchy` objects.

#### Base Class

- `Specific_query`

#### Parameters

- `inverse`: boolean [0..1]

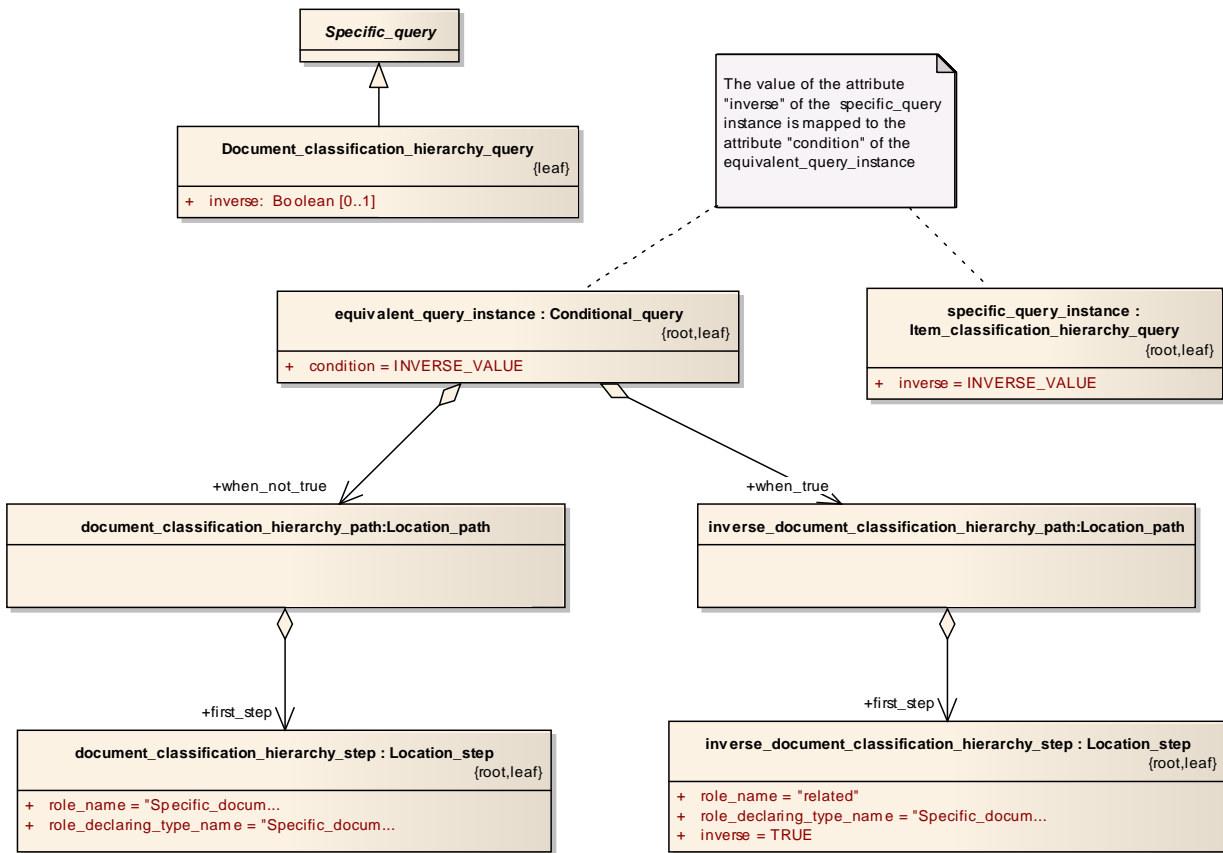


Figure 9.48 - Definition, sample instance and equivalent Location\_path instance of the Document\_classification\_hierarchy\_query

### 9.7.33 Document\_property\_query

The `Document_property_query` traverses the document properties from `Document_representation` objects.

These properties are `Document_creation_property`, `Document_size_property`, `Document_format_property`, `Document_content_property`, and `Document_location_property`.

#### Base Class

- `Specific_query`

#### Parameters

- none

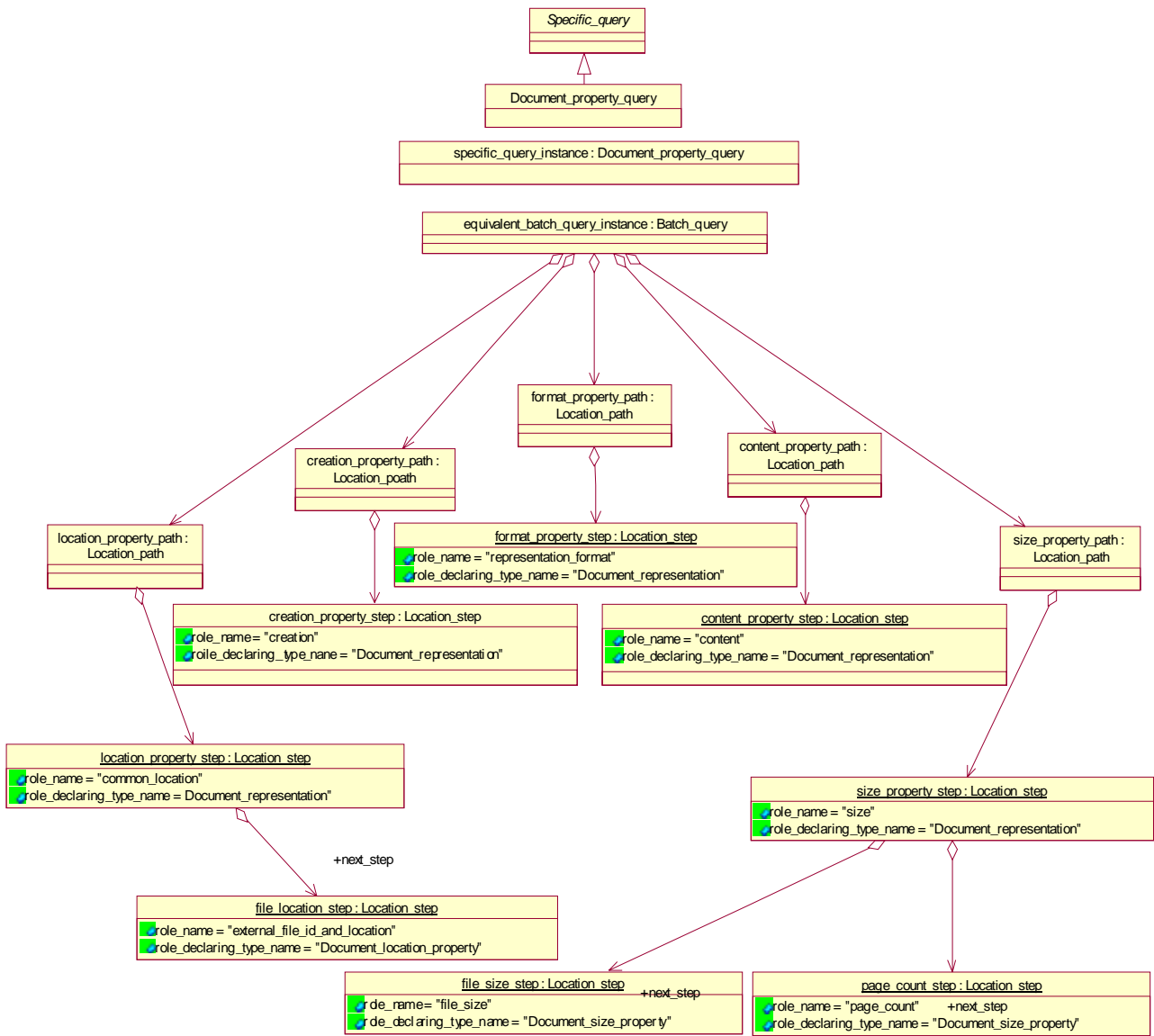


Figure 9.49 - Definition, sample instance and equivalent Location\_path instance of the Document\_property\_query

### 9.7.34 Document\_query

The Document\_query selects Document objects.

#### Parameters

- document\_id : String [0..1]
- document\_id\_scope : String [0..1]

- name : String [0..1]
- name\_language : Language [0..1]
- version\_id : String [0..1]
- classification\_name : String [0..1]

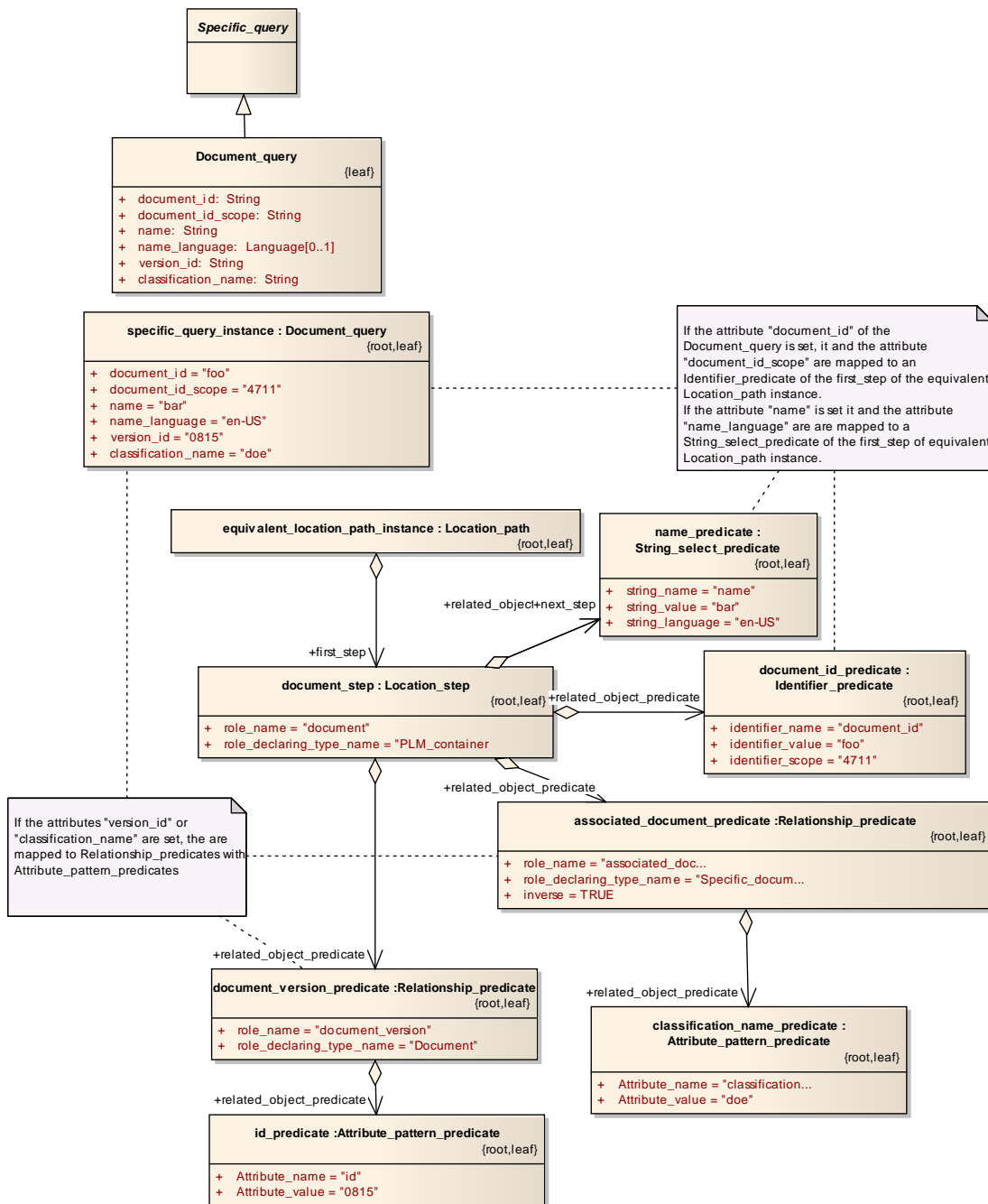


Figure 9.50 - Definition, sample instance and equivalent Location\_path instance of the Document\_query

### 9.7.35 Document\_representation\_query

The Document\_representation\_query traverses Document\_representation objects from Document\_version objects.

#### Parameters

- id : String [0..1]
- id\_scope : String [0..1]

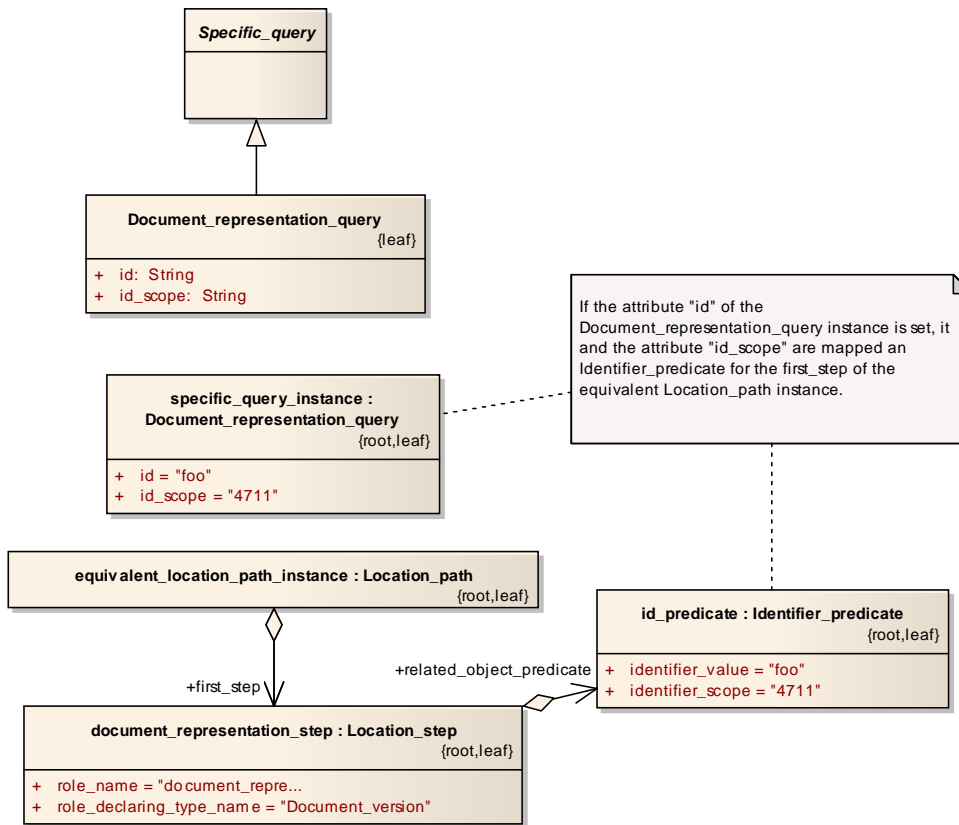


Figure 9.51 - Definition, sample instance and equivalent Location\_path instance of the Document\_representation\_query

### 9.7.36 Document\_structure\_query

The Document\_structure\_query traverses from Document\_representation objects via Document\_structure objects to Document\_representation objects.

#### Base Class

- Relationship\_query

**Parameters**

- relation\_type : String [0..1]
- inverse : Boolean [0..1]

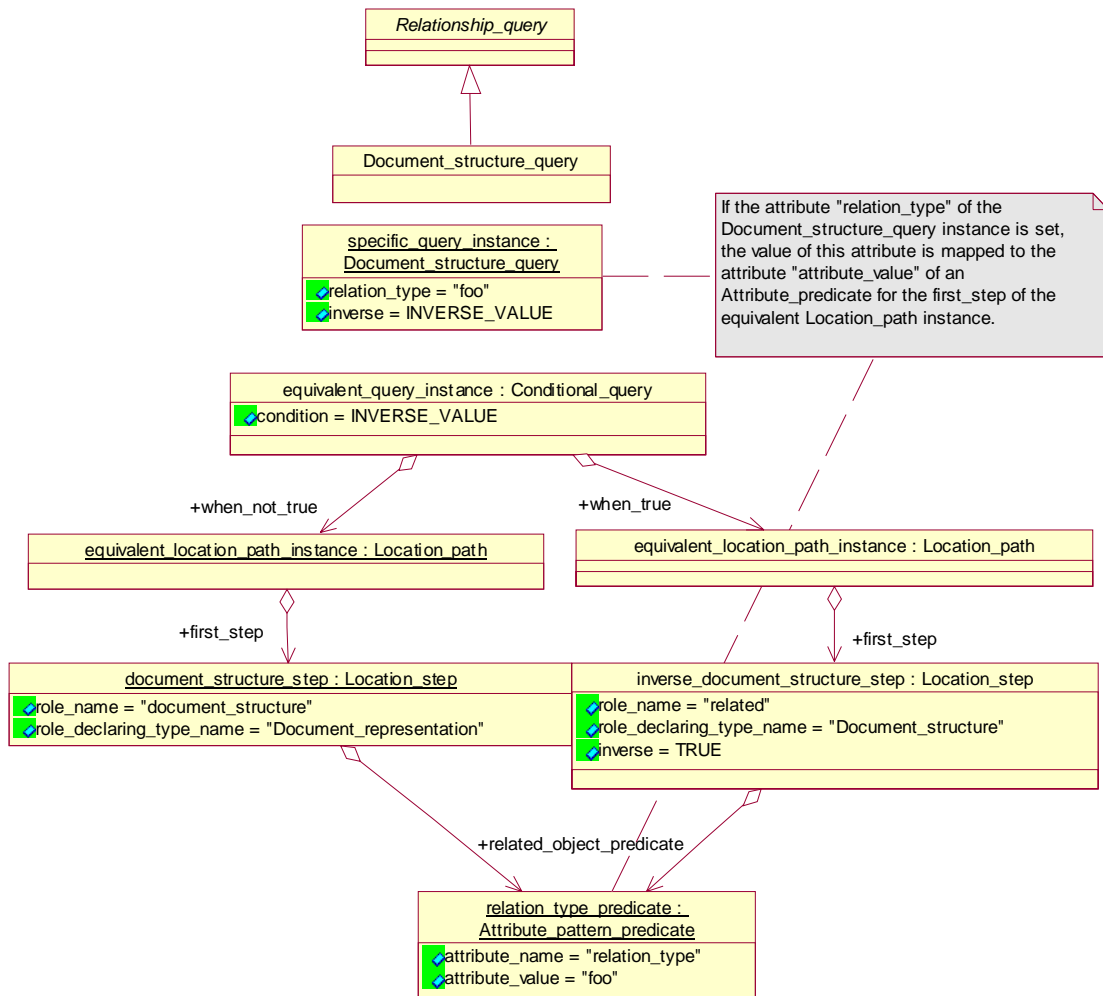


Figure 9.52 - Definition, sample instance and equivalent Location\_path instance of the Document\_structure\_query

**9.7.37 Document\_version\_query**

The Document\_version\_query traverses Document\_version objects of Document objects.

**Parameters**

- id : String [0..1]
- id\_scope : String [0..1]



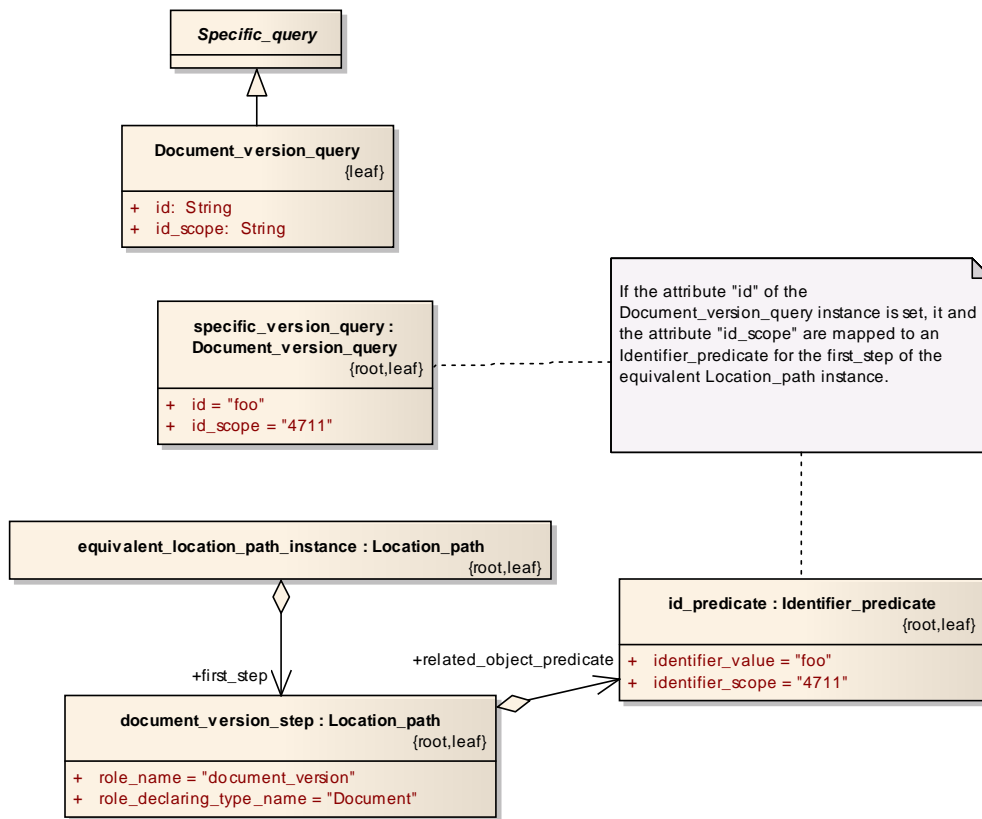


Figure 9.53 - Definition, sample instance and equivalent Location\_path instance of the Document\_version\_query

### 9.7.38 Effectivity\_assignment\_query

The Effectivity\_assignment\_query traverses from Effectivity objects to Effectivity\_element\_select objects via Effectivity\_assignment objects.

#### Base Class

- Specific\_query

#### Parameters

- Role: string [0..1]
- Effectivity\_indication: boolean [0..1]

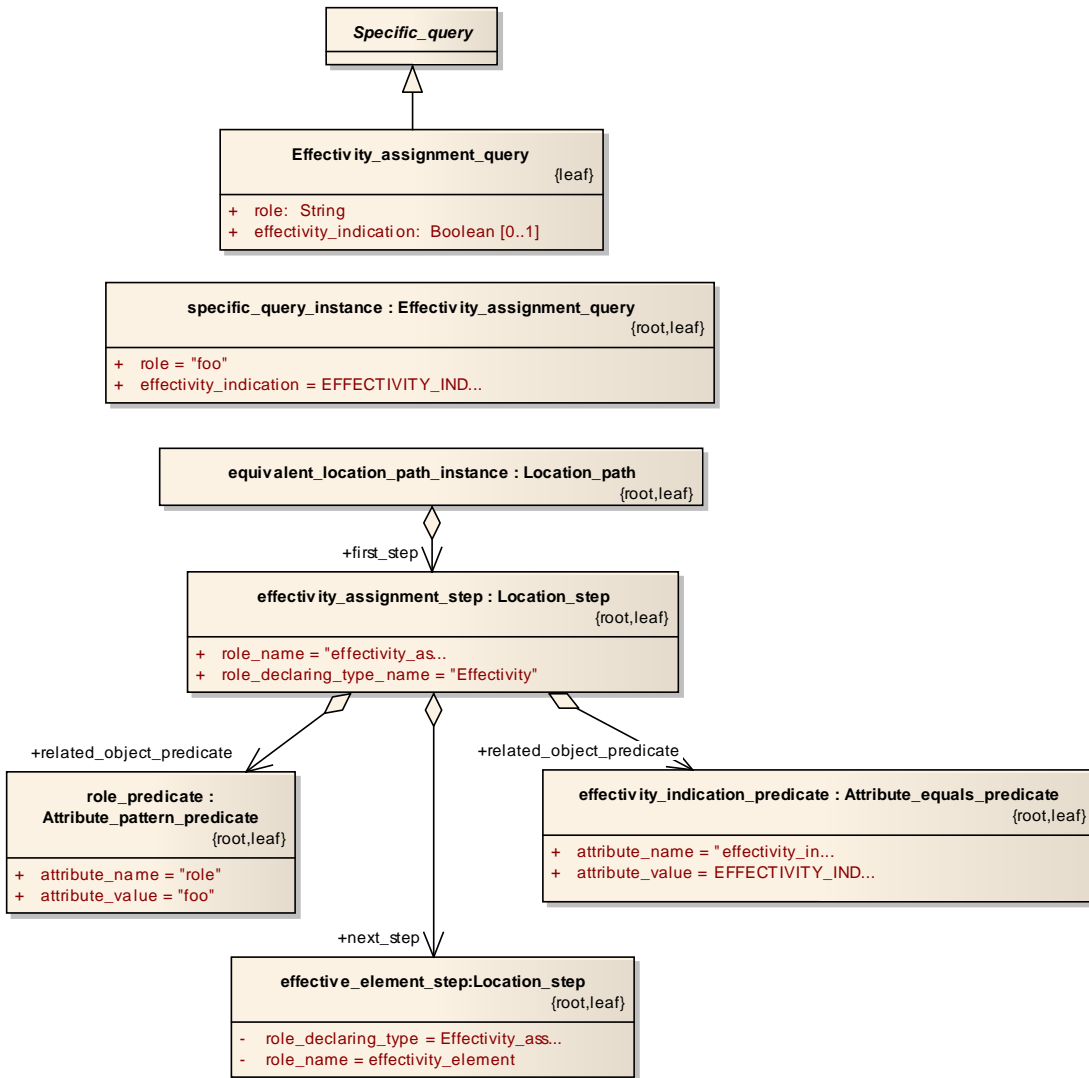


Figure 9.54 - Definition, sample instance and equivalent Location\_path instance of the Effectivity\_assignment\_query

### 9.7.39 Effectivity\_query

The Effectivity\_query selects Effectivity objects with additional related instances of Duration, Organization, and Event\_or\_date\_selected.

#### Parameters

- id : String [0..1]
- Id\_scope : String [0..1]
- version\_id : String [0..1]
- effectivity\_context : String [0..1]
- add\_period : Date\_time [0..1]

- add\_start\_definiton : Date\_time [0..1]
- add\_end\_definition : Date\_time [0..1]

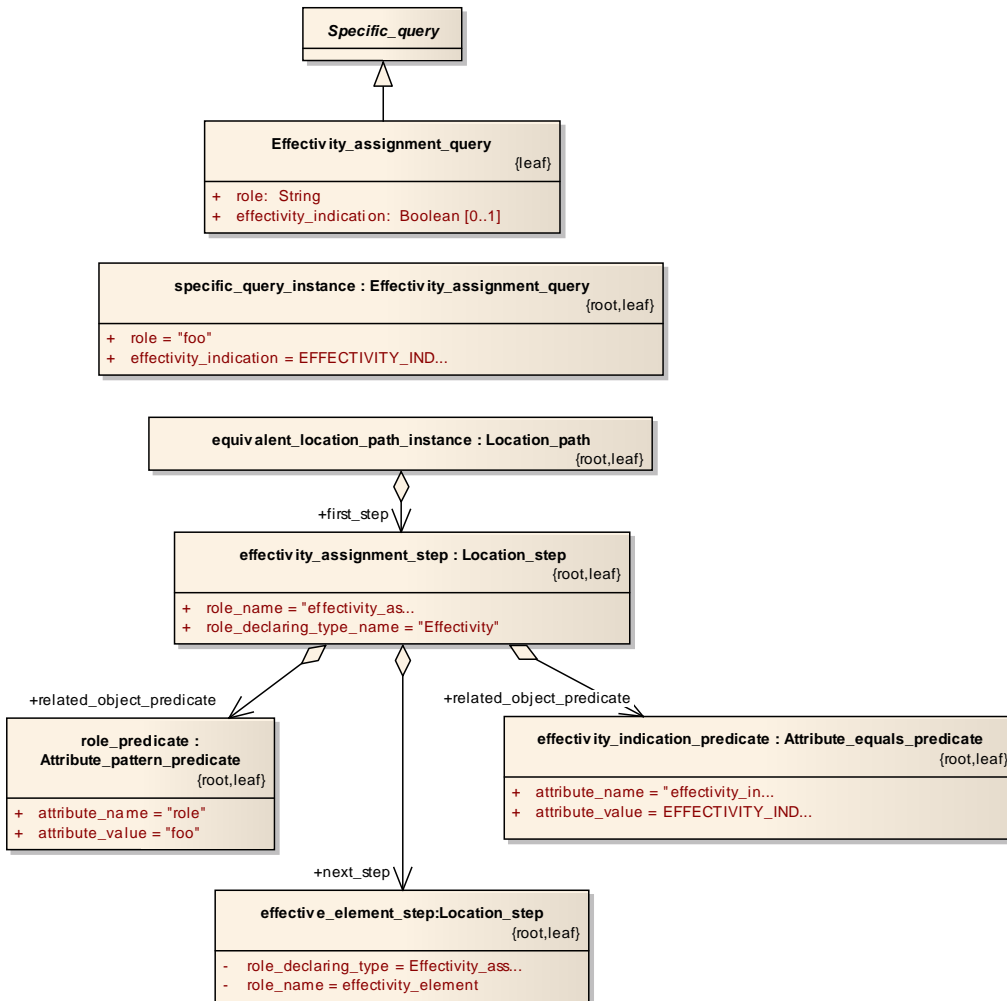


Figure 9.55 - Definition, sample instance and equivalent Location\_path instance of the Effectivity\_query

### 9.7.40 Event\_reference\_query

The Event\_reference\_query selects Event\_reference objects and traverses from the Event\_reference\_objects to their offset and event\_context references.

#### Base Class

- Specific\_query

#### Parameters

- type: string [0..1]

- add\_event\_context: boolean [0..1]
- add\_offset: boolean [0..1]

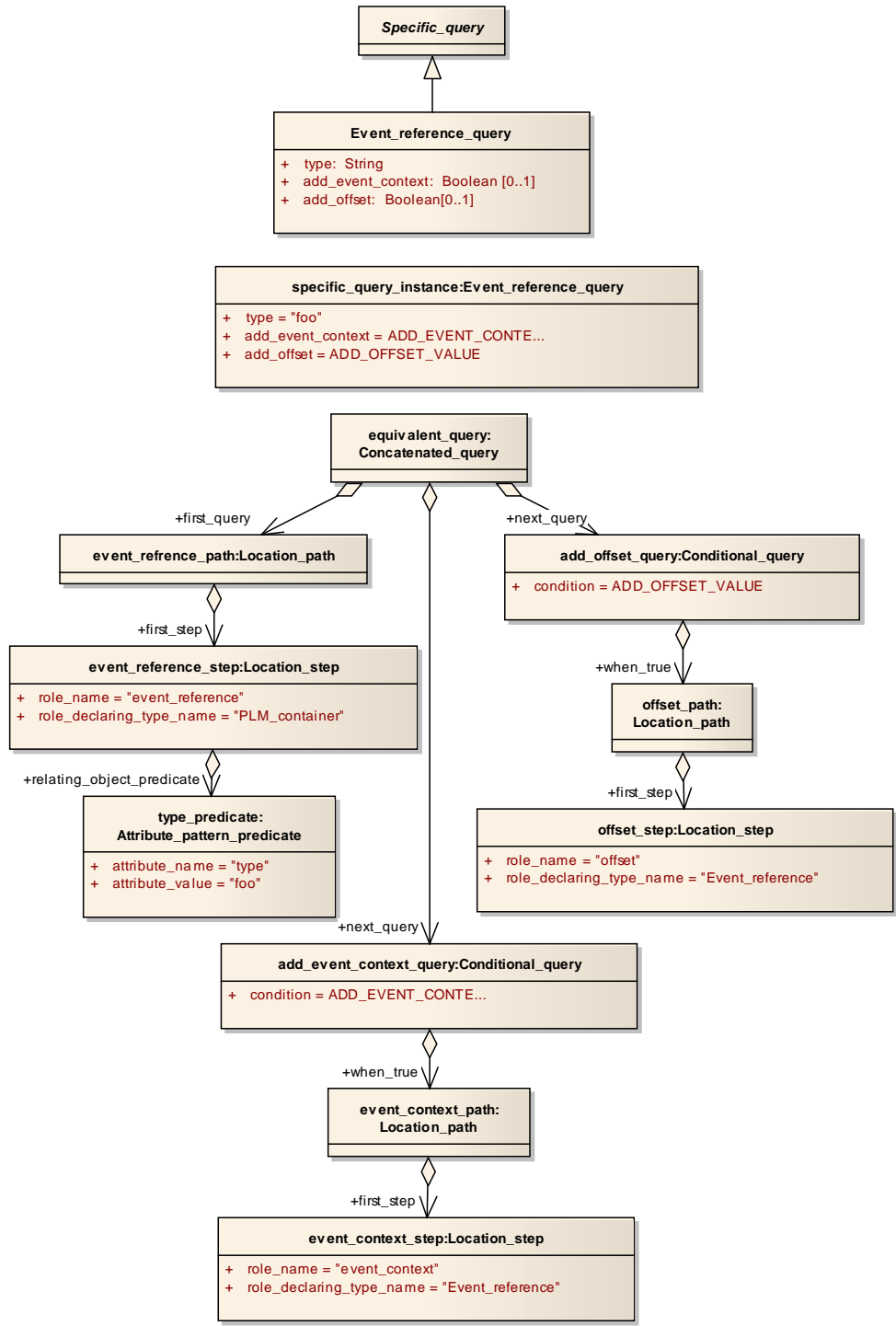


Figure 9.56 - Definition, sample instance and equivalent Location\_path instance of the Event\_reference\_query

### 9.7.41 External\_model\_query

The External\_model\_query traverses from Digital\_file objects to External\_model objects.

#### Base Class

- Specific\_query

#### Parameters

- Model\_id: string [0..1]

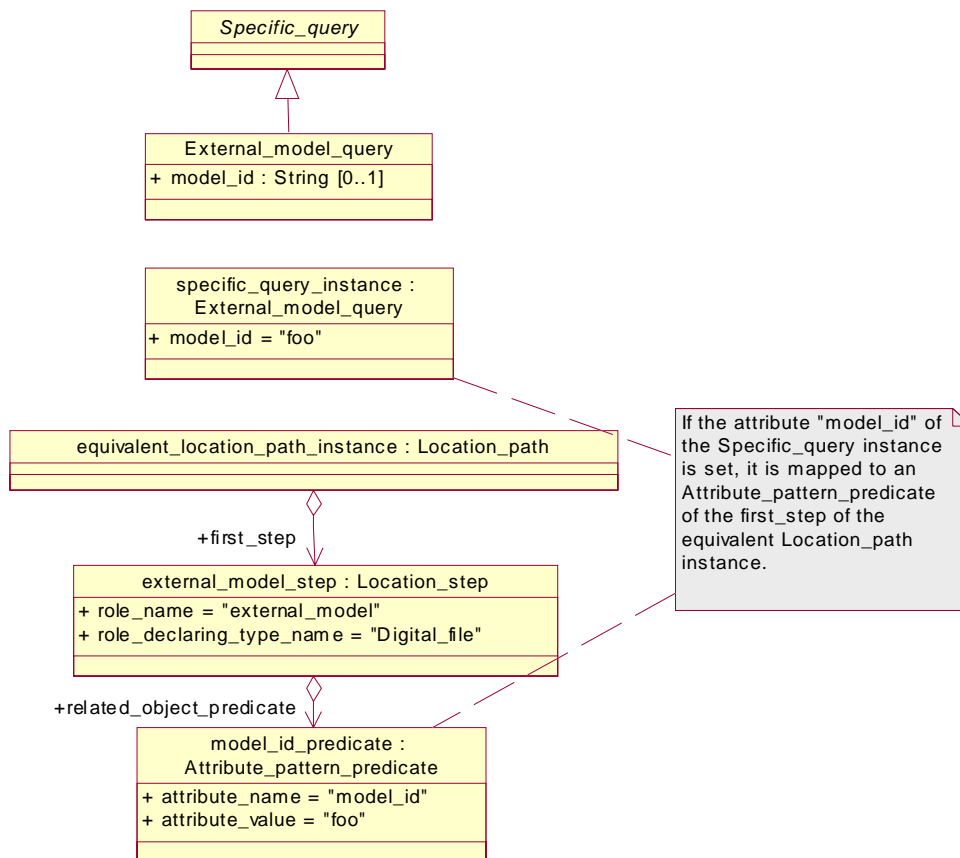


Figure 9.57 - Definition, sample instance and equivalent Location\_path instance of the External\_model\_query

### 9.7.42 File\_property\_query

The File\_property\_query traverses from Document\_file objects their file property objects.

#### Base Class

- Specific\_query

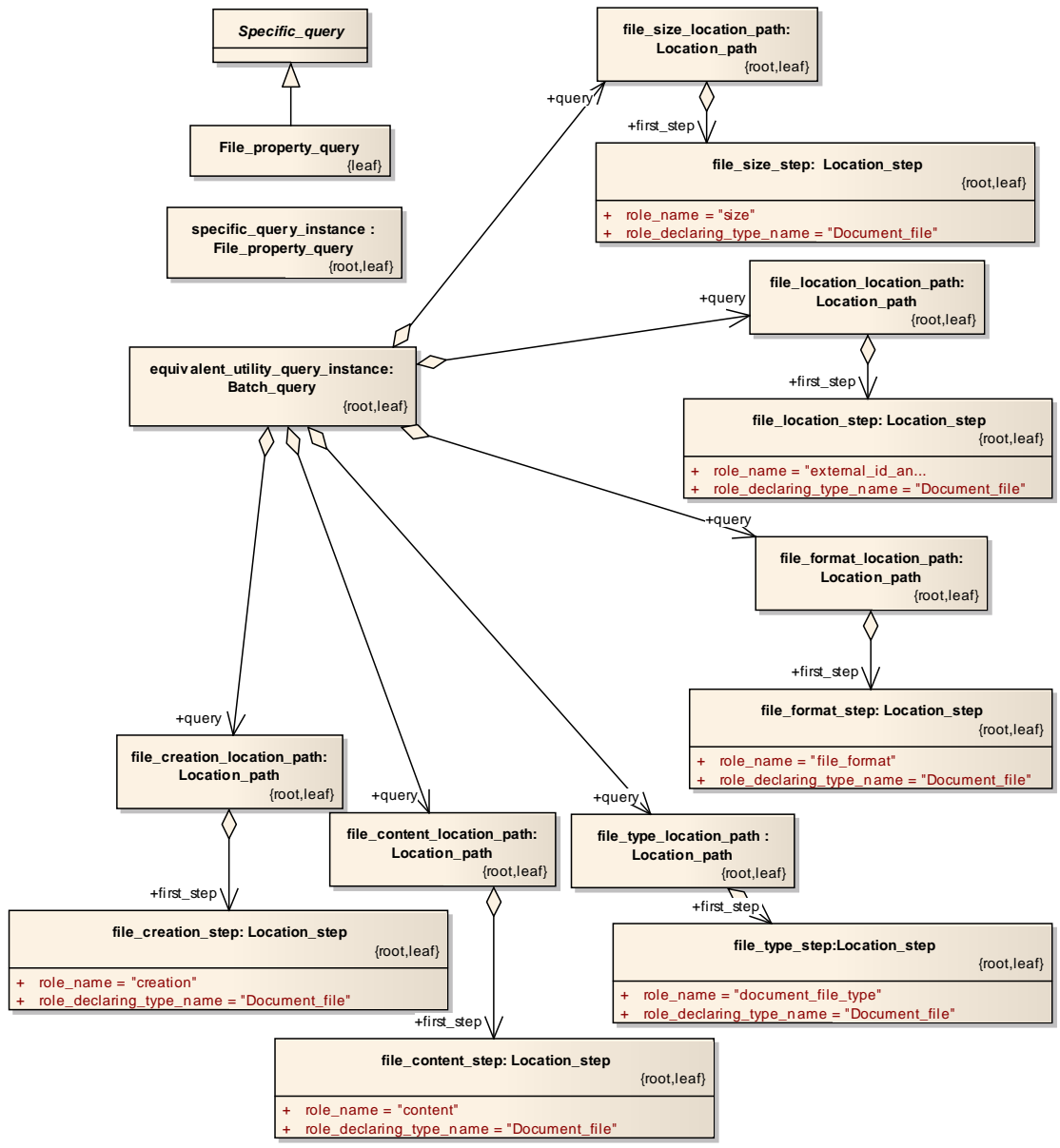


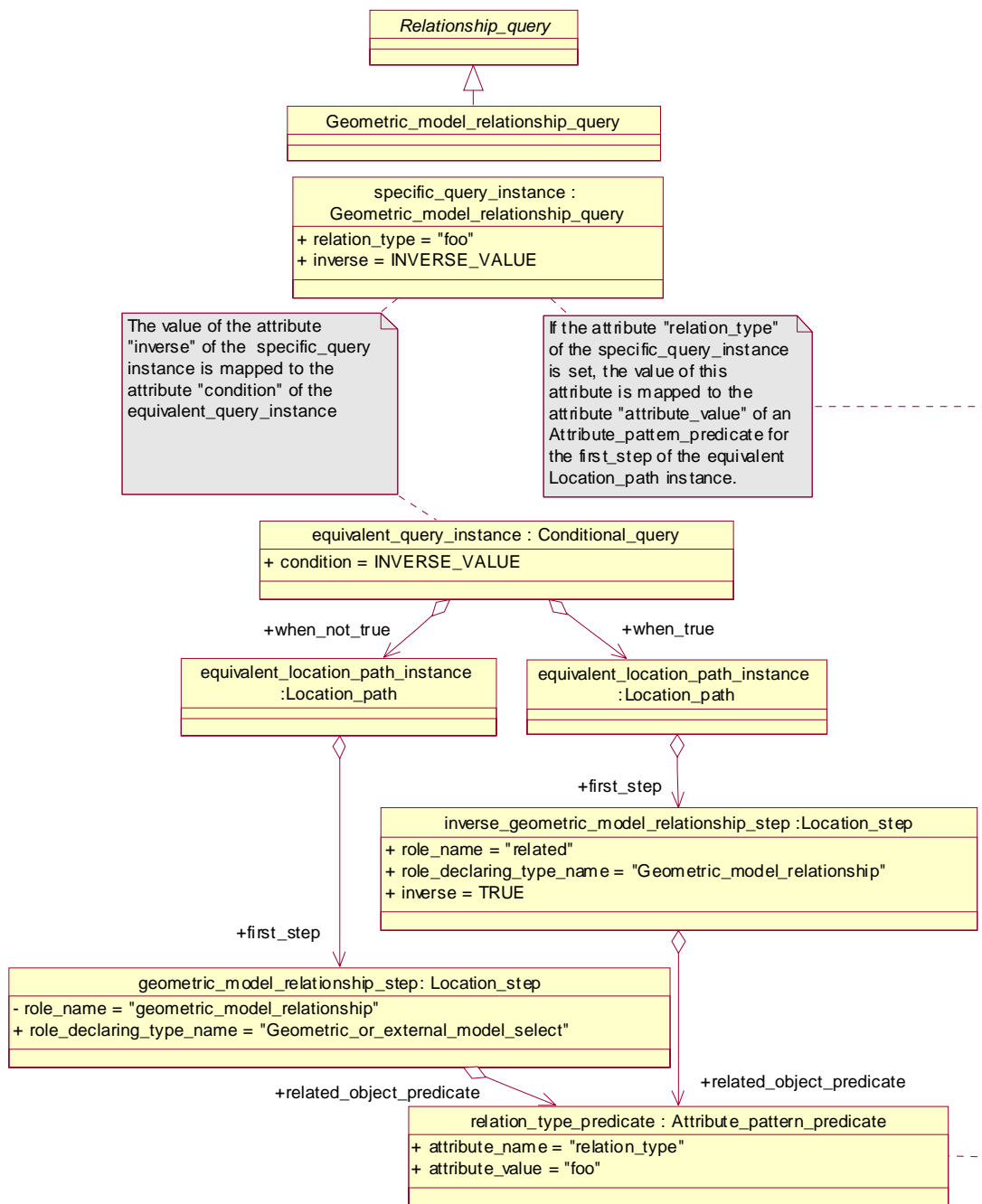
Figure 9.58 - Definition, sample instance and equivalent Location\_path instance of the File\_property\_query

### 9.7.43 Geometric\_model\_relationship\_query

The query traverses from Geometric\_or\_external\_model\_select objects to Geometric\_or\_external\_model\_select\_objects via Geometric\_model\_relationship objects.

#### Base Class

- Relationship\_query



**Figure 9.59 - Definition, sample instance and equivalent Location\_path instance of the Geometric\_model\_relationship\_query**

### 9.7.44 Item\_classification\_query

The Item\_classification\_query traverses the Specific\_item\_classification objects from Item objects.

#### Parameters

- classification\_name : String [0..1

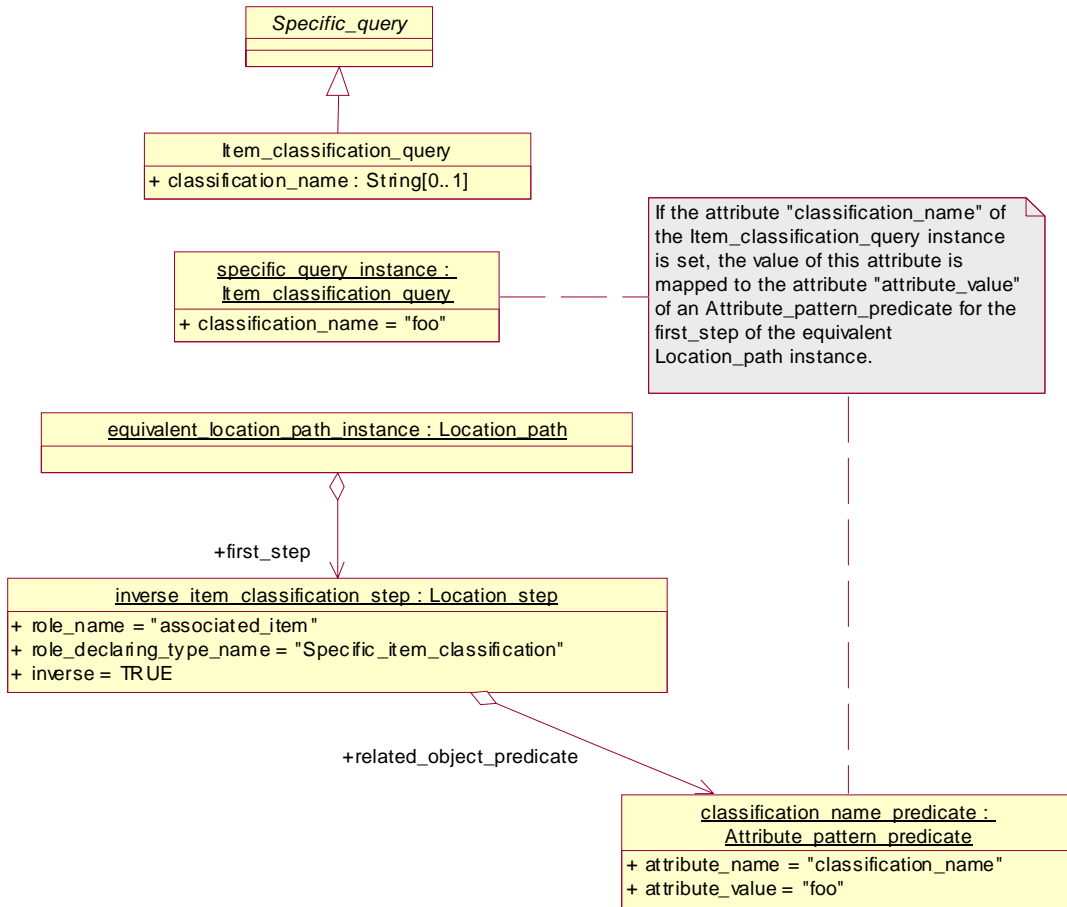


Figure 9.60 - Definition, sample instance and equivalent Location\_path instance of the Item\_classification\_query

### 9.7.45 Item\_classification\_hierarchy\_query

The Item\_classification\_hierarchy\_query traverses from Specific\_item\_classification objects to Specific\_item\_classification objects via Specific\_item\_classification\_hierarchy objects.

#### Base Class

- Specific\_query



## Parameters

- inverse: boolean [0..1]

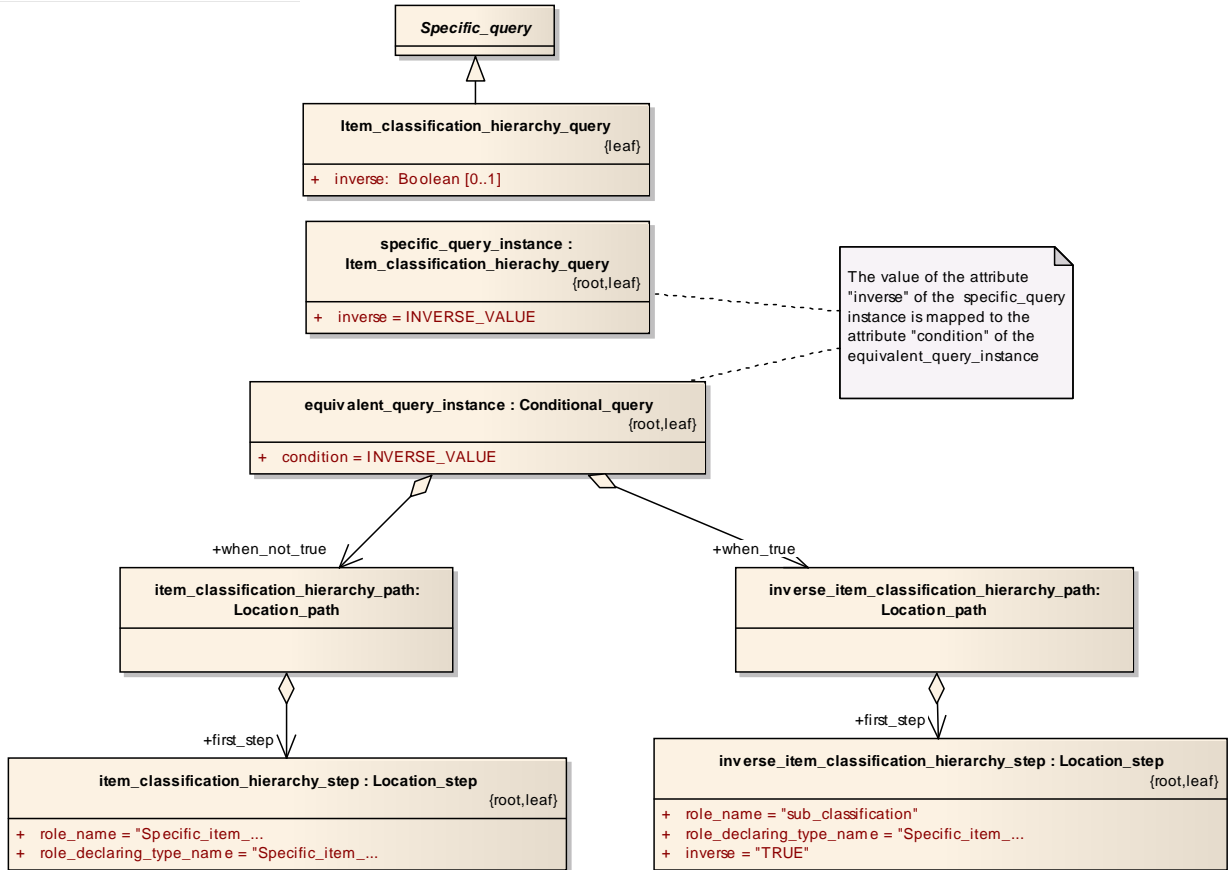


Figure 9.61 - Definition, sample instance and equivalent Location\_path instance of the Item\_classification\_hierarchy\_query

### 9.7.46 Item\_definition\_instance\_relationship\_query

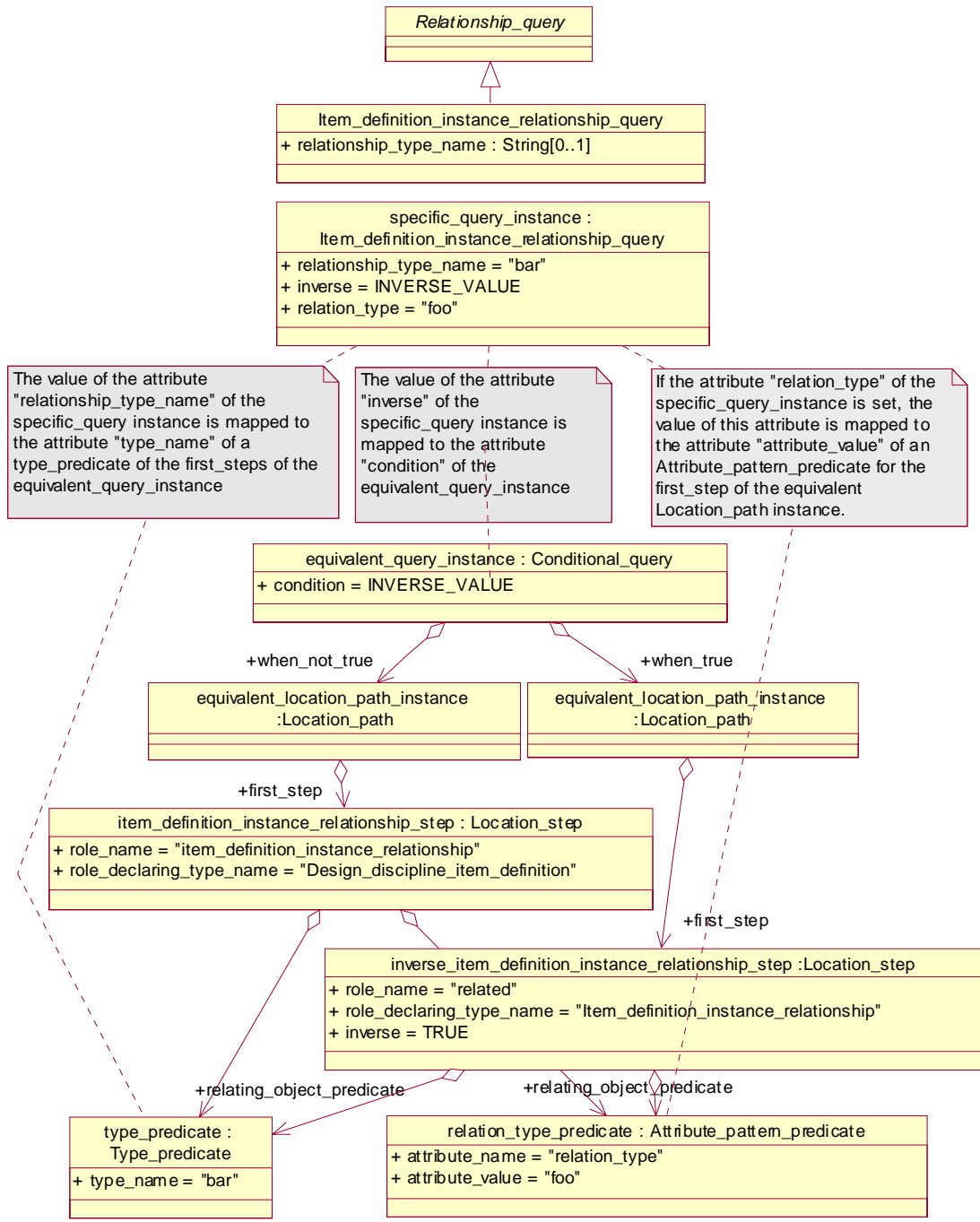
The Item\_definition\_instance\_relationship\_query traverses from Design\_discipline\_item\_definition objects to Item\_instance objects or vice versa via Item\_definition\_instance\_relationship objects.

#### Base Class

- Relationship\_query

#### Parameters

- Relationship\_type\_name: string [0..1]  
Filters the Item\_definition\_instance\_relationship objects by type (e.g., "Assembly\_component\_relationship" or "Next\_higher\_assembly").
- product\_identification\_uid: UID [0..1]  
Identifies Product\_identification object that represents a service specific algorithm to filter the Item\_version objects.



**Figure 9.62 - Definition, sample instance and equivalent Location\_path instance of the Item\_definition\_instance\_relationship\_query**

### 9.7.47 Item\_instance\_query

The Item\_instance\_query traverses from Design\_discipline\_item\_definition and Product\_identification objects to Item\_instance objects.

#### **Base Class**

- Query\_with\_relating\_type\_predicate

#### **Parameters**

- Id: string [0..1]
- Id\_scope: string [0..1]
- Name: string [0..1]
- Name\_language: language [0..1]

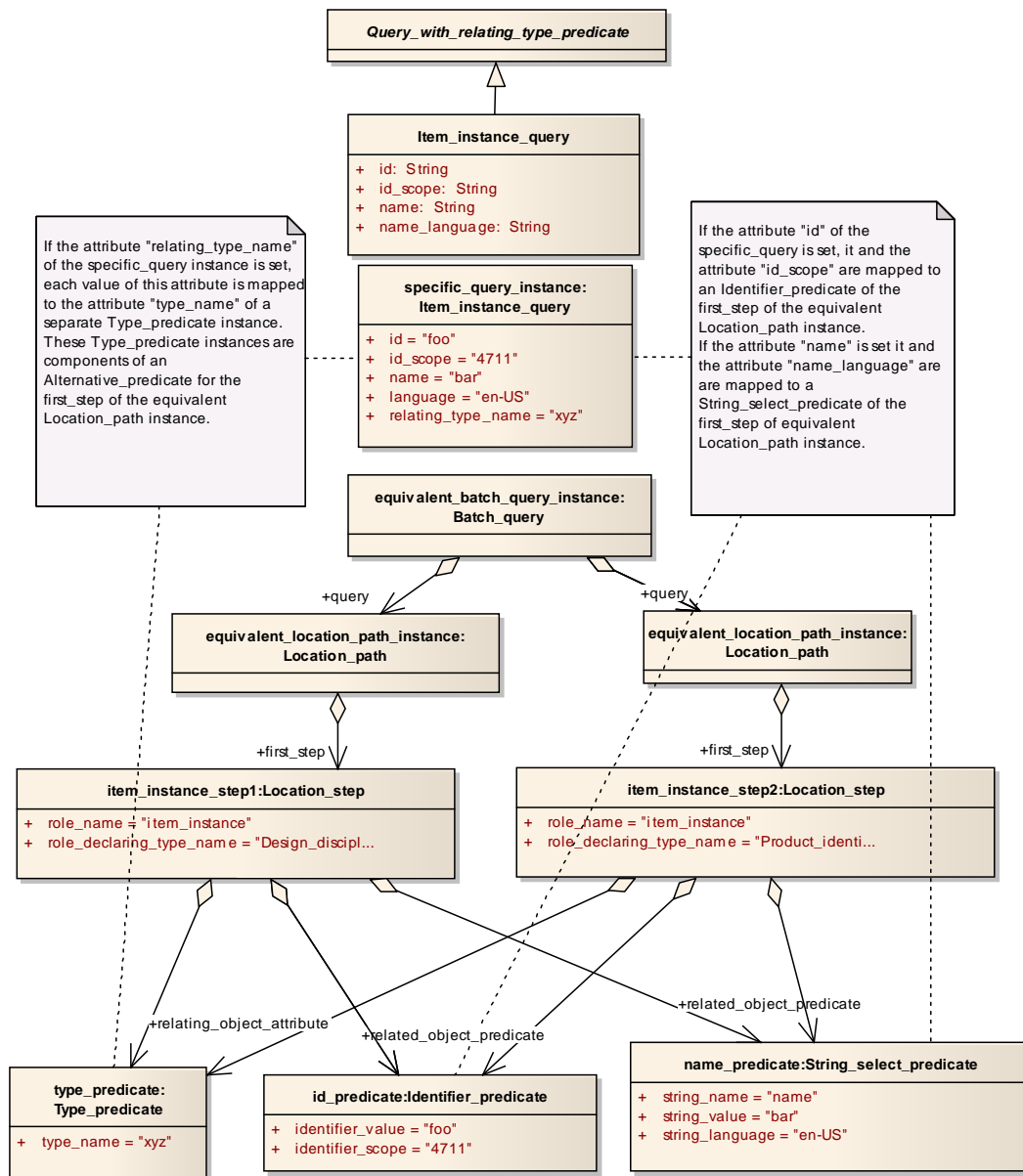


Figure 9.63 - Definition, sample instance and equivalent Location\_path instance of the Item\_instance\_query

### 9.7.48 Item\_query

The Item\_query selects Item objects.

#### Parameters

- id : String
- id\_scope : String [0..1]

- name : String
- name\_language : Language
- version\_id : String [0..1]
- classification\_name : String [0..1]

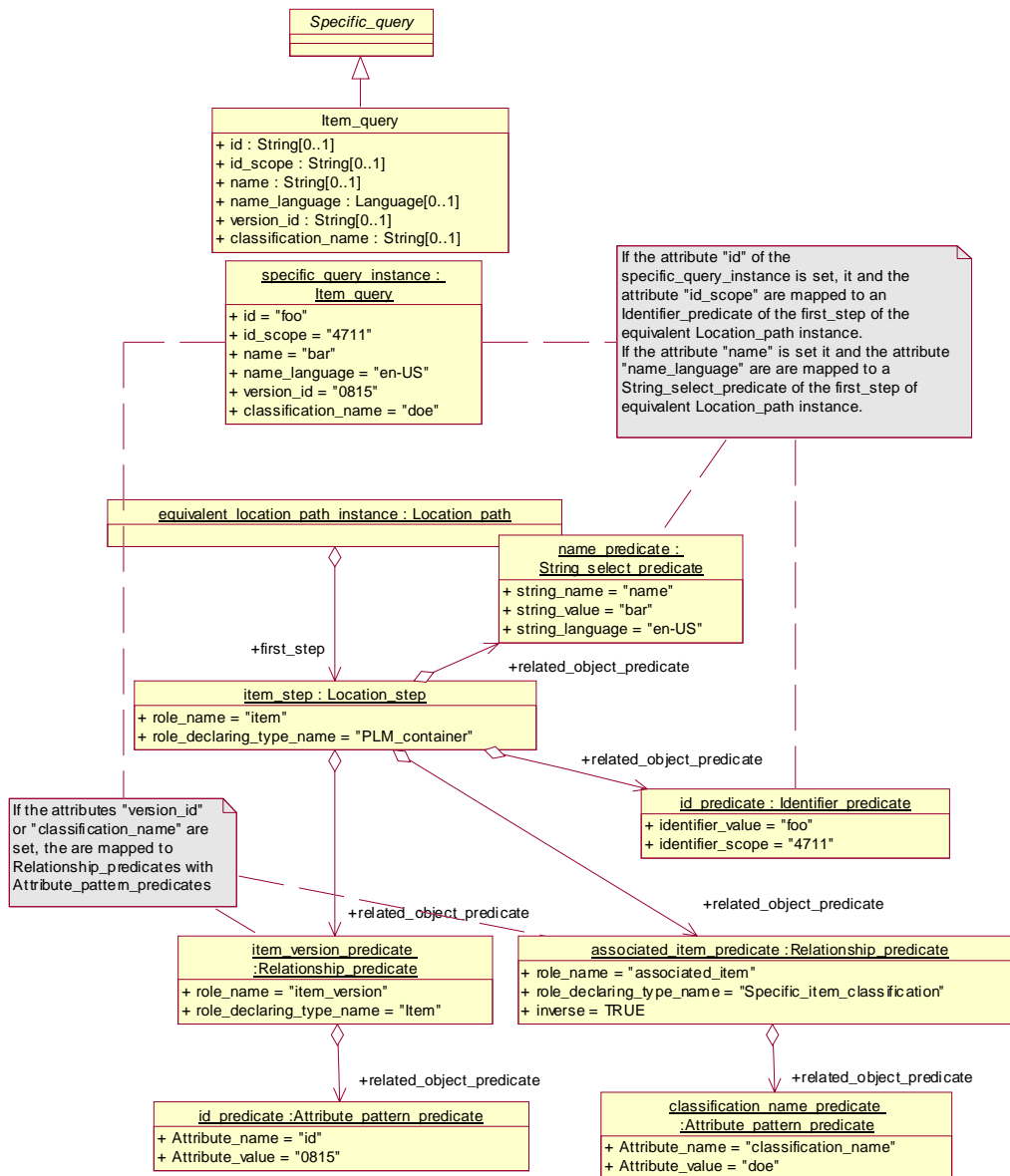


Figure 9.64 - Definition, sample instance and equivalent Location\_path instance of the Item\_query

### 9.7.49 Item\_use\_query

The Item\_use\_query traverses those assemblies from Design\_discipline\_item\_definition objects where the Design\_discipline\_item\_definition objects are used as components.

#### Parameters

- none

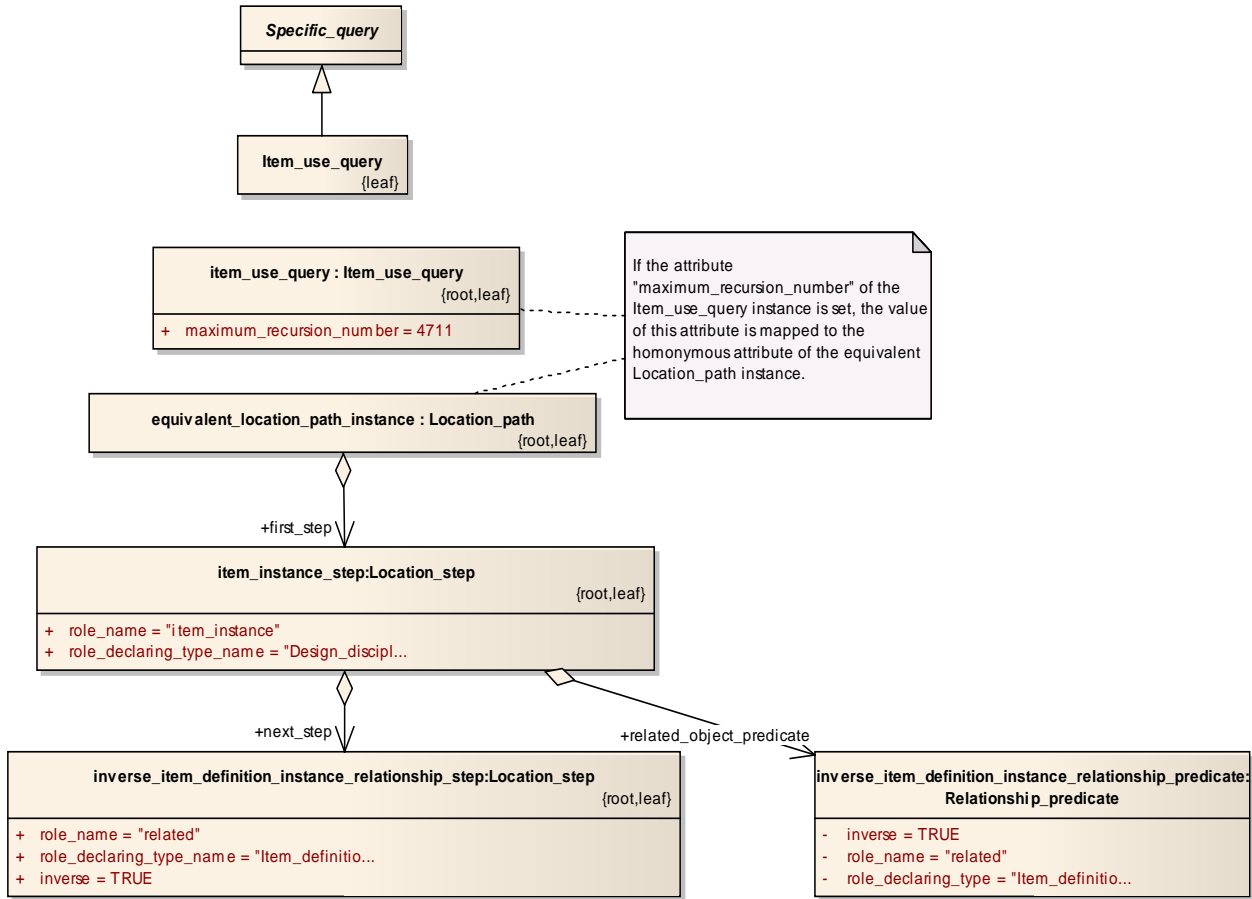


Figure 9.65 - Definition, sample instance and equivalent Location\_path instance of the Item\_use\_query

### 9.7.50 Item\_version\_query

The Item\_version\_query traverses Item\_version objects from Item objects.

#### Parameters

- id : String [0..1]
- id\_scope : String [0..1]
- product\_identification\_uid: UID [0..1]  
 Identifies Product\_identification object that represents a service specific algorithm to filter the Item\_version objects.

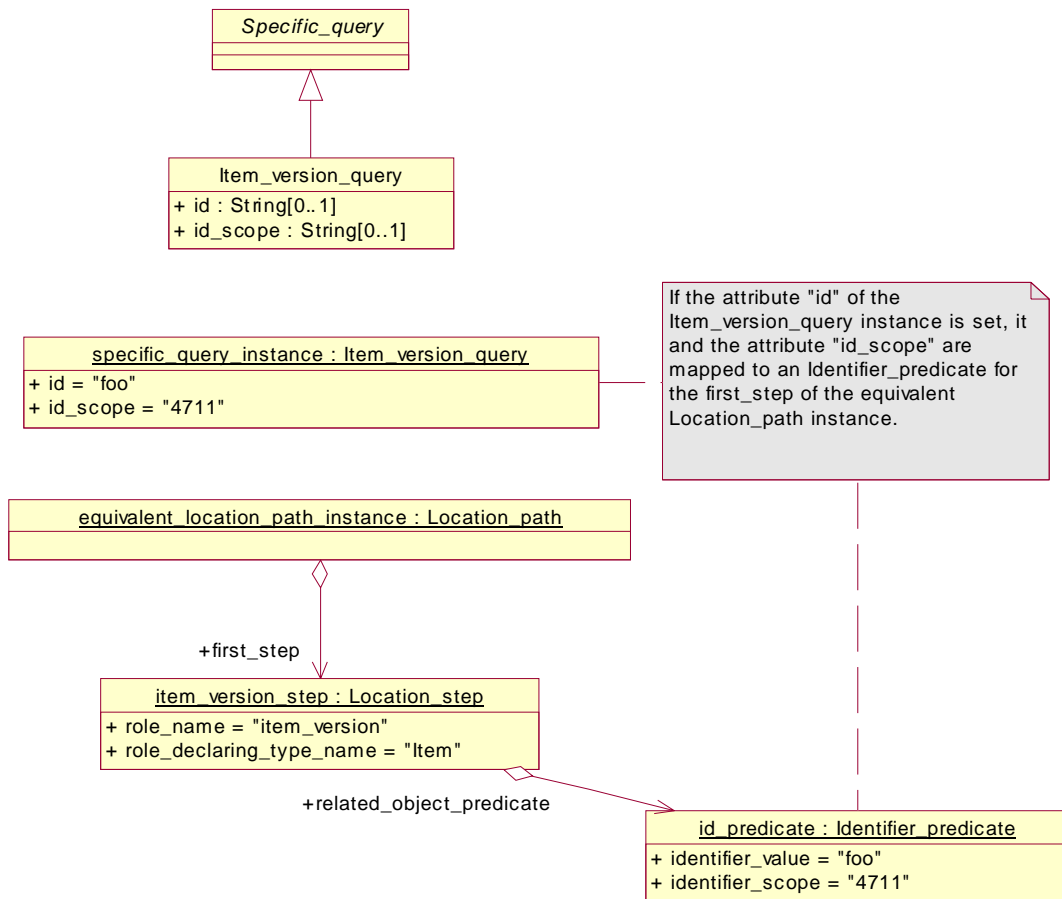


Figure 9.66 - Definition, sample instance and equivalent Location\_path instance of the Item\_version\_query

### 9.7.51 Item\_version\_relationship\_query

The Item\_version\_relationship\_query traverses from Item\_version objects via Item\_version\_relationship objects to Item\_version objects.

#### Parameters

- relation\_type : String [0..1]  
The relation\_type attribute of the queried relationships.
- inverse : Boolean [0..1]

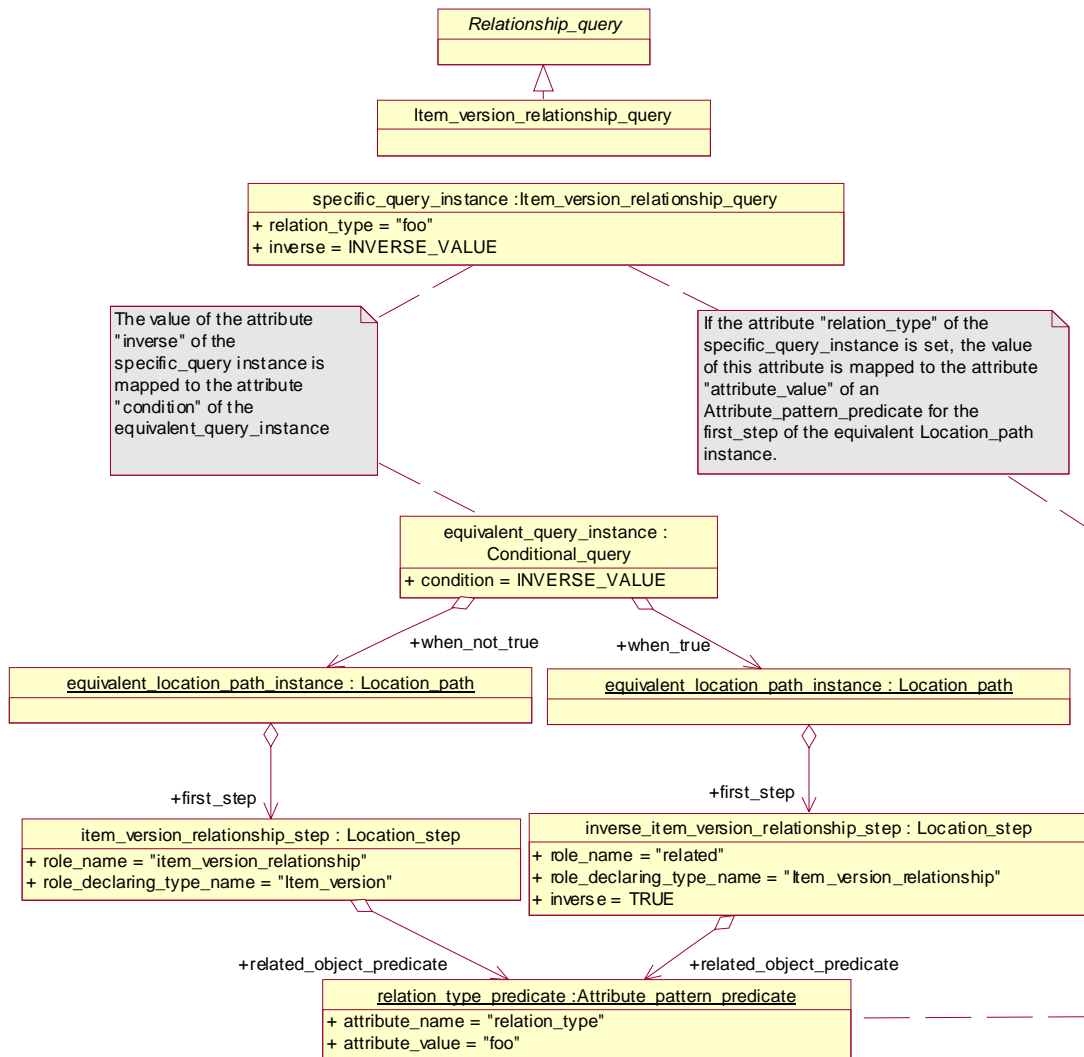


Figure 9.67 - Definition, sample instance and equivalent Location\_path instance of the Item\_version\_relationship\_query

### 9.7.52 Object\_by\_uid\_query

The Object\_by\_uid\_query selects an object by its uid.

#### Parameters

- uid : UID
- opaque\_server\_data : Byte[0..\*]



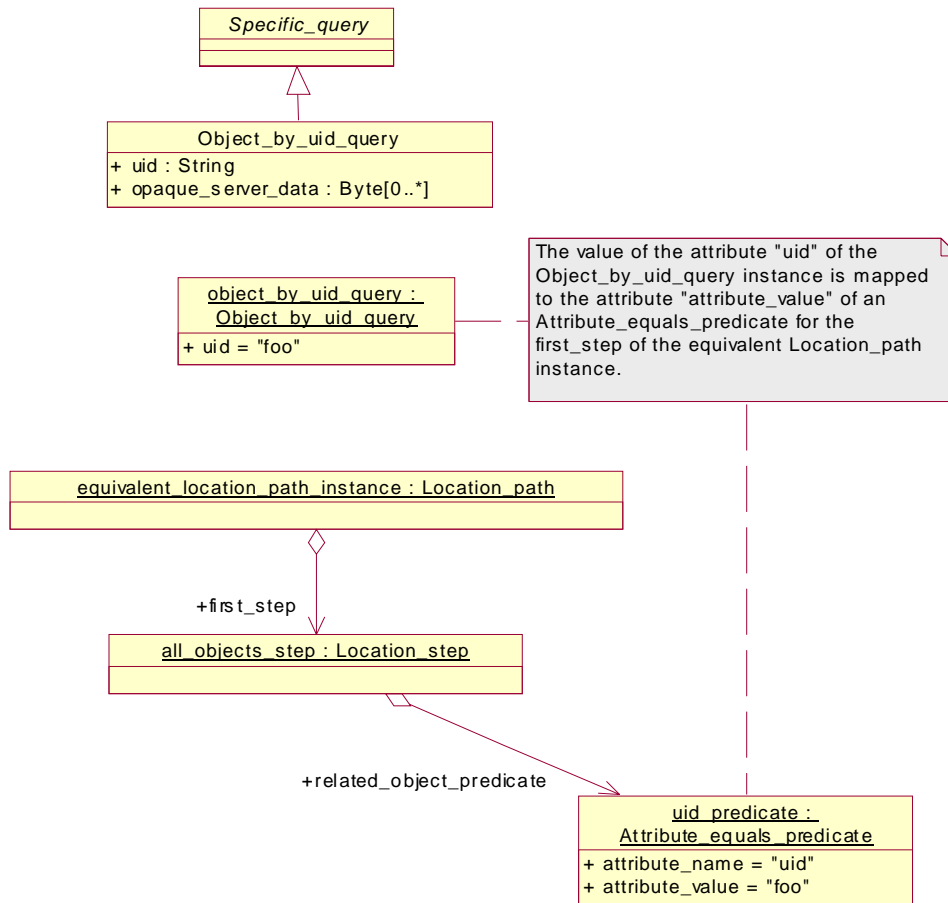


Figure 9.68 - Definition, sample instance and equivalent Location\_path instance of the Object\_by\_uid\_query

### 9.7.53 Objects\_by\_uids\_query

The Objects\_by\_uids\_query selects a set of objects by its uids.

#### Parameters

- uids : UID[1..\*]

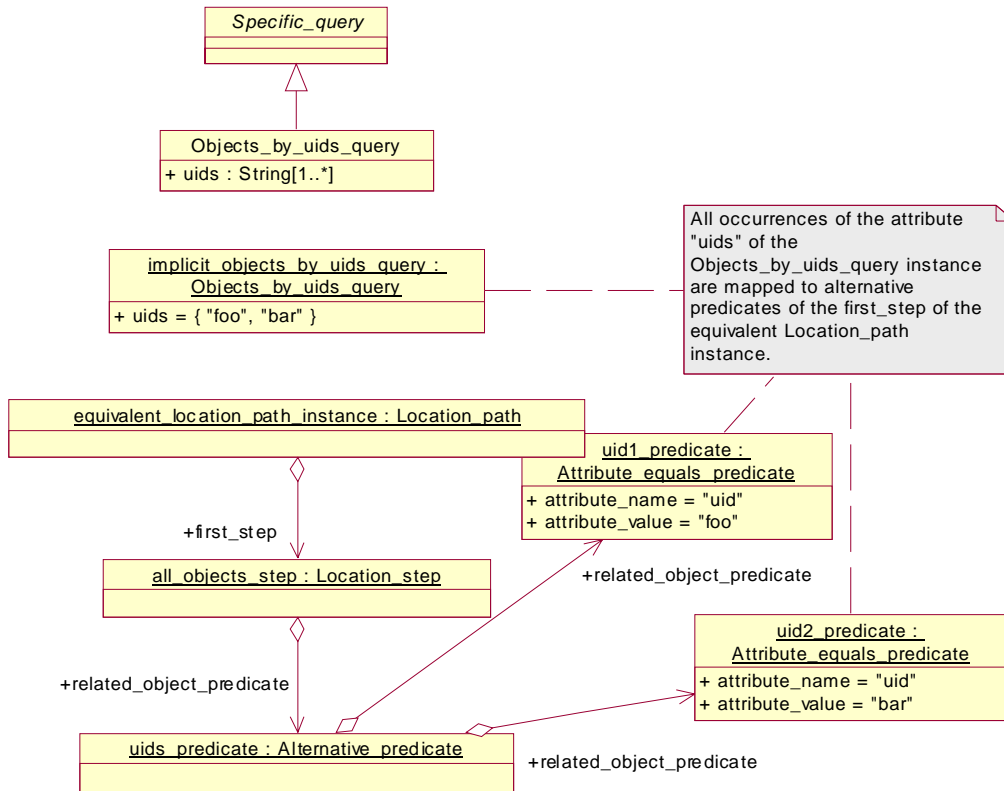


Figure 9.69 - Definition, instance and equivalent explicit Location\_path instance of the Objects\_by\_uids\_query

### 9.7.54 Organization\_query

The Organization\_query selects Organization objects.

#### Parameters

- id : String [0..1]  
The id of the Organization for which the information is queried.
- Id\_scope : String [0..1]
- organization\_name : String [0..1]
- organization\_type : String [0..1]

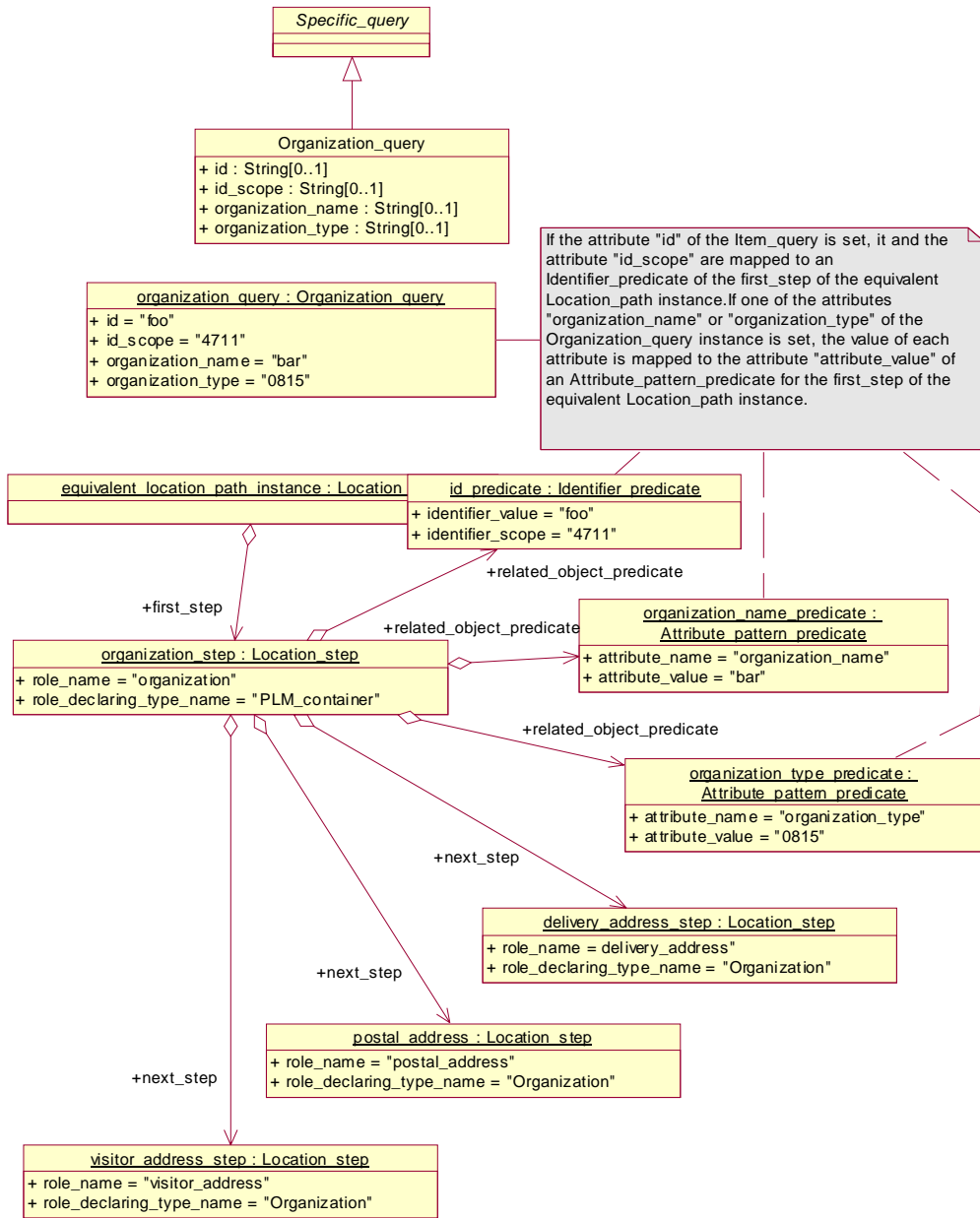


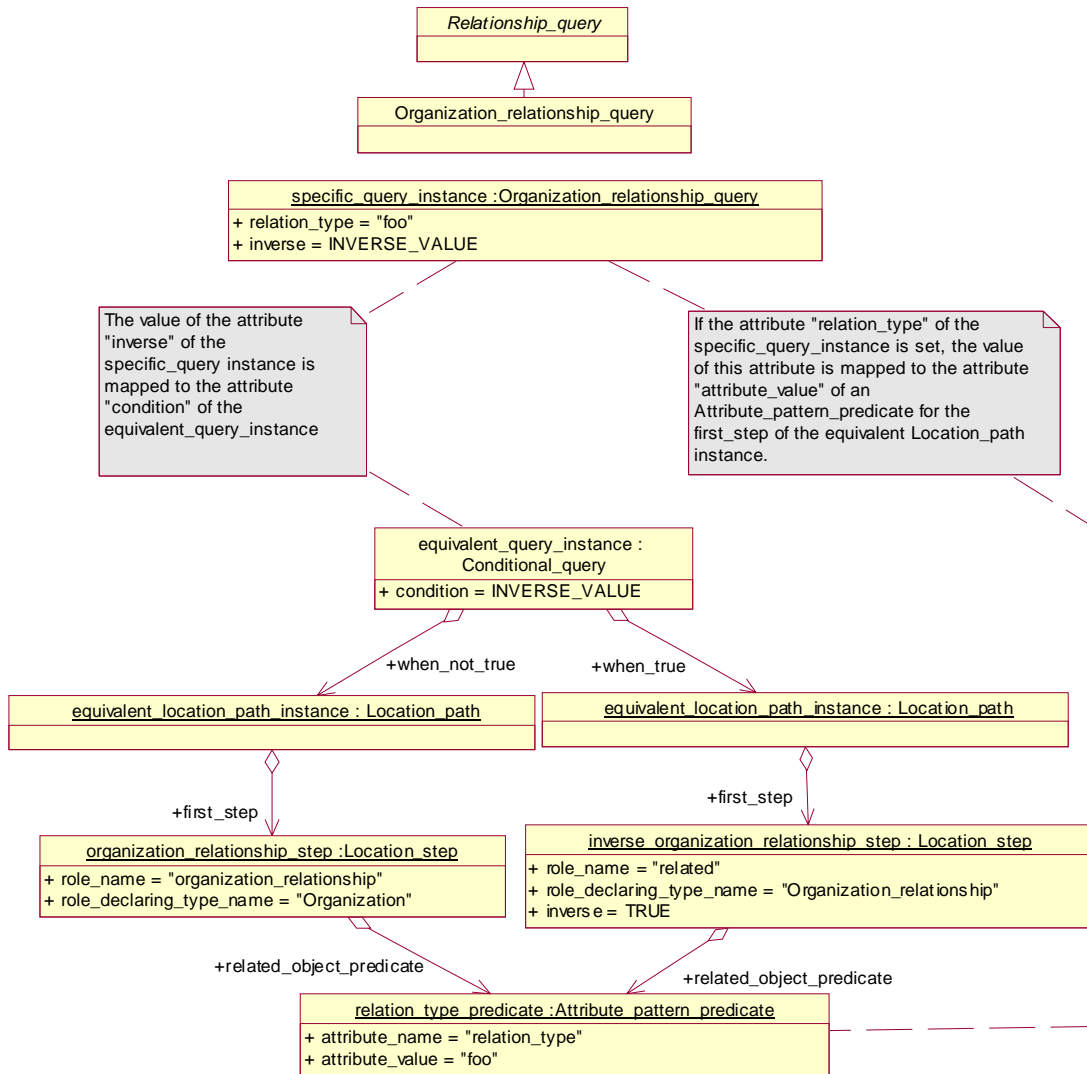
Figure 9.70 - Definition, sample instance and equivalent `Location_path` instance of the `Organization_query`

### 9.7.55 `Organization_relationship_query`

The `Organization_relationship_query` traverses from `Organization` objects via `Organization_relationship` objects to `Organization` objects.

**Parameters**

- relation\_type : String [0..1]
- inverse : Boolean [0..1]



**Figure 9.71 - Definition, sample instance and equivalent Location\_path instance of the Organization\_relationship\_query**

**9.7.56 Person\_in\_organization\_query**

The Person\_in\_organization\_query traverses from Person objects via Person\_in\_organization objects to Organization objects.

## Parameters

- person\_name : String [0..1]
- id: String [0..1]
- role: String [0..1]
- organization\_id : String [0..1]

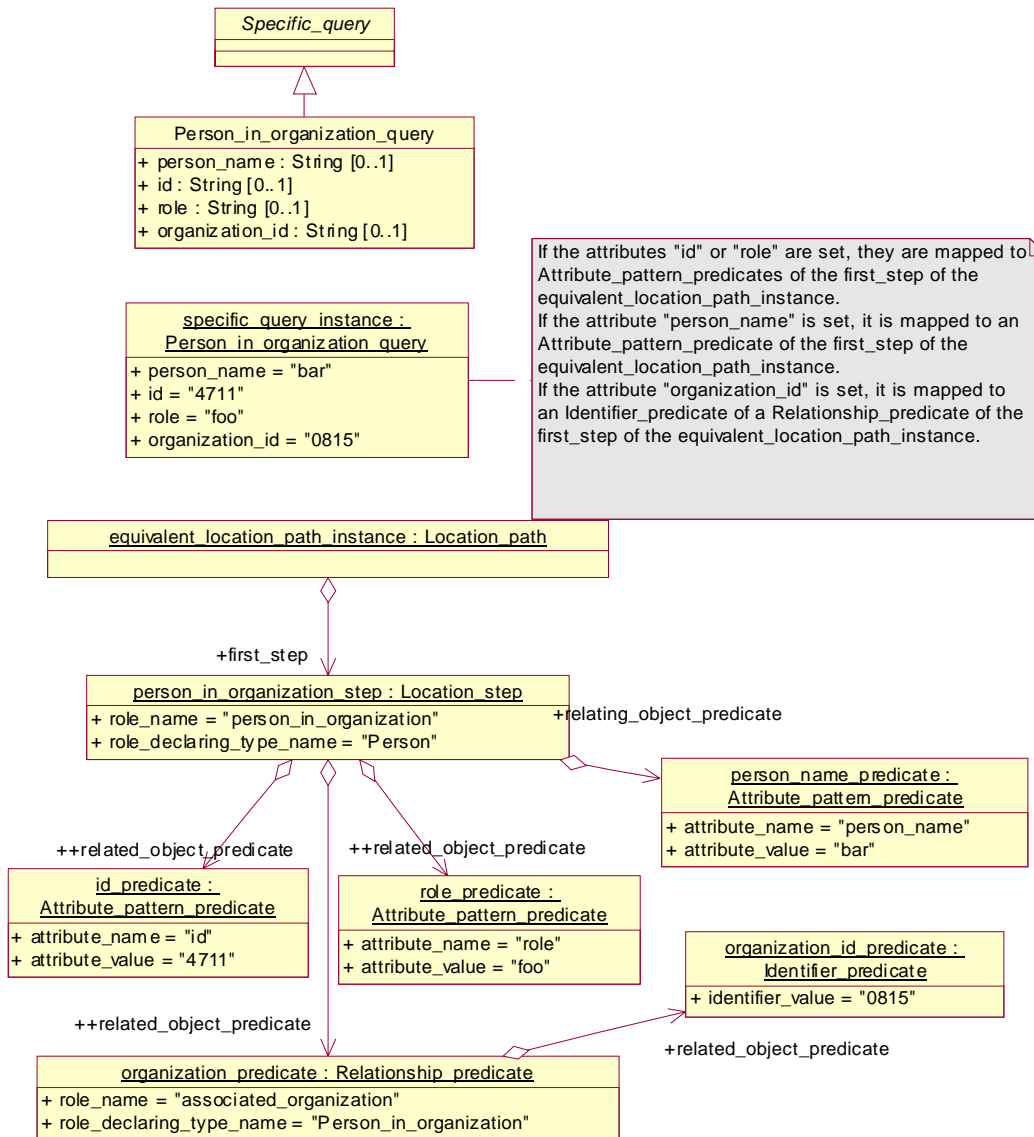


Figure 9.72 - Definition, sample instance and equivalent Location\_path instance of the Person\_in\_organization\_query

### 9.7.57 Person\_in\_organization\_relationship\_query

The Person\_in\_organization\_relationship\_query traverses from Person\_in\_organization objects via Person\_in\_organization\_relationship objects to Person\_in\_organization objects.

#### Parameters

- relation\_type : String [0..1]
- inverse : Boolean [0..1]

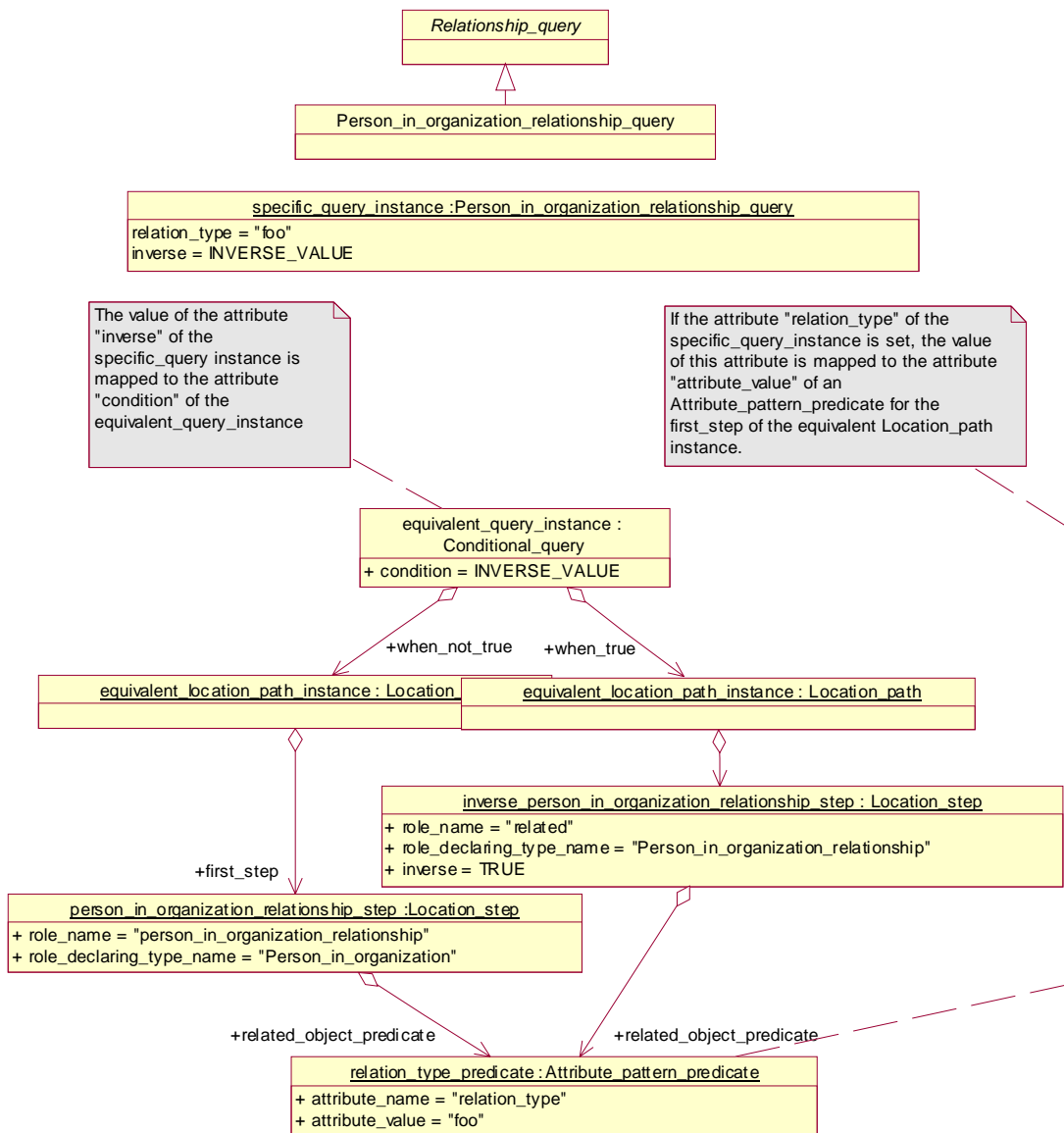


Figure 9.73 - Definition, sample instance and equivalent Location\_path instance of the Person\_in\_organization\_relationship\_query

### 9.7.58 Person\_organization\_assignment\_query

The Person\_organization\_assignment\_query traverses from Person\_organization\_select objects via Person\_organization\_assignment objects to Date\_time\_person\_organization\_element\_select objects.

#### Parameters

- role : String [0..1]
- related\_type\_name : String [0..\*]

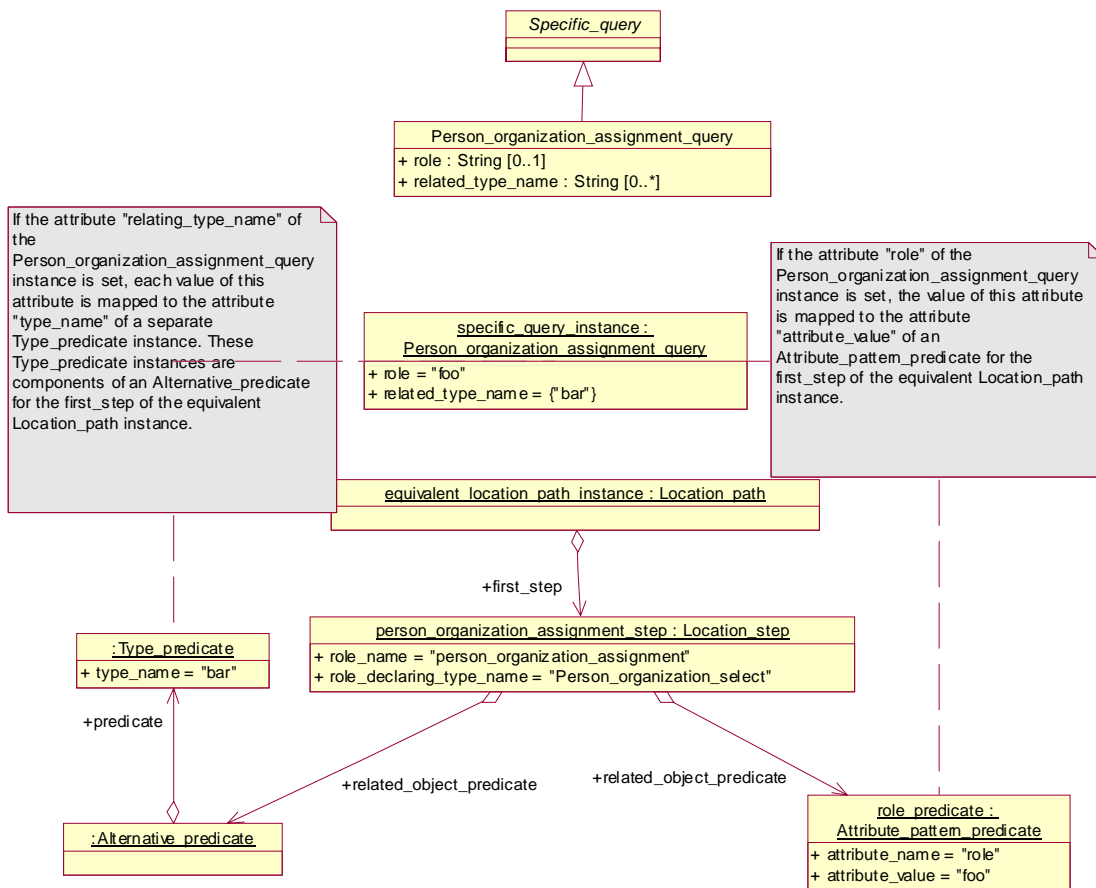


Figure 9.74 - Definition, sample instance and equivalent Location\_path instance of the Person\_organization\_assignment\_query

### 9.7.59 Person\_query

The Person\_query selects Person objects.

#### Parameters

- person\_name : String [0..1]

- id: String [0..1]
- role: String [0..1]
- organization\_id : String [0..1]

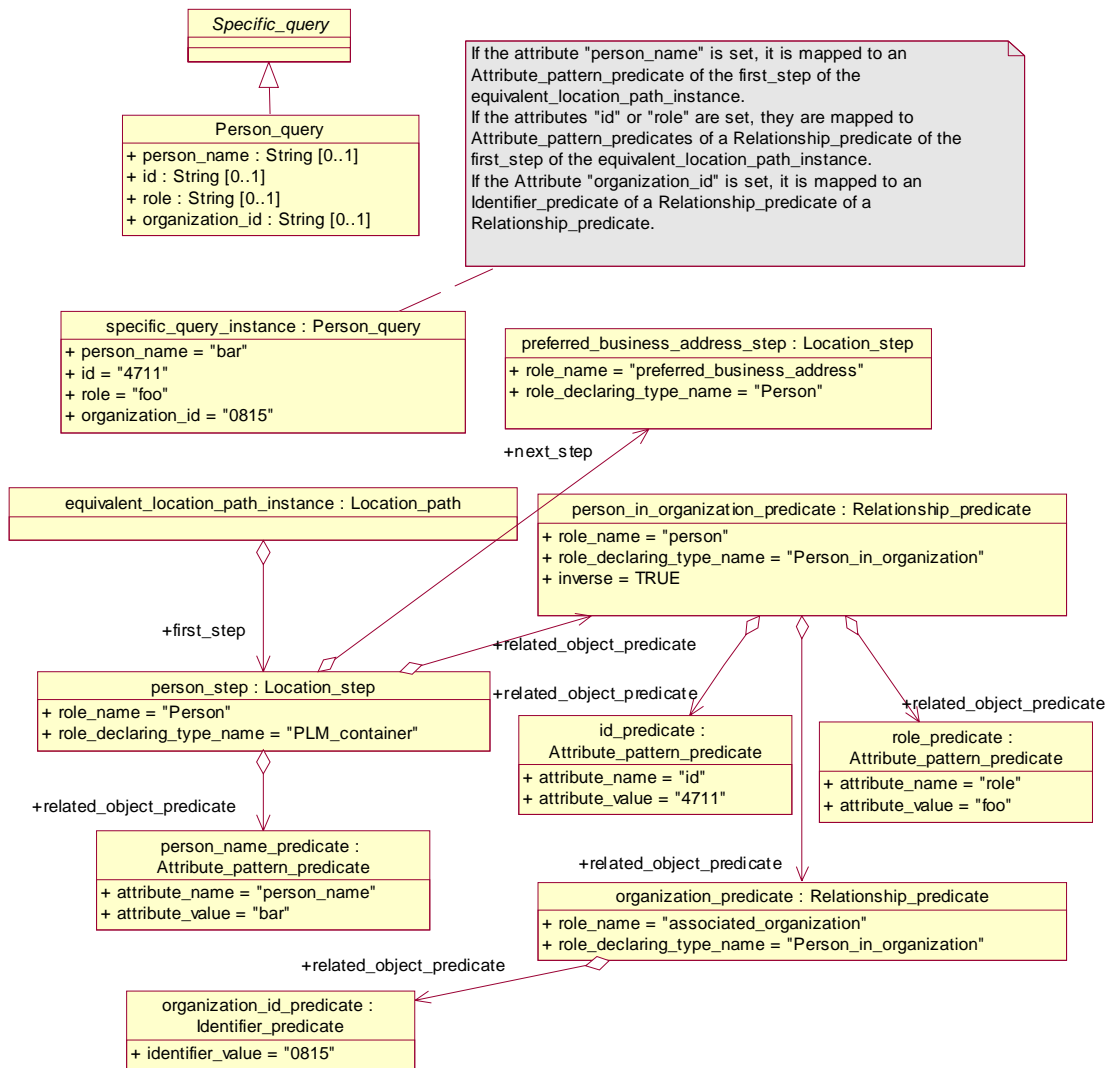


Figure 9.75 - Definition, sample instance and equivalent Location\_path instance of the Person\_query

### 9.7.60 Product\_class\_query

The `Product_class_query` selects `Product_class` objects.

#### Parameters

- id : String



- id\_scope : String [0..1]
- name : String
- name\_language : Language

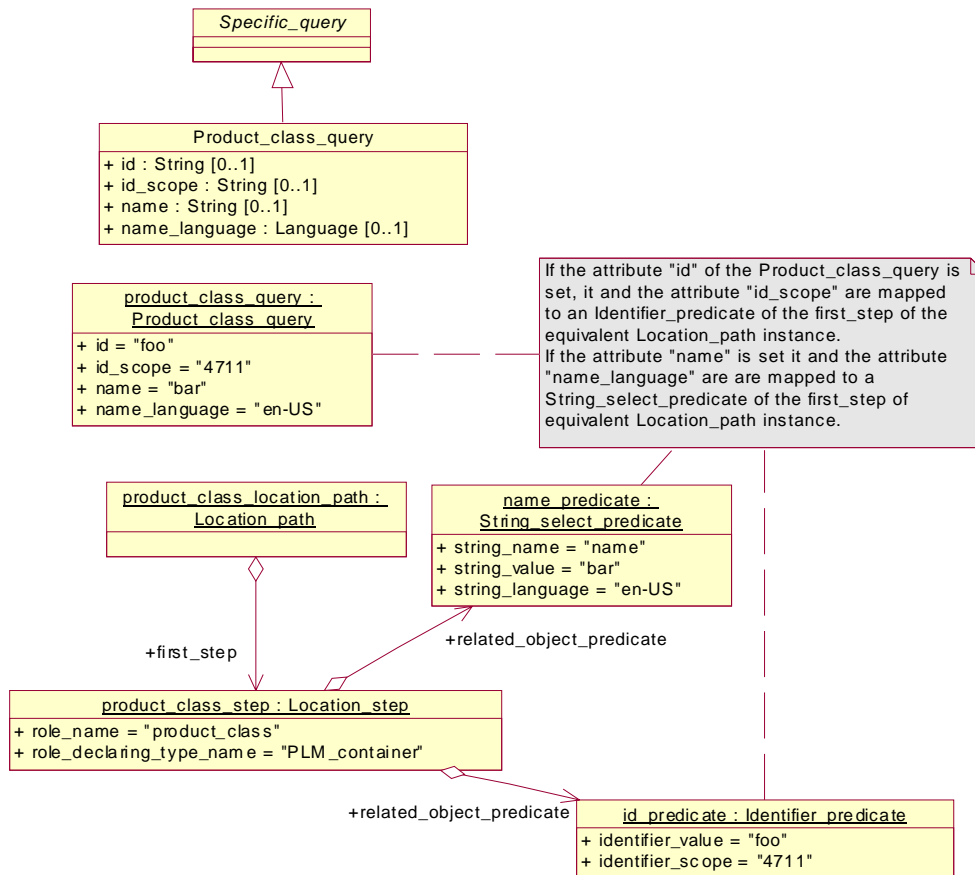


Figure 9.76 - Definition, sample instance and equivalent Location\_path instance of the Product\_class\_query

### 9.7.61 Product\_identification\_query

The Product\_identification\_query selects Product\_identification objects.

#### Base Class

- Specific\_query

#### Parameters

- id: string [0..1]
- id\_scope: string [0..1]
- name: string [0..1]

- name\_language: language [0..1]
- version\_id: string [0..1]

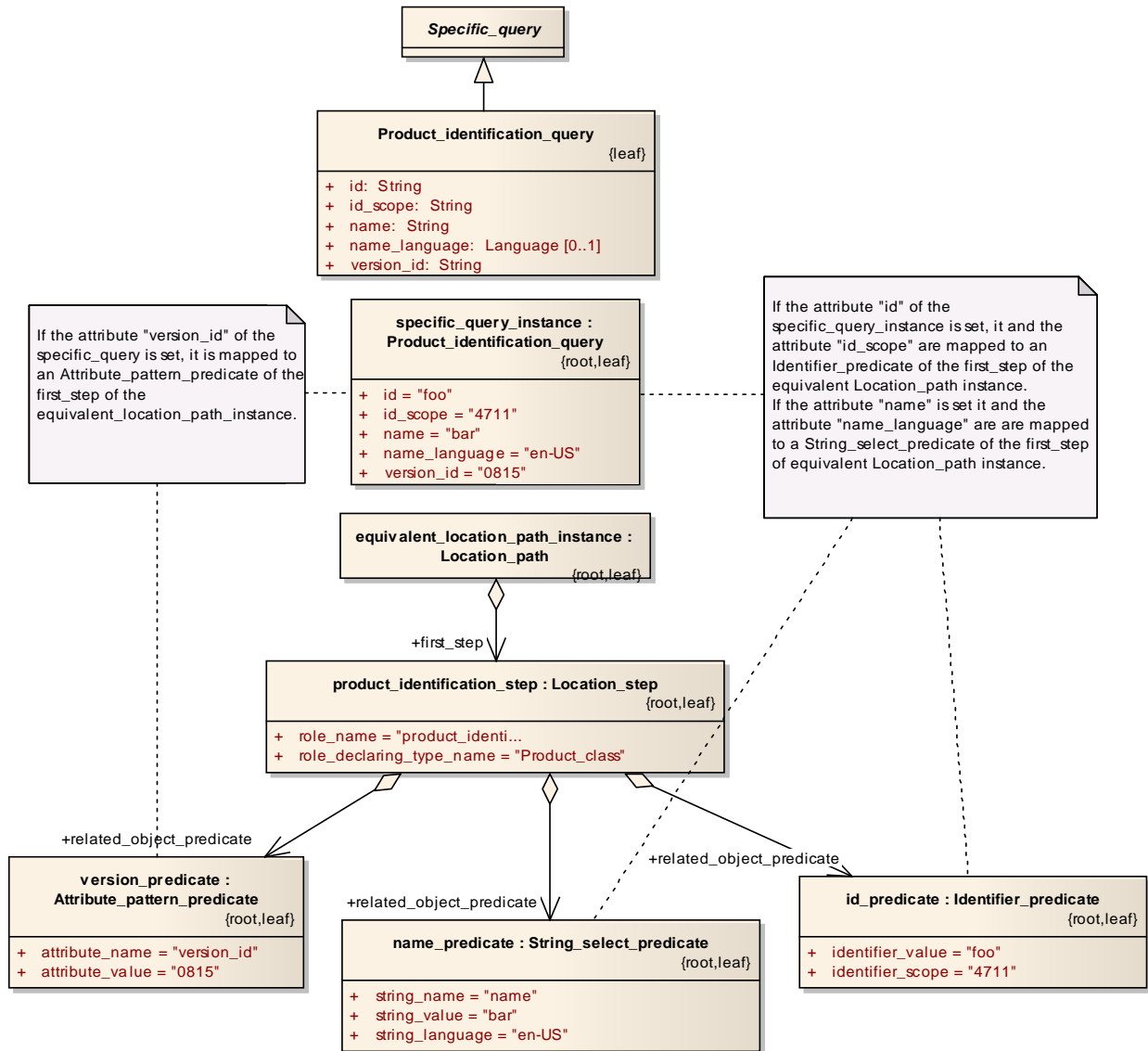


Figure 9.77 - Definition, sample instance and equivalent Location\_path instance of the Product\_identification\_query

### 9.7.62 Product\_structure\_query

The `Product_structure_query` traverses `Product_structure_relationship` objects from `Complex_product` objects.

#### Parameters

- relation\_type : String[0..1]

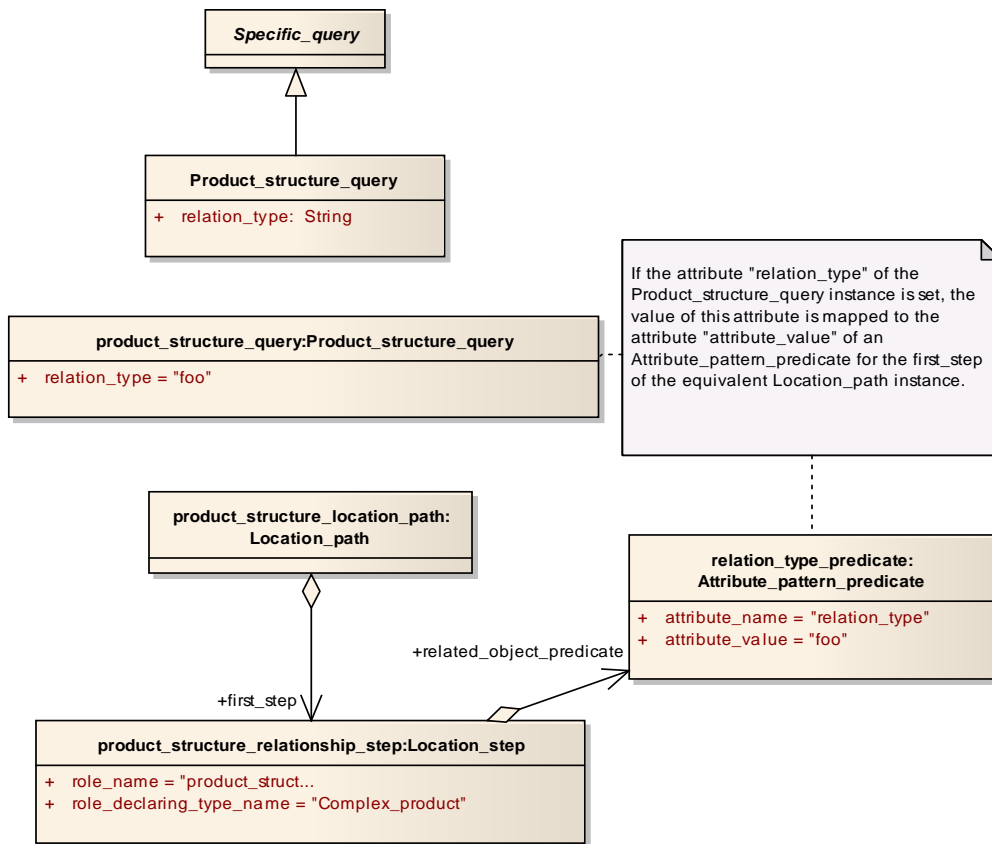


Figure 9.78 - Definition, sample instance and equivalent Location\_path instance of the Product\_structure\_query

### 9.7.63 Project\_assignment\_query

The Project\_assignment\_query traverses from Project objects via Project\_assignment objects to Project\_information\_select objects.

#### Parameters

- role : String [0..1]
- related\_type\_name : String [0..\*]

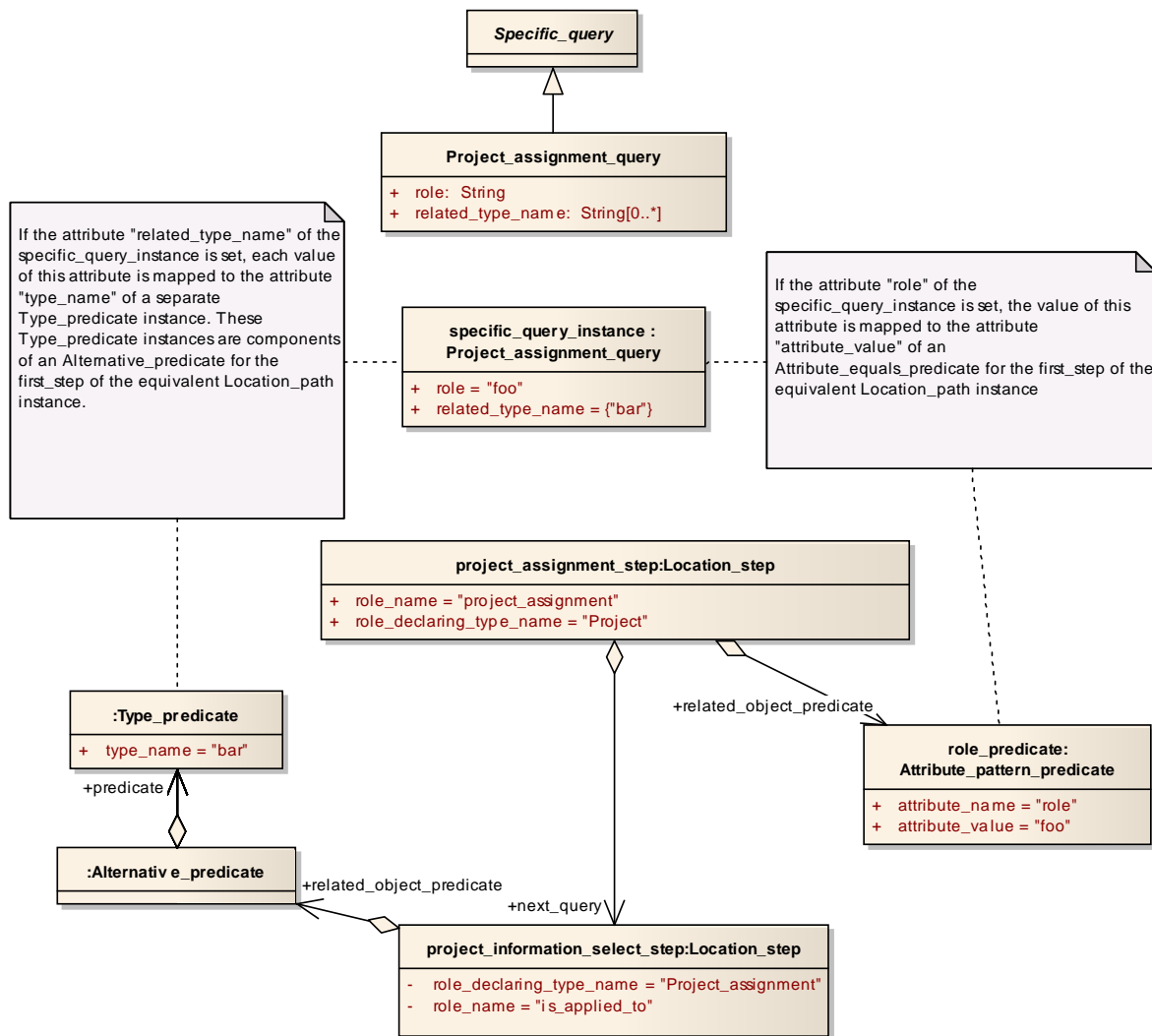


Figure 9.79 - Definition, sample instance and equivalent Location\_path instance of the Project\_assignment\_query

### 9.7.64 Project\_query

The Project\_query selects Project objects.

#### Base Class

- Specific\_query

#### Parameters

- id: string [0..1]
- id\_scope: string [0..1]
- name: string [0..1]
- name\_language: language [0..1]

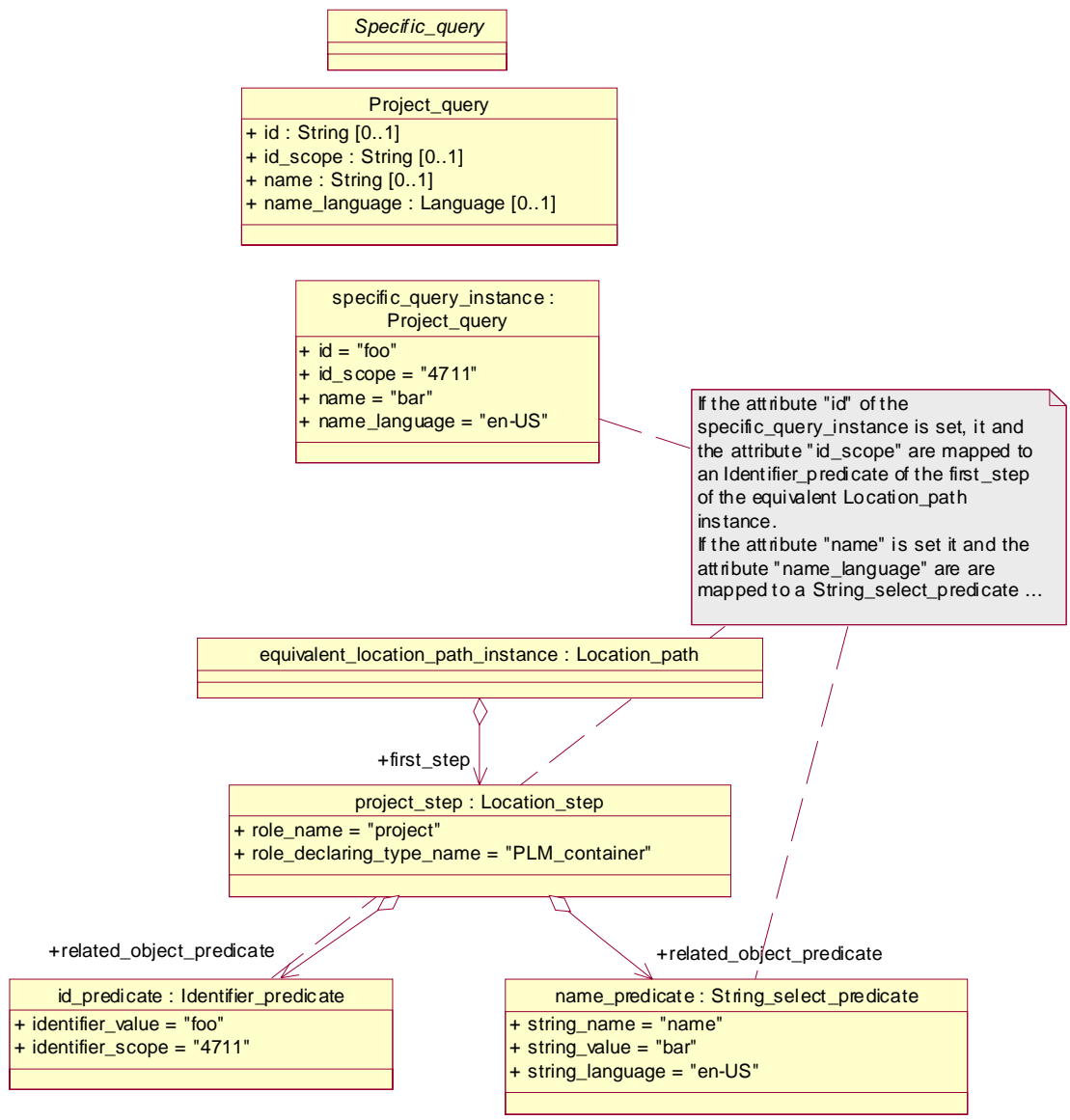


Figure 9.80 - Definition, sample instance and equivalent Location\_path instance of the Project\_query

### 9.7.65 Simple\_property\_value\_query

The Simple\_property\_query traverses from Simple\_property\_select objects via Simple\_property\_association objects to Simple\_property\_value objects.

#### Parameters

- relating\_type\_name : String [0..\*]
- value\_name : String [0..1]

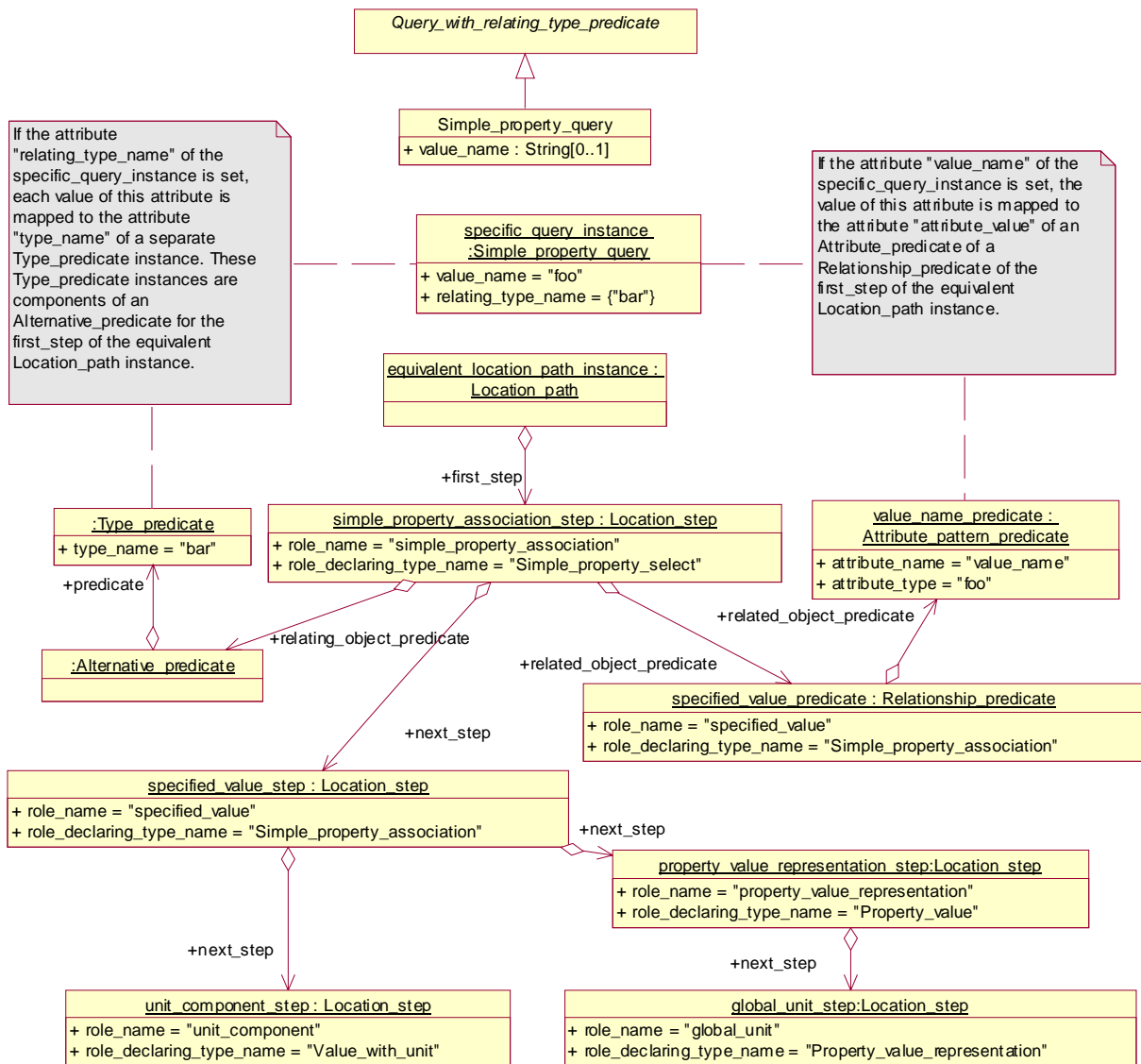


Figure 9.81 - Definition, sample instance and equivalent Location\_path instance of the Simple\_property\_value\_query

### 9.7.66 Work\_order\_query

The Work\_order\_query selects Work\_order objects.

#### Base Class

- Specific\_query

**Parameters**

- id: string [0..1]
- id\_scope: string [0..1]
- work\_order\_type: string [0..1]
- version\_id: string [0..1]
- classification\_role: string [0..1]
- classification\_id: string [0..1]
- description: string [0..1]
- description\_language: string [0..1]

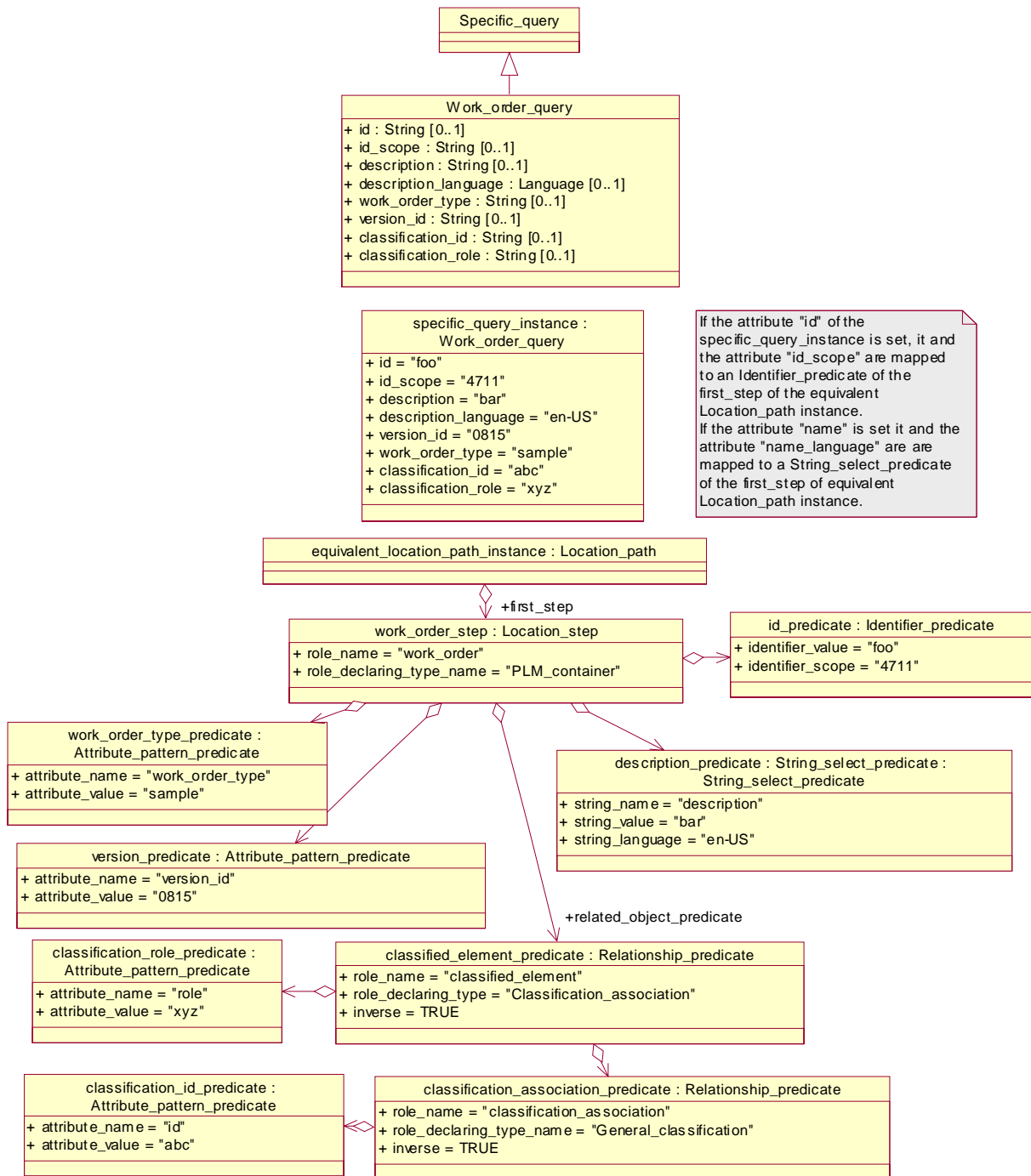


Figure 9.82 - Definition, sample instance and equivalent Location\_path instance of the Work\_order\_query



### 9.7.67 Work\_order\_is\_controlling\_query

The Work\_order\_is\_controlling\_query traverses from Work\_order objects to the Activity objects that are referenced by the is\_controlling reference.

#### Base Class

- Specific\_query

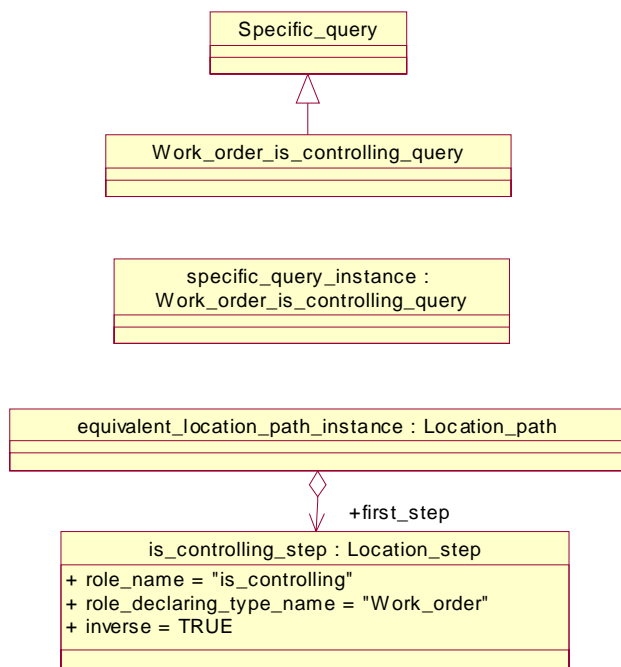


Figure 9.83 - Definition, sample instance and equivalent Location\_path instance of the Work\_order\_is\_controlling\_query

### 9.7.68 Work\_request\_activity\_query

The Work\_request\_activity\_query traverses from Work\_request objects to Activity objects.

#### Parameters

- none

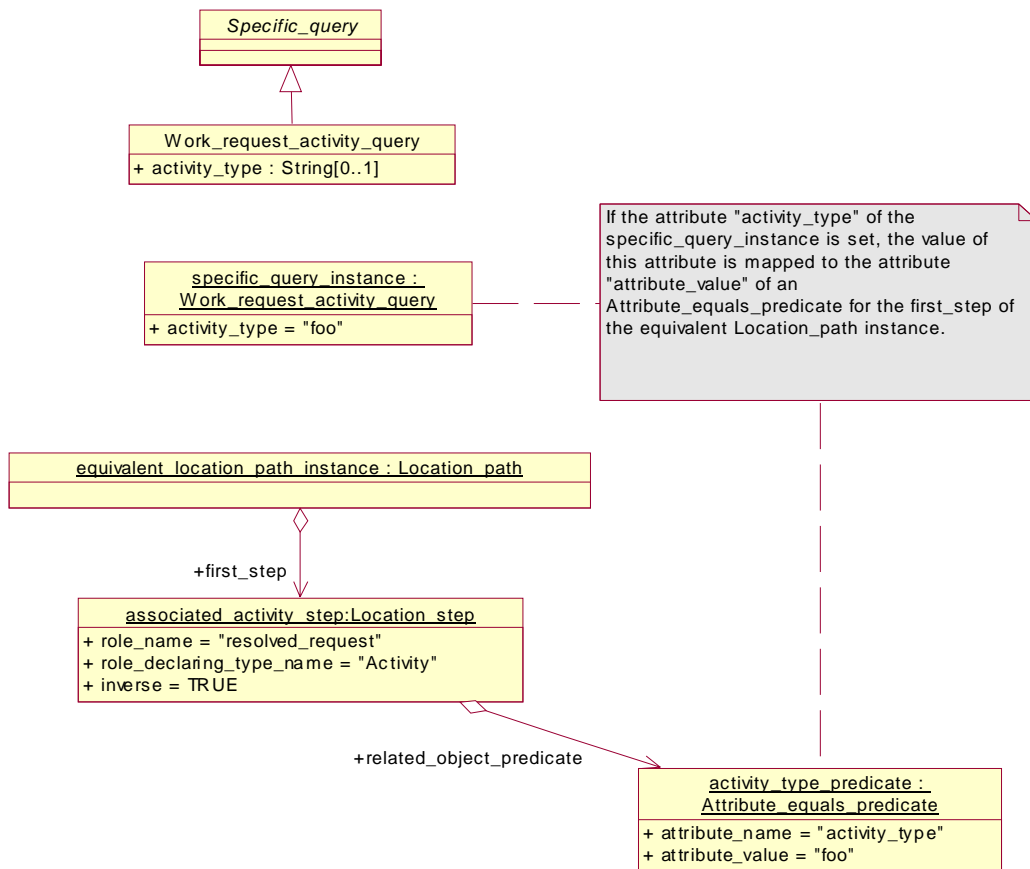


Figure 9.84 - Definition, sample instance and equivalent Location\_path instance of the Work\_request\_activity\_query

### 9.7.69 Work\_request\_query

The Work\_request\_query selects Work\_request objects.

#### Parameters

- id : String [0..1]
- request\_type : String [0..1]
- status : String [0..1]
- version\_id : String [0..1]
- classification\_role : String [0..1]
- classification\_id : String [0..1]

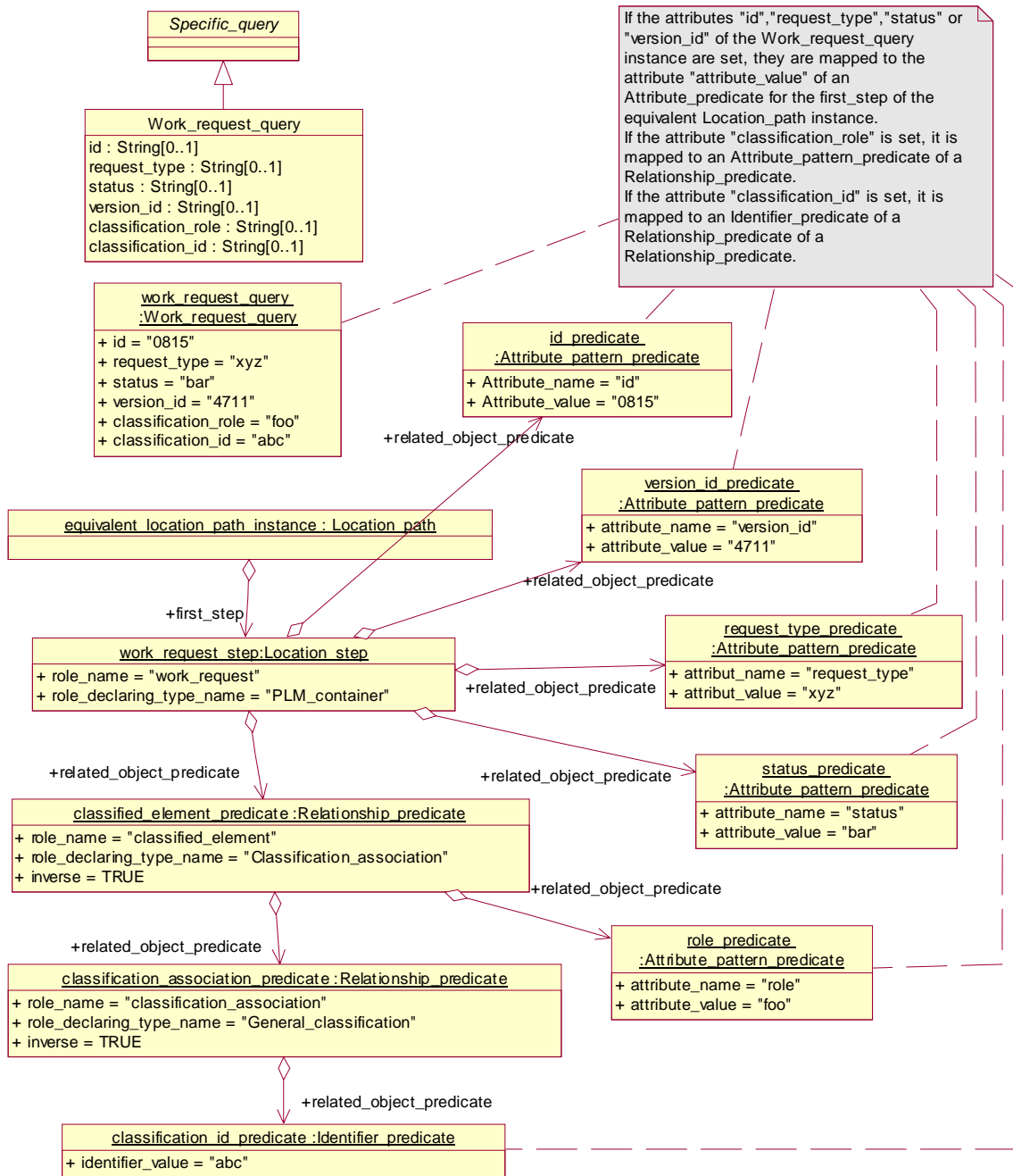


Figure 9.85 - Definition, sample instance and equivalent Location\_path instance of the Work\_request\_query

### 9.7.70 Work\_request\_relationship\_query

The Work\_request\_relationship\_query traverses from Work\_request objects via Work\_request\_relationship objects to Work\_request objects.

#### Parameters

- relation\_type : String [0..1]
- inverse : Boolean [0..1]

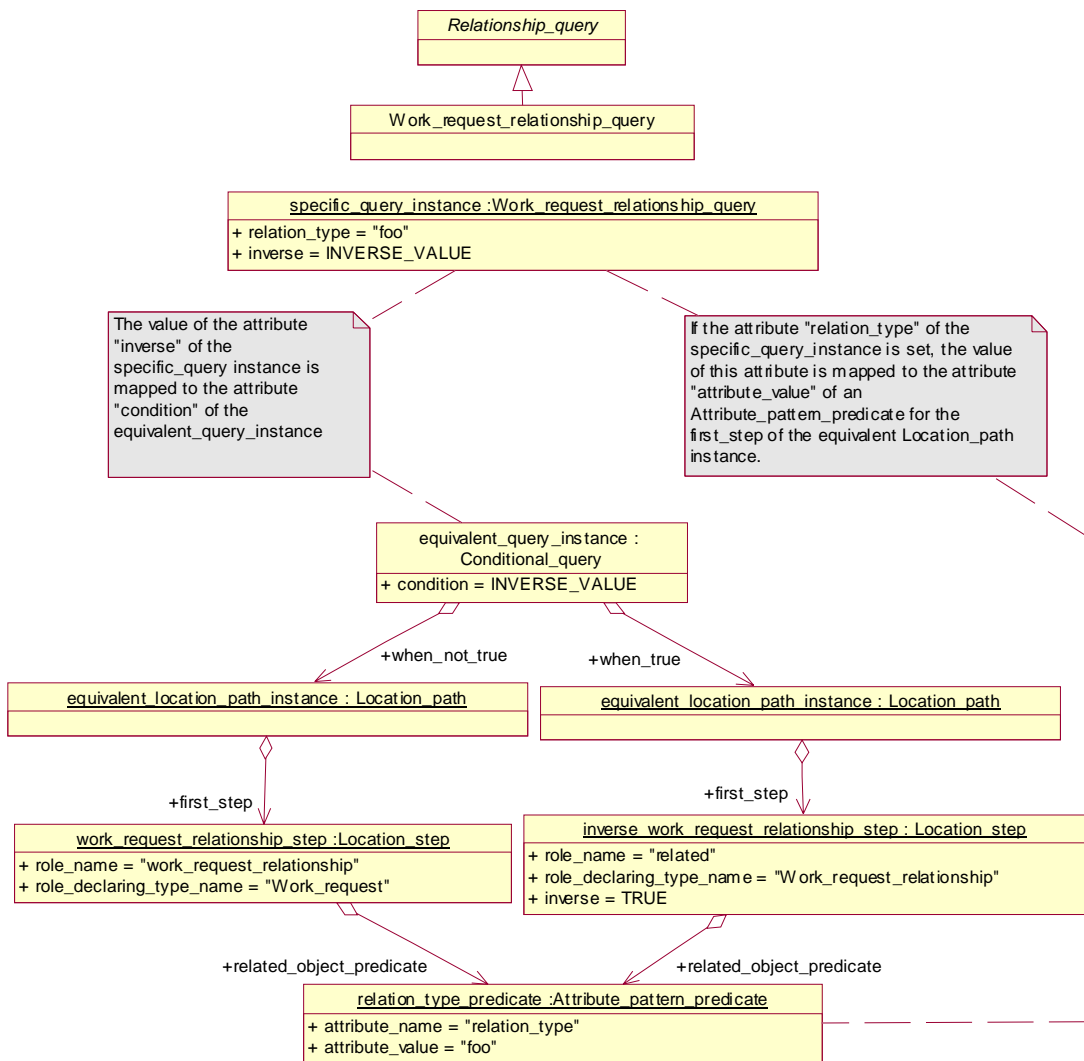


Figure 9.86 - Definition, sample instance and equivalent Location\_path instance of the Work\_request\_relationship\_query

### 9.7.71 Work\_request\_scope\_query

The Work\_request\_scope\_query traverses from Work\_request objects to the Activity\_element\_select objects that are the scope of the Work\_request objects.

#### Parameters

- none

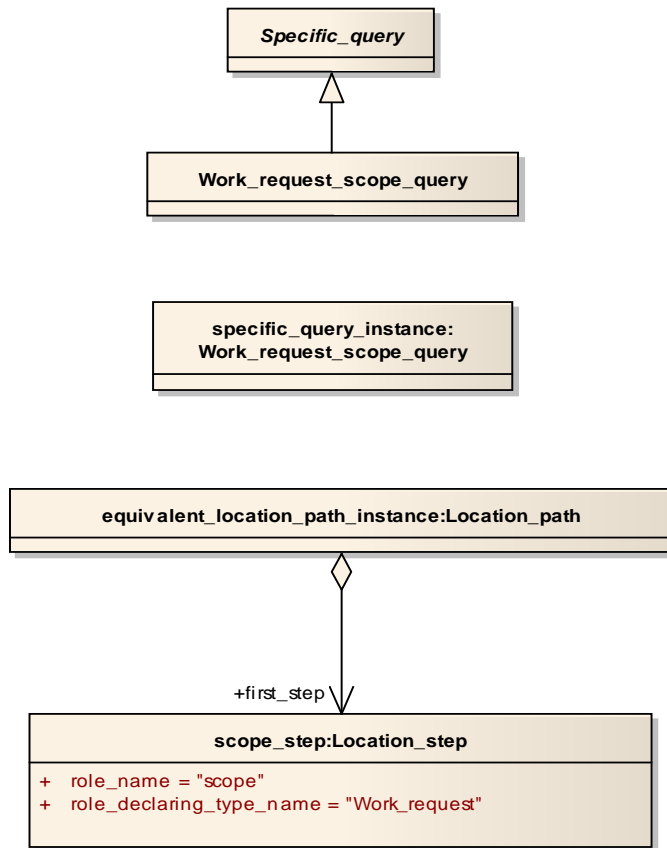
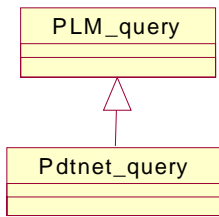


Figure 9.87 - Definition, sample instance and equivalent Location\_path instance of the Work\_request\_scope\_query

## 9.8 PDTnet Queries Conformance Point

The PDTnet conformance point defines a set of specialized queries that fulfill the requirements of the use cases described in Section 7.2. The semantic of each specialized query of this conformance point is defined by an equivalent Location\_path instance. The semantic of Location\_path is defined in the Generic Queries Conformance Point, Section 9.3.

The semantics of each specialized query of this conformance point is defined by an equivalent Query instance from the Specific Queries Conformance Point. All queries of the PDTnet Queries Conformance Point extends directly or indirectly the abstract base class Pdtnet\_query.



**Figure 9.88 - Class diagram of the Pdtnet\_query**

### 9.8.1 General\_detail\_query

The General\_detail\_query returns general detail information from objects selected by a uid.

#### **Parameters**

- uids : UID [1..\*]
- relating\_type\_name : String [0..\*]
- add\_aliases : Boolean [0..1]
- lias\_id : String [0..1]
- add\_authorizations : Boolean [0..1]
- authorization\_role : String [0..1]
- add\_dates : Boolean [0..1]
- date\_role : String [0..1]
- add\_properties : Boolean [0..1]
- property\_name : String [0..1]
- add\_classifications : Boolean [0..1]
- classification\_role : String [0..1]
- add\_approvals : Boolean [0..1]
- approval\_level : String [0..1]
- add\_activities : Boolean [0..1]
- activity\_role : String [0..1]
- add\_effectivities : Boolean [0..1]
- effectivity\_role : String [0..1]

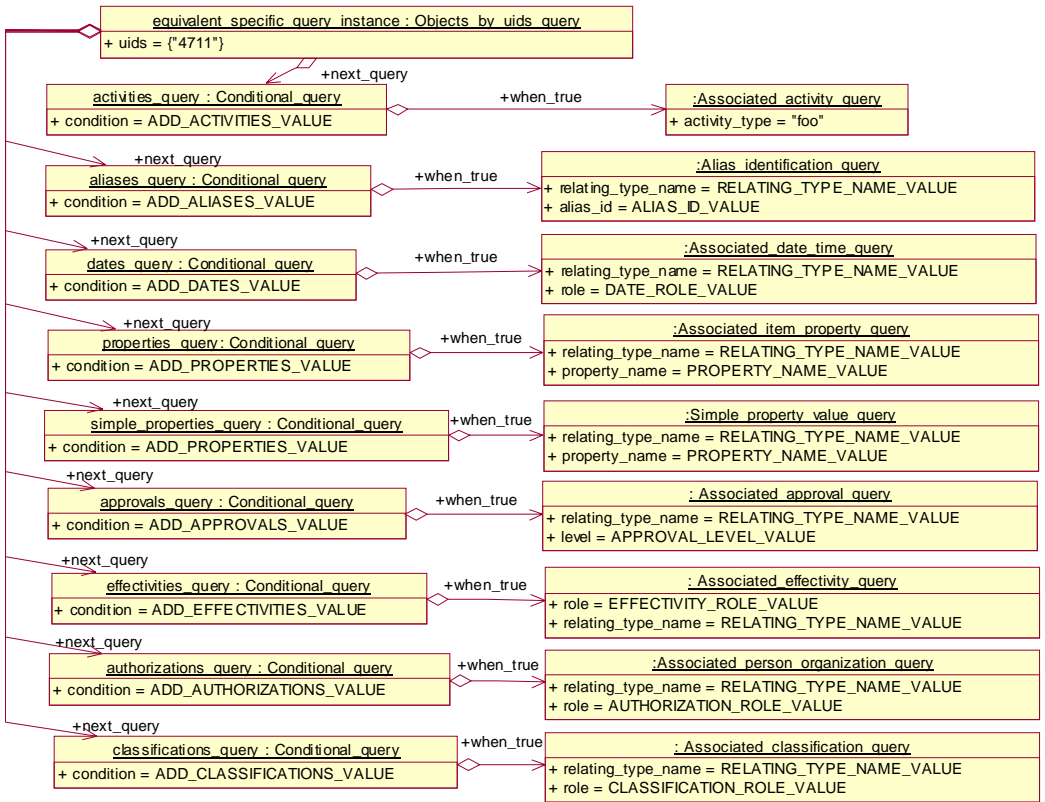
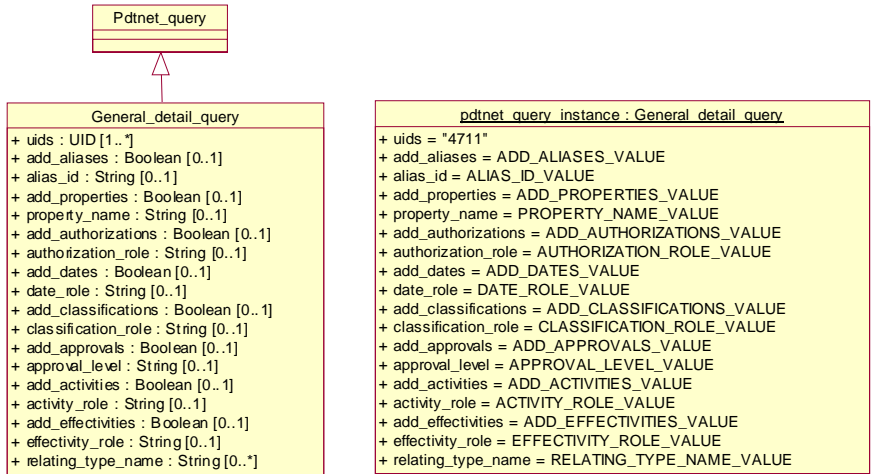


Figure 9.89 - Definition, sample instance and equivalent specific query instance of the General\_detail\_query

## 9.8.2 Document\_detail\_query

The Document\_detail\_query returns detail information of a Document, Document\_version, or Document\_representation object selected by a uid.

### Parameters (without inherited)

- classification\_name : String [0..1]
- add\_versions : Boolean [0..1]
- version\_id : String [0..1]
- add\_representations : Boolean [0..1]

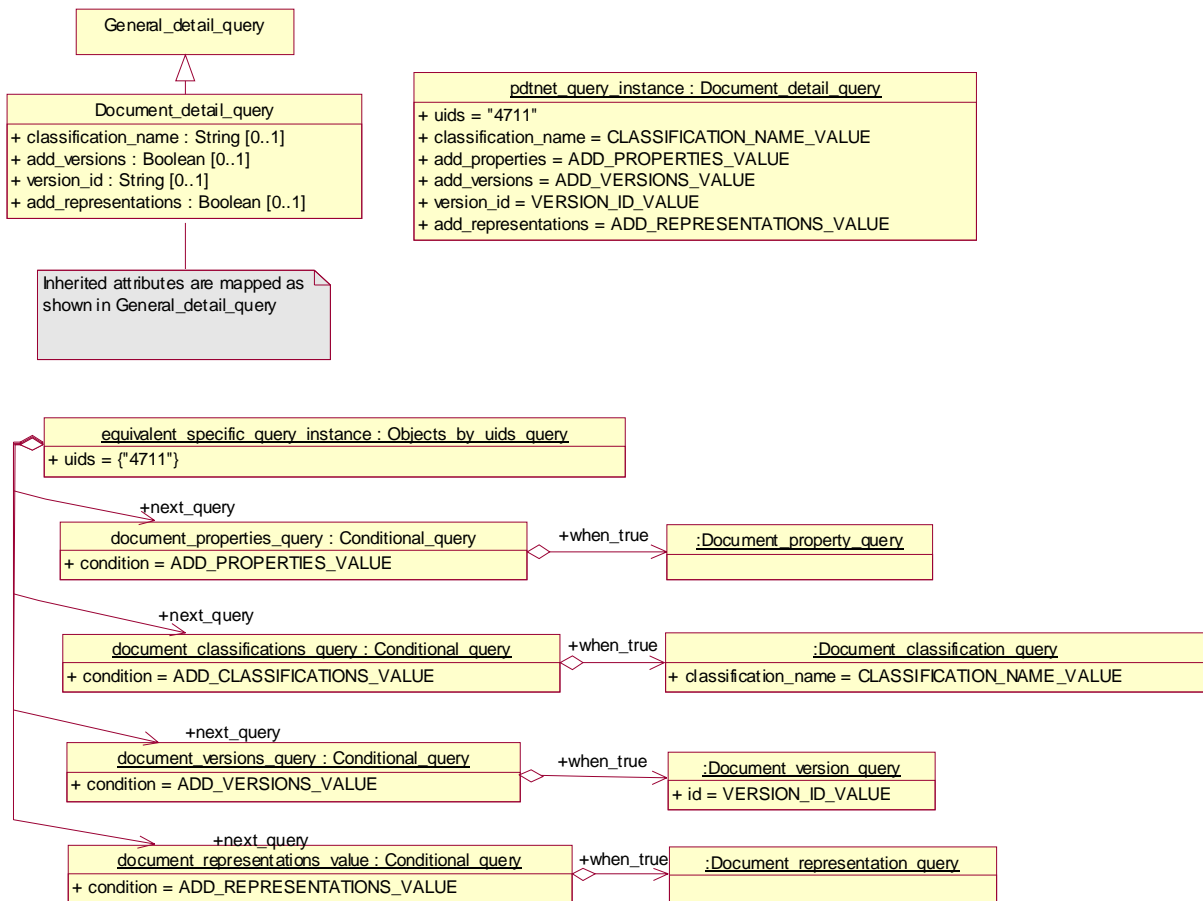


Figure 9.90 - Definition, sample instance and equivalent specific query instance of the Document\_detail\_query

## 9.8.3 Document\_selection\_query

The Document\_selection\_query selects objects of class Document and includes related Document\_version and Document\_representation objects.



**Parameters**

- name : String [0..1]
- id : String [0..1]
- version\_id : String [0..1]
- classification\_name : String [0..1]

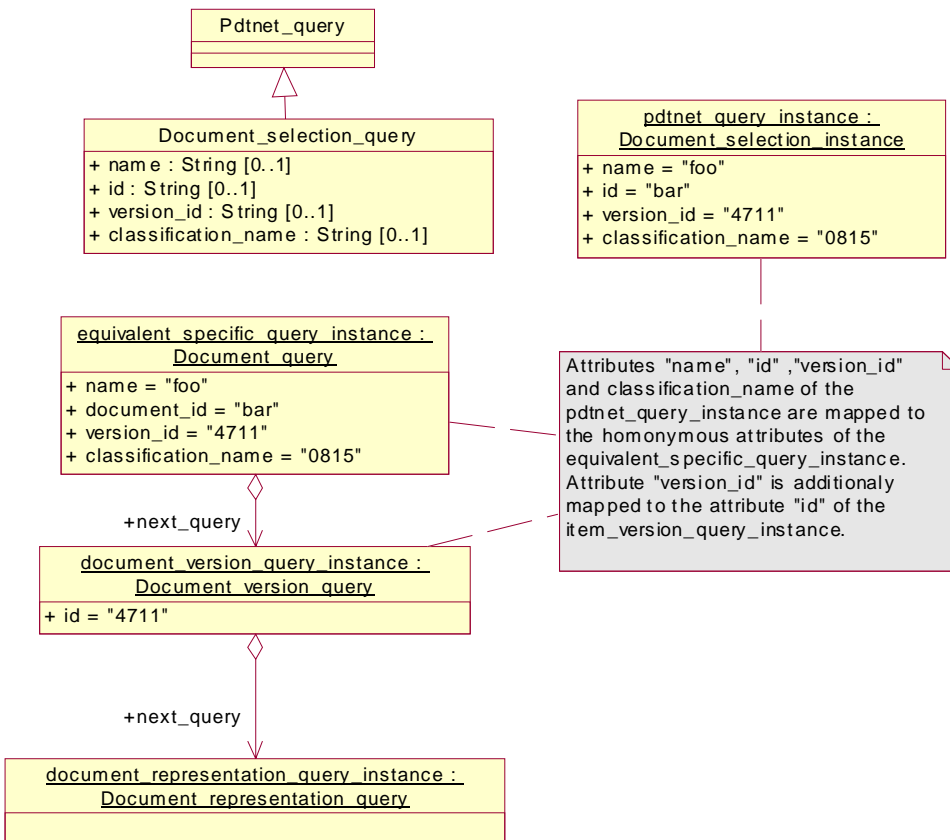


Figure 9.91 - Definition, sample instance and equivalent specific query instance of the Document\_selection\_query

**9.8.4 Document\_traversal\_query**

The Document\_traversal\_query traverses from a Document\_representation object selected by its uid via Document\_structure objects to related Document\_representation objects in a document structure.

**Parameters**

- uid : UID
- relation\_type : String [0..1]

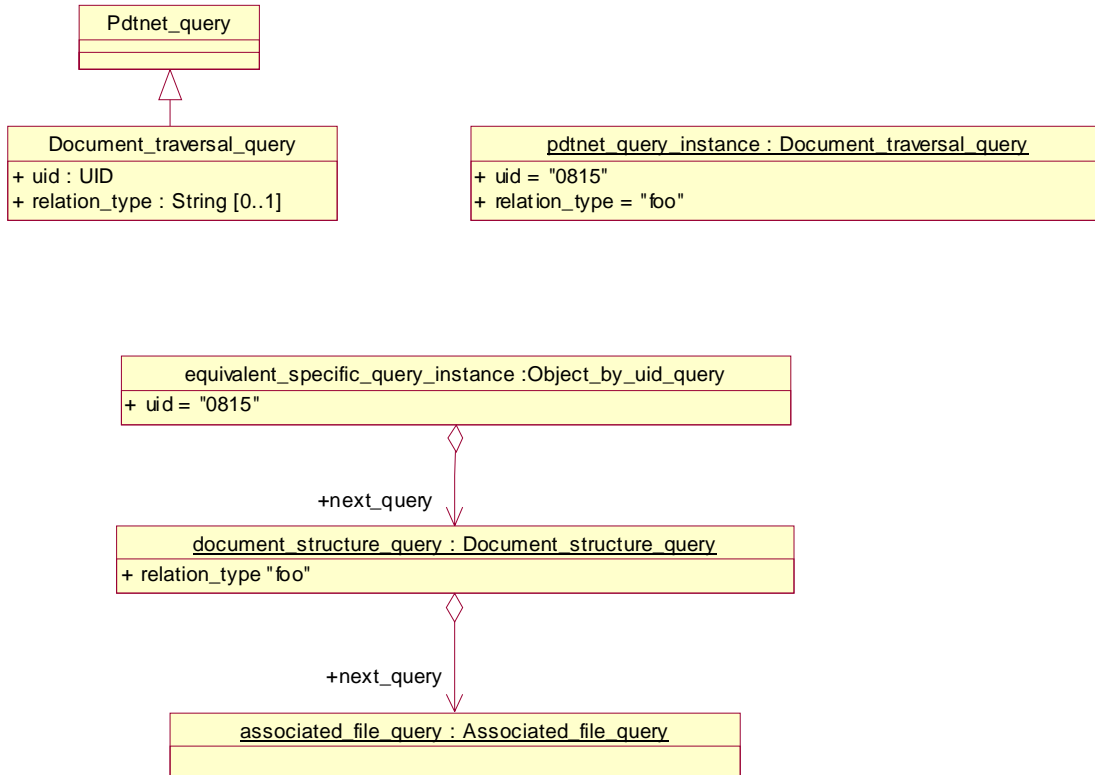


Figure 9.92 - Definition, sample instance and equivalent specific query instance of the Document\_traversal\_query

### 9.8.5 Item\_detail\_query

The Item\_detail\_query returns detail information of an Item, Item\_version, Design\_discipline\_item\_definition, Item\_instance or Assembly\_component\_relationship object selected by a uid.

#### Parameters (without inherited)

- add\_version\_relationships : Boolean [0..1]
- version\_relationship\_type : String [0..1]
- add\_documents : Boolean [0..1]
- document\_role : String [0..1]
- classification\_name : String [0..1]
- add\_placement : Boolean [0..1]

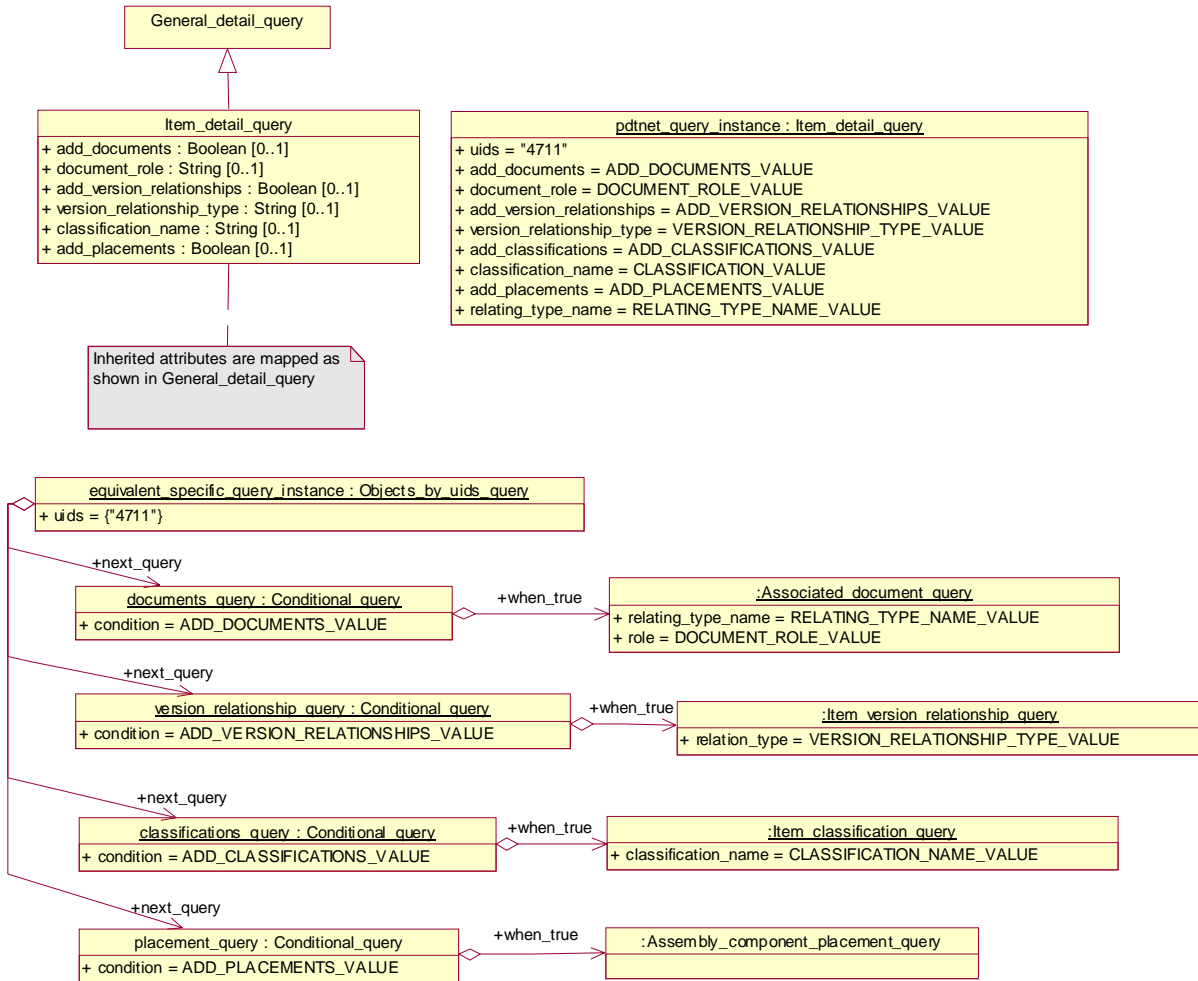


Figure 9.93 - Definition, sample instance and equivalent specific query instance of the Item\_detail\_query

### 9.8.6 Item\_selection\_query

The Item\_selection\_query selects objects of class Item and includes related Item\_version and Design\_discipline\_item\_definition objects.

#### Parameters

- name : String [0..1]
- id : String [0..1]
- version\_id : String [0..1]
- classification\_name : String [0..1]

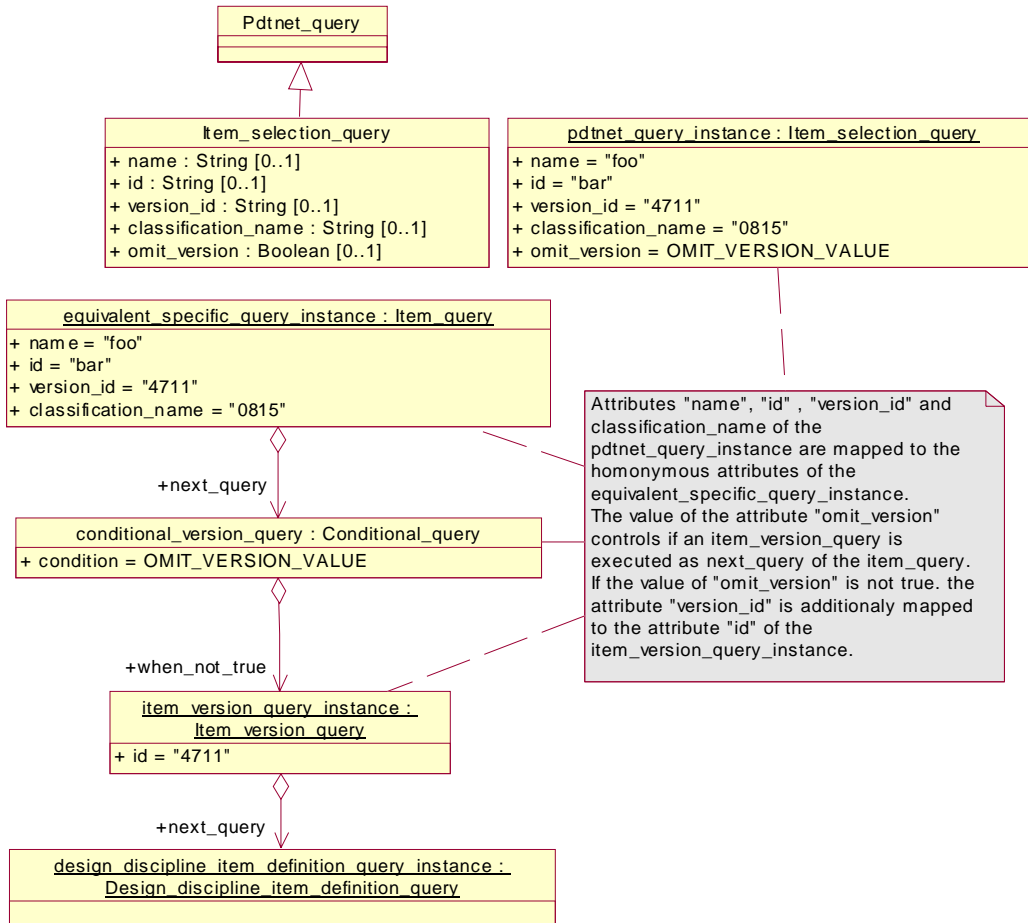


Figure 9.94 - Definition, sample instance and equivalent specific query instance of the Item\_selection\_query

### 9.8.7 Item\_traversal\_query

The Item\_traversal\_query traverses from a Design\_discipline\_item\_definition object to the next higher or next lower Design\_discipline\_item\_definition object in an assembly structure.

#### Parameters

- uid : UID
- inverse\_direction : Boolean [0..1]

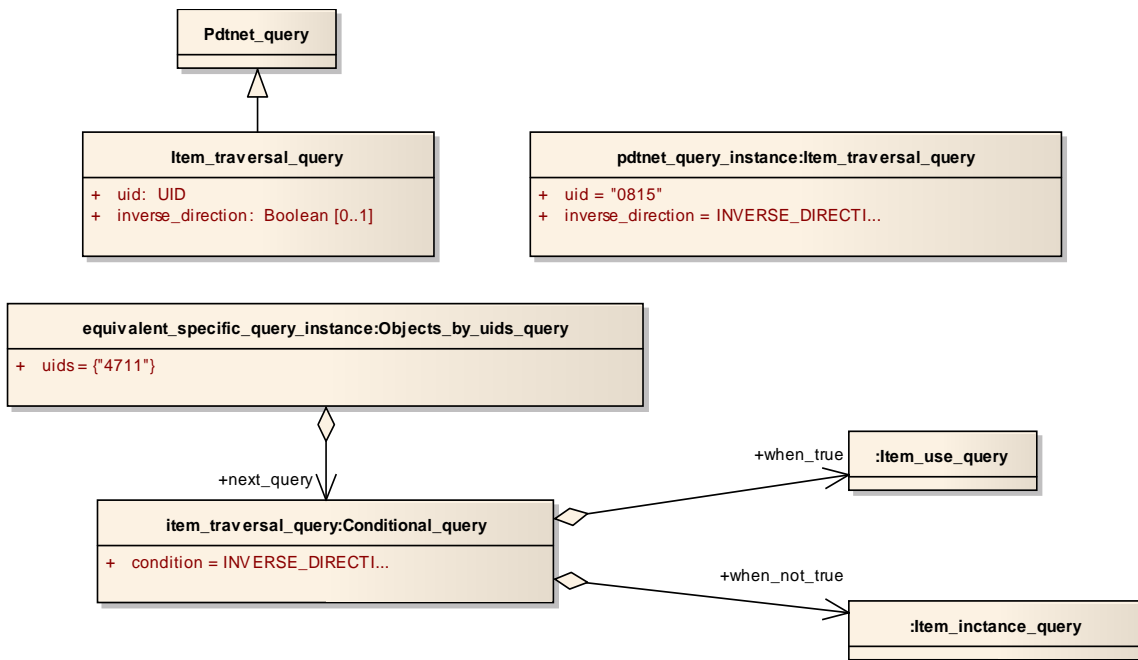


Figure 9.95 - Definition, sample instance and equivalent specific query instance of the Item\_traversal\_query

### 9.8.8 Product\_detail\_query

The Product\_detail\_query returns detail information of a Complex\_product object selected by a uid.

#### Parameters (without inherited)

- add\_configurations : Boolean [0..1]
- configuration\_type : String [0..1]
- add\_documents : Boolean [0..1]
- document\_role : String [0..1]

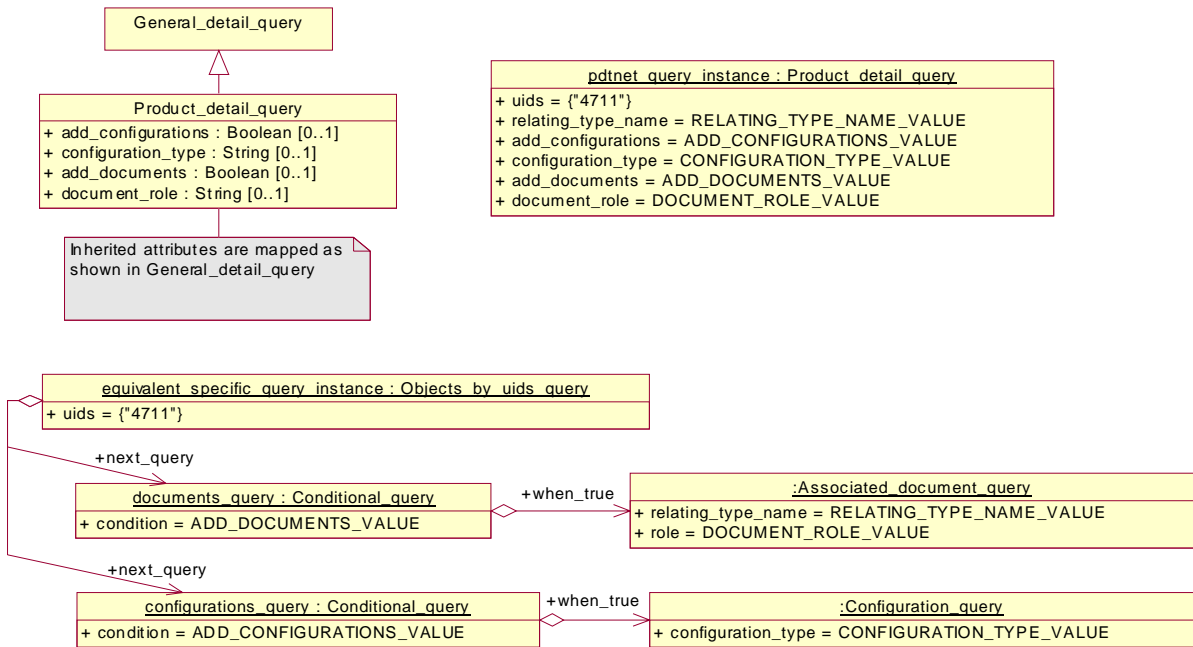


Figure 9.96 - Definition, sample instance and equivalent specific query instance of the Product\_detail\_query

### 9.8.9 Product\_selection\_query

The Product\_selection\_query selects objects of class Product\_class and includes Product\_function\_component\_select related via Class\_structure\_relationship objects.

#### Parameters

- name : String [0..1]
- id : String [0..1]

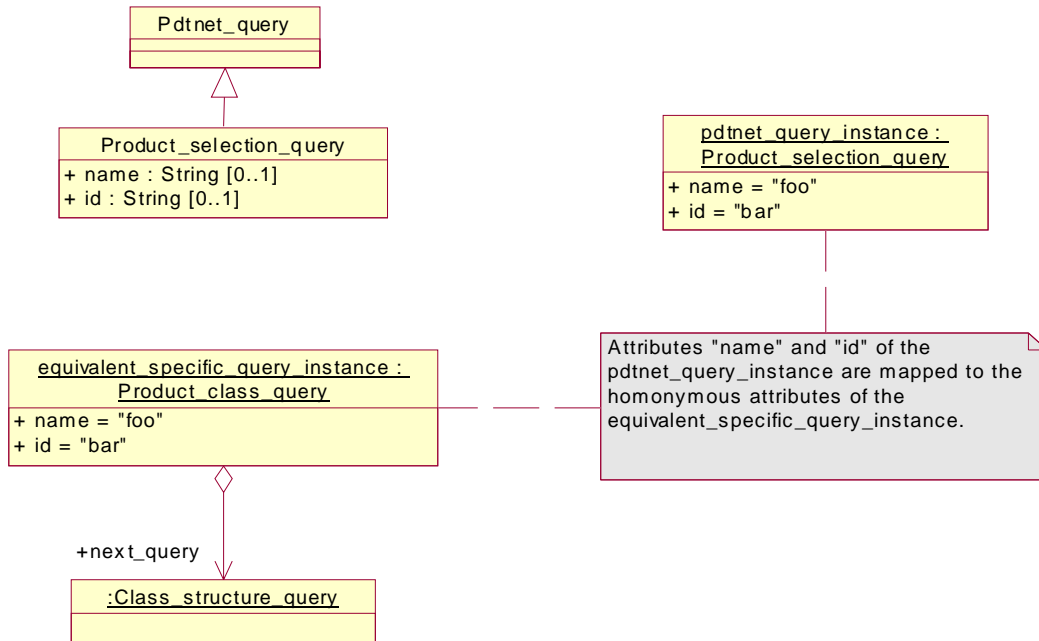


Figure 9.97 - Definition, sample instance, and equivalent specific query instance of the Product\_selection\_query

### 9.8.10 Product\_traversal\_query

The Product\_traversal\_query traverses from a Complex\_product object selected by its uid via Product\_structure\_relationship or Alternative\_solution\_objects to related Complex\_product or Item\_instance objects in a product structure.

#### Parameters

- uid : UID
- relation\_type : String [0..1]

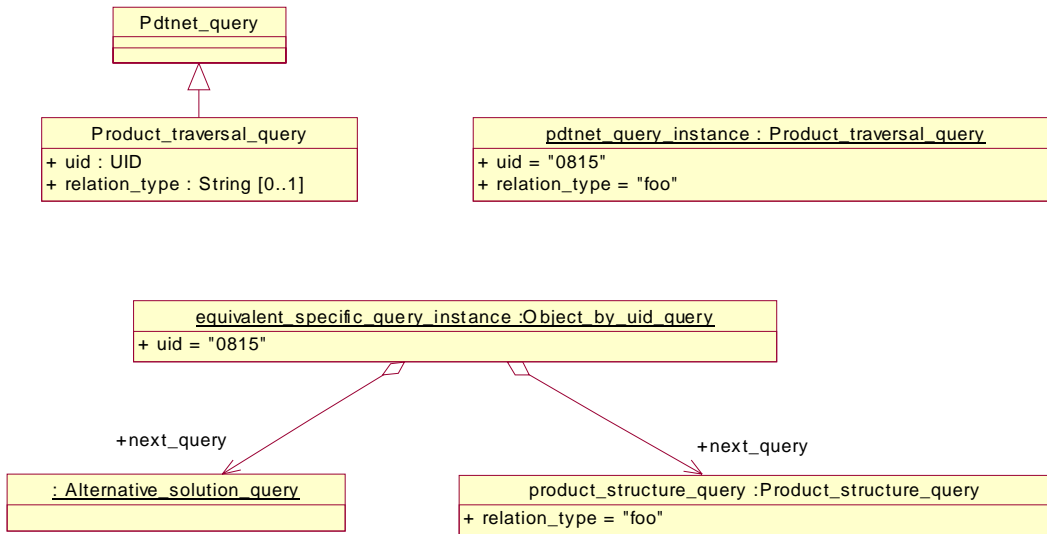


Figure 9.98 - Definition, sample instance, and equivalent specific query instance of the `Product_traversal_query`

### 9.8.11 `Work_order_selection_query`

The `Work_order_selection_query` selects objects of class `Work_order`.

#### **Parameters**

- `Id`: String [0..1]
- `work_order_type`: String [0..1]
- `version_id`: String [0..1]
- `classification_role`: String [0..1]
- `classification_id`: String [0..1]
- `description`: String [0..1]



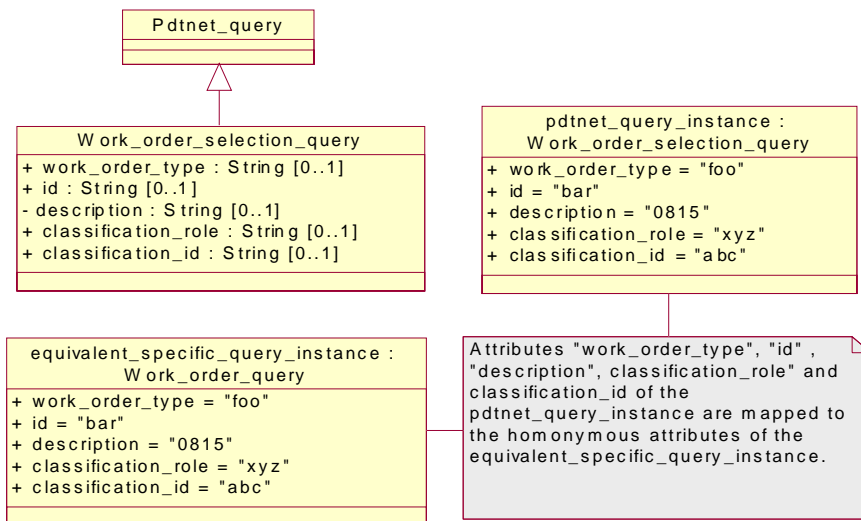


Figure 9.99 - Definition, sample instance and equivalent specific query instance of the Work\_order\_selection\_query

### 9.8.12 Work\_order\_detail\_query

The Work\_order\_detail\_query returns detail information of a Work:order object selected by a uid.

#### Parameters (without inherited)

- add\_documents : Boolean [0..1]
- document\_role : String [0..1]

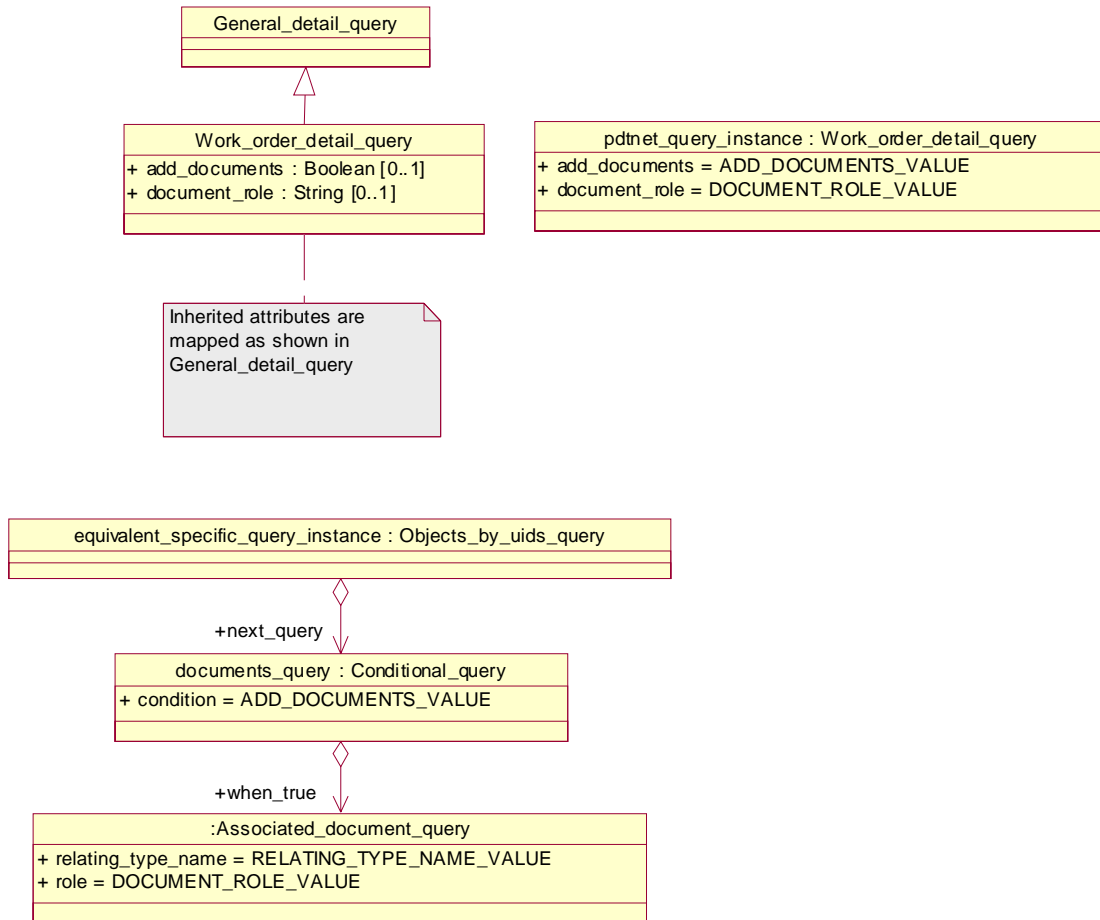


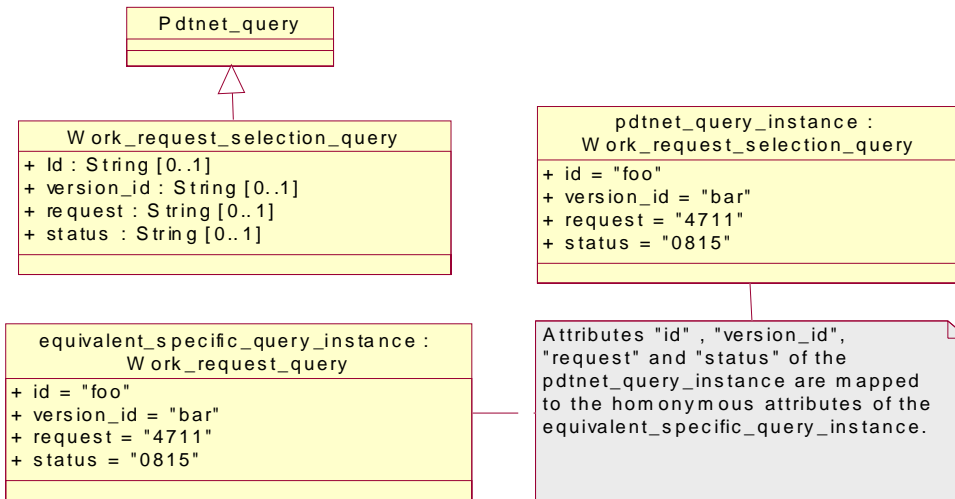
Figure 9.100 - Definition, sample instance and equivalent specific query instance of the Work\_order\_detail\_query

### 9.8.13 Work\_request\_selection\_query

The Work\_order\_selection\_query selects objects of class Work\_order.

#### Parameters

- Id : String [0..1]
- request: String [0..1]
- version\_id: String [0..1]
- status: String [0..1]



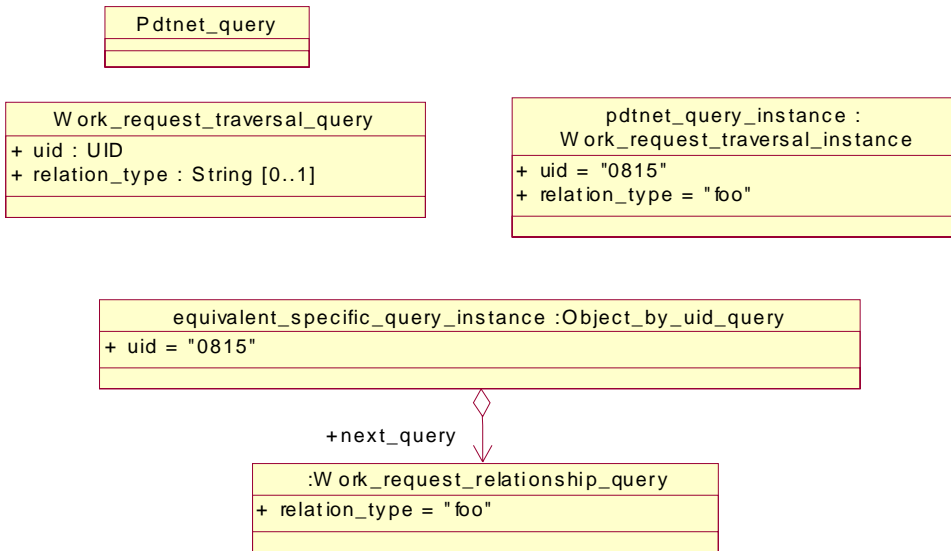
**Figure 9.101 - Definition, sample instance and equivalent specific query instance of the Work\_request\_selection\_query**

### 9.8.14 Work\_request\_traversal\_query

The Work\_request\_traversal\_query traverses from Work\_request objects via Work\_request\_relationship objects to other Work\_request objects.

#### **Parameters**

- uid : UID
- relation\_type: String [0..1]



**Figure 9.102 - Definition, sample instance and equivalent specific query instance of the `Work_request_traversal_query`**

### 9.8.15 `Work_request_detail_query`

The `Work_order_detail_query` returns detail information of a `Work:order` object selected by a `uid`.

**Parameters (without inherited)**

- `add_documents` : Boolean [0..1]
- `document_role` : String [0..1]

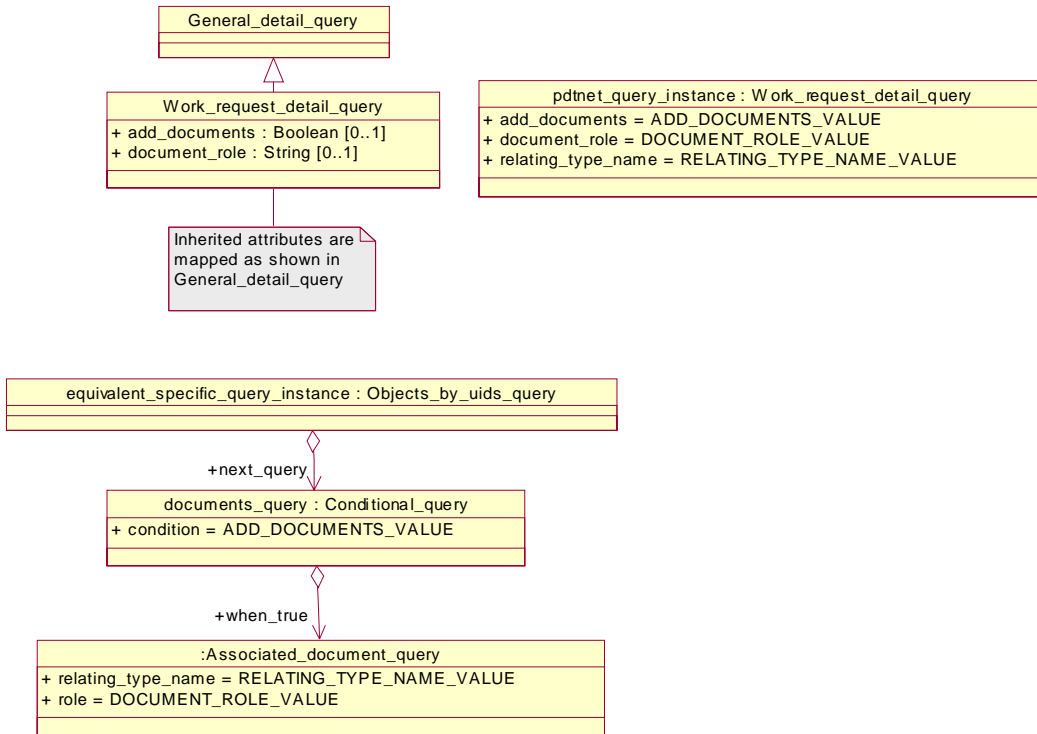


Figure 9.103 - Definition, sample instance and equivalent specific query instance of the Work\_request\_detail\_query

## 9.9 Message Queries Conformance Point

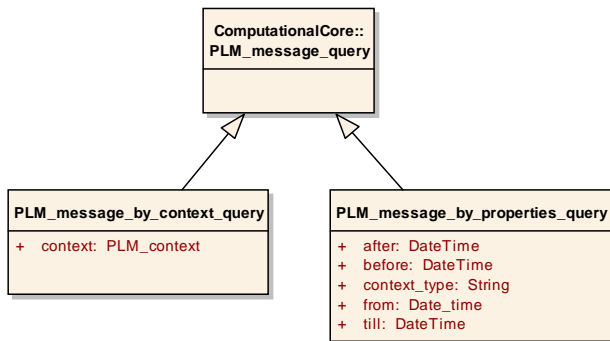


Figure 9.104 - Message Queries

The Message Queries Conformance Point defines queries derived from the abstract class PLM\_message\_query that can be used as parameters for the query\_messages operation of the PLM\_message\_connection interface.

### 9.9.1 Class Message\_by\_context\_query

The class Message\_by\_context\_query can be used to identify PLM\_message objects unambiguously by their contexts.

#### **Base**

- PLM\_message\_query

#### **Parameters**

- PLM\_context: PLM\_context [0..\*]

### 9.9.2 Class Message\_by\_properties\_query

The class Message\_by\_properties\_query can be used to query PLM\_message objects by their properties.

#### **Base**

- PLM\_message\_query

#### **Parameters**

- omit\_container : Boolean [0..1]  
If the omit\_container is TRUE in all PLM\_message elements of the result list of the query\_messages operation, the contained PLM\_core\_container is a NIL object.
- context\_type: Type [0..1]  
Selects messages if their contained context is an instance of the given type.
- before: dateTime [0..1]  
Selects messages if their server incoming time is less than the given dateTime value.
- till: dateTime [0..1]  
Selects messages if their server incoming time is less than or equals the given dateTime value.
- from: dateTime [0..1]  
Selects messages if their server incoming time is greater than or equals the given dateTime value.
- after: dateTime [0..1]  
Selects messages if their server incoming time is greater than the given dateTime value.



# 10 WebServices PSM

## 10.1 Overview (informative)

In the following sections a projection of the PIM into the platform specific model (PSM) with an execution infrastructure given by XML is defined. The projection is done via an enrichment of the model by a customized UML profile for XML Schema. This UML profile is given here for informal purposes.

## 10.2 UML Profile for XML Schema (informative)

To enrich the UML Informational PIM for XML Schema representation a UML profile is used. A UML profile has three key items namely stereotypes, tagged values called properties and constraints.

### 10.2.1 UML Model

On the entire UML model level the stereotype <<XSDschema >> is applied. It can have the following tagged values:

**Table 10.1 - Stereotype <<XSDschema>>**

Property	Value	Description
targetNamespace	namespace URI	The URI that uniquely identifies this schema's namespace.
elementFormDefault	qualified unqualified	Specifies whether elements are qualified or unqualified.
attributeFormDefault	qualified unqualified	Specifies whether attributes are qualified or unqualified.
version	string value	The version of this schema.
modelGroup	all sequence choice none omitComplexType	Specifies the content model used for generating complexType definitions.
globalElement	true false	Specifies if global element declarations are created for complex types.
attributeMapping	element attribute	Specifies the mapping for UML attributes.
roleMapping	element attribute	Specifies the mapping for roles of UML associations.
anonymousRole	true false	Specifies if role names of UML attributes are mapped to elements.
anonymousType	true false	Specifies if the types of UML attributes are mapped to elements.



**Table 10.1 - Stereotype <<XSDschema>>**

typeContainment	true false	Specifies if types are contained instead of referencing them.
elementNamingMapping	firstLetterUpperCase firstLetterLowerCase upperCamelCase lowerCamelCase omitElement	Specifies the naming for elements.
attributeNamingMapping	firstLetterUpperCase firstLetterLowerCase upperCamelCase lowerCamelCase omitAttribute	Specifies the naming for attributes.

The four stereotypes XSDschema, XSDtranslatableString, XSDelement, and XSDattribute form a hierarchy in that order. The derived stereotypes need to redefine only those values of the named properties that require new values.

**Example**

```

Model "PLM_services"
Stereotype << XSDschema >>
targetNamespace = urn:omg.org/plm20/informational/core
elementFormDefault = qualified
attributeFormDefault = unqualified
version = 2.0
modelGroup = sequence
globalElement = false
attributeMapping = element
roleMapping = element
anonymousRole = false
anonymousType = false
typeContainment = false
elementNamingMapping = firstLetterUpperCase
attributeNamingMapping = firstLetterLowerCase

```

```

<xs:schema
targetNamespace='urn:omg.org/plm20/informational/core'
xmlns:ns2='urn:omg.org/plm20/informational/core'
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
version="2.0">

```

**10.2.2 UML Package**

On the UML package “Multi\_language\_support” the stereotype << XSDtranslatableString >> is applied. It doesn't have any tagged values.

XML provides its own mechanism to specify the language used in the contents and attribute values of any element in an XML document, the predefined attribute xml:lang. Based on this an XML specific concept has been developed to map the multi language support for string values of the PIM model.

If the UML Interface “String\_select” is used by any UML composition, the type “Translatable\_string” is used in XML instead. Therefore the predefined complex types “Translatable\_string,” “Translation,” and “Translations” are introduced in the XML schema.

**Table 10.2 - Stereotype << XSDtranslatableString>>**

Can be applied to	UML package
-------------------	-------------

**Example**

Package "Multi language support"

Stereotype << XSDtranslatableString >>

```

<xs:complexType name="Item">
  ...
  <xs:element name="Name" type="Translatable_string"/>
  ...
</xs:complexType>

<xs:complexType name="Translatable_string">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="translations" type="xs:IDREF" use="optional"/>
      <xs:annotation>
        <xs:documentation>REFERENCE TO Translations</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute ref="xml:lang" use="optional"/>
  </xs:extension>
</xs:simpleContent>
</xs:complexType>

<xs:complexType name="Translation">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute ref="xml:lang" use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<xs:complexType name="Translations">
  <xs:complexContent>
    <xs:extension base="PLM_root_object">
      <xs:sequence>
        <xs:element name="Translation" type="Translation" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

The mapping of instance values from UML to XML is as follows.

**Table 10.3 - Mapping of instance values from UML to XML**

UML	XML
Default_language_string	Translatable_string
Multi_language_string .primary_language_dependent_string .String_with_language.contents	Translatable_string
Multi_language_string .primary_language_dependent_string .String_with_language .language_specification .Language.language_code	Translatable_string /@xml:lang
Multi_language_string .primary_language_dependent_string .String_with_language .language_specification .Language.country_code	Translatable_string /@xml:lang
Multi_language_string .additional_language_dependent_string .String_with_language.contents	Translation
Multi_language_string .additional_language_dependent_string .String_with_language .language_specification .Language.language_code	Translation /@xml:lang
Multi_language_string .additional_language_dependent_string .String_with_language .language_specification .Language.country_code	Translation /@xml:lang

### 10.2.3 UML Classes

On each UML class the stereotype <<XSDcomplexType>> is applied. It can have the following tagged values:

**Table 10.4 - Stereotype <<XSDcomplexType>>**

Can be applied to	UML class	
Property	Value	Description
modelGroup	all sequence choice multiChoice omitComplexType	Specifies the content model used for generating this complexType definition.

**Table 10.4 - Stereotype <<XSDcomplexType>>**

globalElement	true false	Specifies if a global element declaration is created for this complexType.
attributeMapping	element attribute	Specifies the mapping for UML attributes within this complexType.
roleMapping	element attribute	Specifies the mapping for roles of UML associations within this complexType.
anonymousRole	true false	Specifies if the role names of UML attributes are mapped to elements within this complex type.
anonymousType	true false	Specifies if the types of UML attributes are mapped to elements within this complex type.
typeContainment	true false	Specifies if types are contained instead of referencing them within this complex type.
elementNamingMapping	firstLetterUpperCase firstLetterLowerCase upperCamelCase lowerCamelCase omitElement	Specifies the naming for elements within this complex type.
attributeNamingMapping	firstLetterUpperCase firstLetterLowerCase upperCamelCase lowerCamelCase omitAttribute	Specifies the naming for attributes within this complex type.

Each of the above named properties shall apply to all UML classes, attributes, associations, and compositions that do not have an own stereotype overwriting these values.

### Generalization

Only single inheritance is treated by the UML to XML Schema mapping. This is sufficient since the PIM UML model does not contain any multiple inheritance. Each subclass will be a complexType with complexContent and extension base="superclass". Abstract classes are mapped to complex types that are abstract.

### Example

```

Class "PLM_container"
Stereotype << XSDcomplexType >>
modelGroup          = multiChoice
globalElement       = true

    <xs:element name="PLM_container" type="PLM_container"/>
    <xs:complexType name="PLM_container">
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            ...
        </xs:choice>
    </xs:complexType>

```

## 10.2.4 UML Interfaces

UML interfaces are not treated by the UML to XML Schema mapping since the interfaces are only referenced by other classes. These references are mapped to XML schema references of type IDREF and IDREFS, which point to the underlying types of an interface.

## 10.2.5 UML Attributes, Associations and Compositions

On each UML attribute, association, and composition the stereotypes << XSDelement >> or << XSDattribute >> are applied. They can have the following tagged values:

**Table 10.5 - Stereotype <<XSDelement>>**

Can be applied to	UML attribute, UML association, UML composition	
<b>Property</b>	<b>Value</b>	<b>Description</b>
position	integer value	Causes the elements to be ordered within a sequence model group of the containing complexType.
anonymousRole	true false	Specifies if the role name of a UML attribute is mapped to an element.
anonymousType	true false	Specifies if the type of a UML attribute is mapped to an element.
typeContainment	true false	Specifies if the type is contained instead of referencing it.
elementNamingMapping	firstLetterUpperCase firstLetterLowerCase upperCamelCase lowerCamelCase omitElement	Specifies the naming for this element.

### Example

#### Attribute "relation type"

```
Stereotype << XSDelement >>
position           = 03
anonymousRole     = false
anonymousType     = true
typeContainment   = true
```

#### Composition "description"

```
Stereotype << XSDelement >>
position           = 02
anonymousRole     = false
anonymousType     = true
typeContainment   = true
```

Composition "change"

```
Stereotype << XSDelement >>
position           = 04
anonymousRole     = true
anonymousType     = false
typeContainment   = true
```

Association "related"

```
Stereotype << XSDelement >>
position           = 01
anonymousRole     = false
anonymousType     = false
typeContainment   = false
```

```
<xs:complexType name="Item_version_relationship">
  <xs:complexContent>
    <xs:extension base="PLM_object">
      <xs:sequence>
        <xs:element name="Related" type="xs:IDREF">
          <xs:annotation>
            <xs:documentation>REFERENCE TO Item_version</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="Description" type="Translatable_string" minOccurs="0"/>
        <xs:element name="Relation_type" type="xs:string"/>
        <xs:element name="Change" type="Change" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

**Table 10.6 - Stereotype <<XSDDattribute>>**

Can be applied to	UML attribute, UML association, UML composition	
<b>Property</b>	<b>Value</b>	<b>Description</b>
attributeType	qualified name	Specifies the type of the attribute.
use	prohibited optional required fixed	Specifies the use of the attribute.
attributeNamingMapping	firstLetterUpperCase firstLetterLowerCase upperCamelCase lowerCamelCase omitAttribute	Specifies the naming for this attribute.

**Example**

Attribute "uid"

```
Stereotype << XSDDattribute >>
```

```

attributeType      = xs:ID
use                = required

<xs:complexType name="PLM_object" abstract="true">
  <xs:attribute name="uid" type="xs:ID" use="required"/>
</xs:complexType>

```

Most of the UML attributes, associations, and compositions are mapped to elements in the XML Schema and a position of these elements are needed if the modelGroup is a sequence. This is done by applying a position value to a UML attribute, association, or composition.

If the type of the UML attribute is a data type, it is mapped to a corresponding primitive data type of the XML Schema Definition:

UML datatype	XSD primitive type
String	xs:string
Double	xs:double
Boolean	xs:boolean
Integer	xs:integer
UID	xs:ID

The multiplicity of a UML attribute, association, or composition is mapped to the corresponding multiplicity in the XML schema. For elements the values minOccurs and maxOccurs are used, for attributes the value use.

### 10.3 Informational viewpoint with applied stereotypes (normative)

The Informational viewpoint with applied stereotypes is given in a separate OMG document in XMI that is normative. Additional informative description is given below.

#### 10.3.1 Informational Core

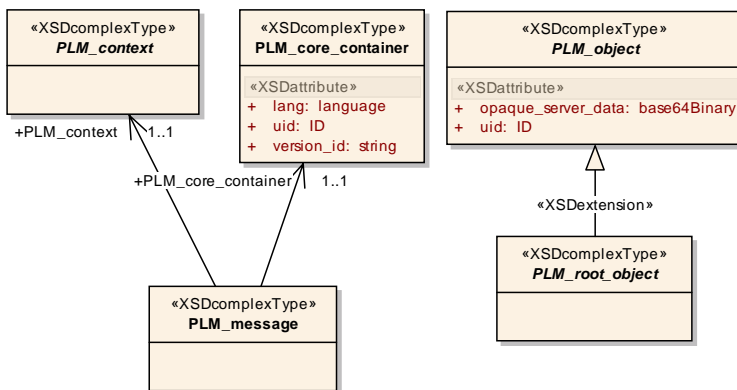


Figure 10.1 - Informational Core

## 10.4 Computational viewpoint with applied stereotypes (normative)

The Computational viewpoint with applied stereotypes is given in a separate OMG document in XMI that is normative. Additional informative description is given below.

### 10.4.1 Computational Core

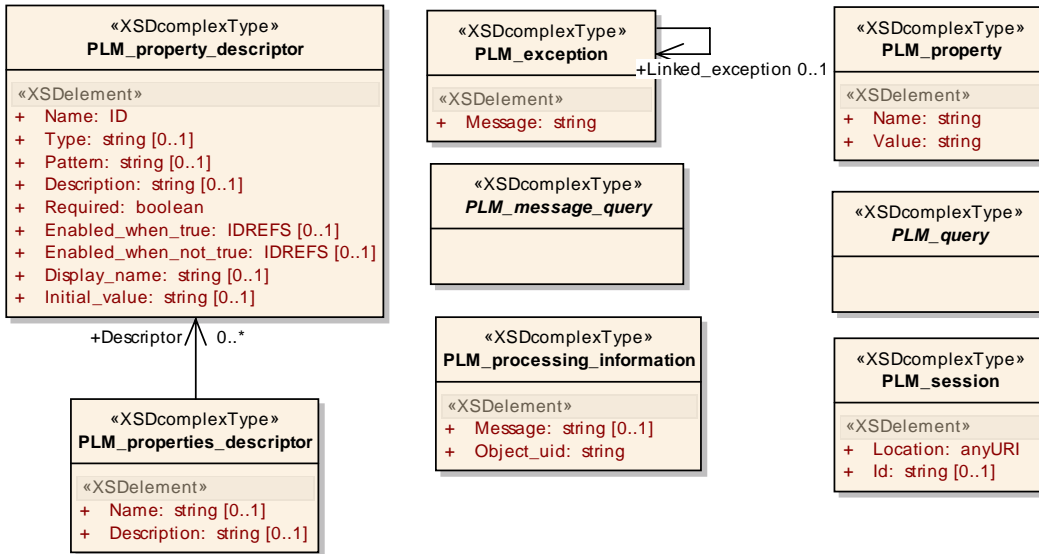


Figure 10.2 - Computational Core



## 10.4.2 Exceptions

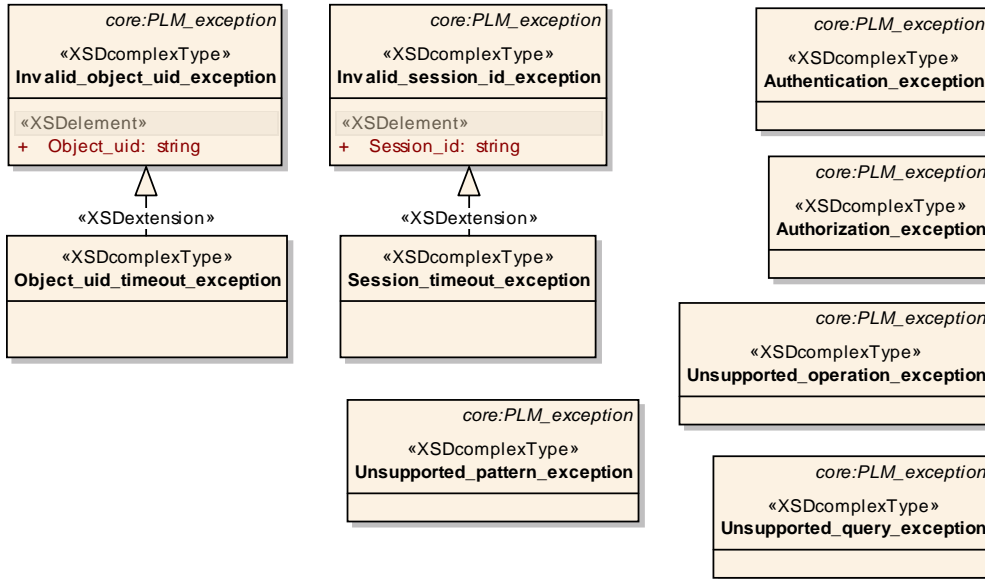


Figure 10.3 - Exceptions

### 10.4.3 Generic Queries

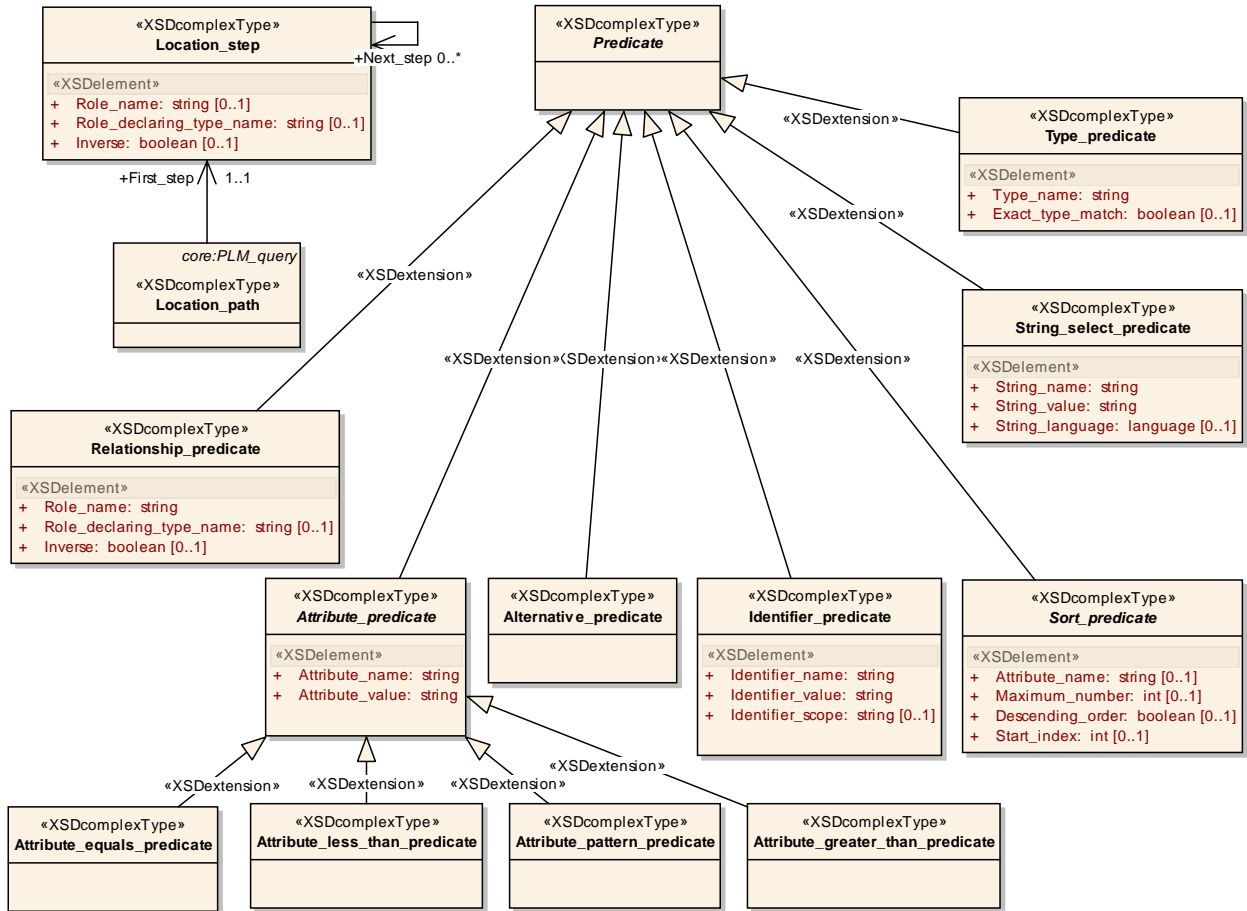


Figure 10.4 - Generic Queries

## 10.4.4 Informations

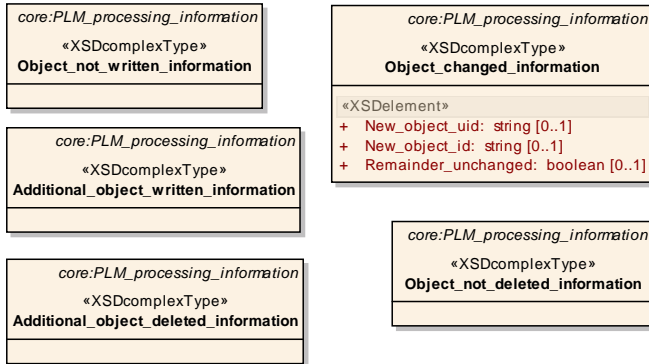


Figure 10.5 - Informations

## 10.4.5 Parameters



Figure 10.6 - Parameters

## 10.4.6 PDTnet Queries

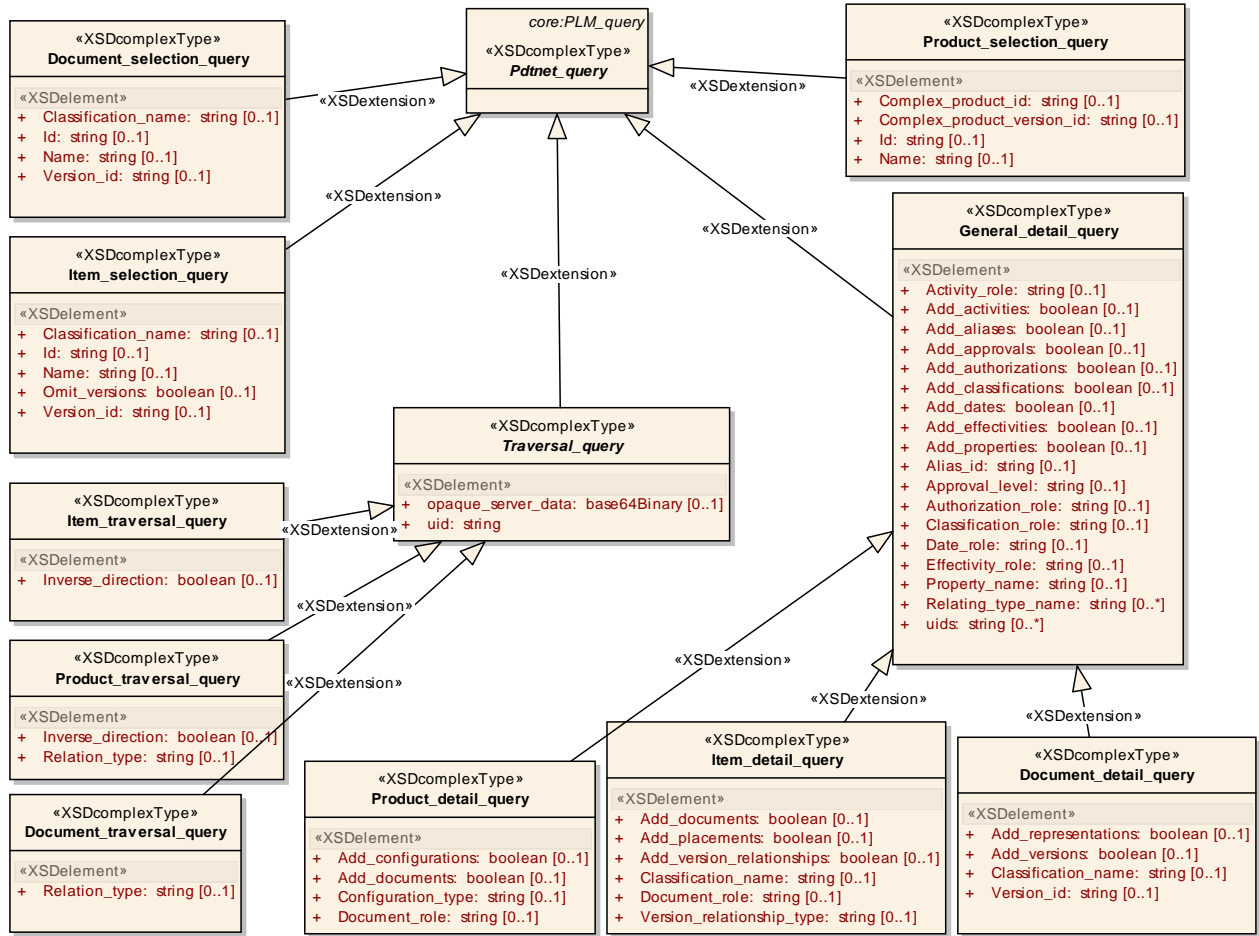


Figure 10.7 - PDTnet Queries

### 10.4.7 Proxy Queries

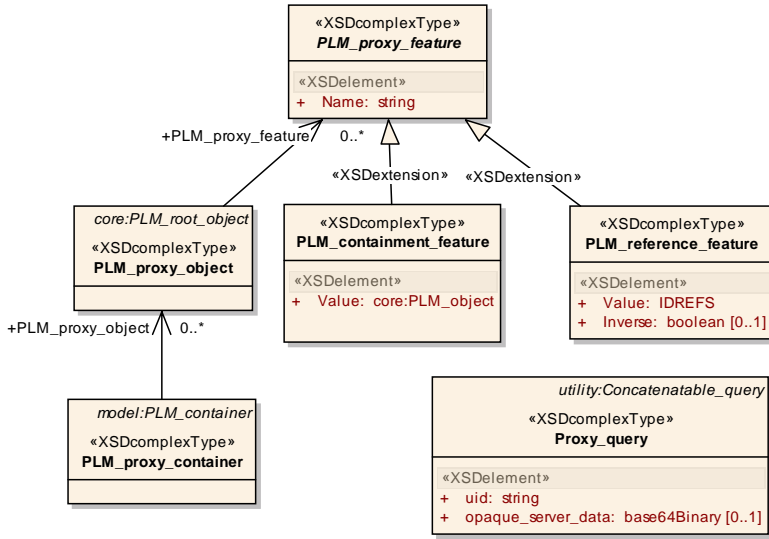


Figure 10.8 - Proxy Queries

### 10.4.8 Schema Infos

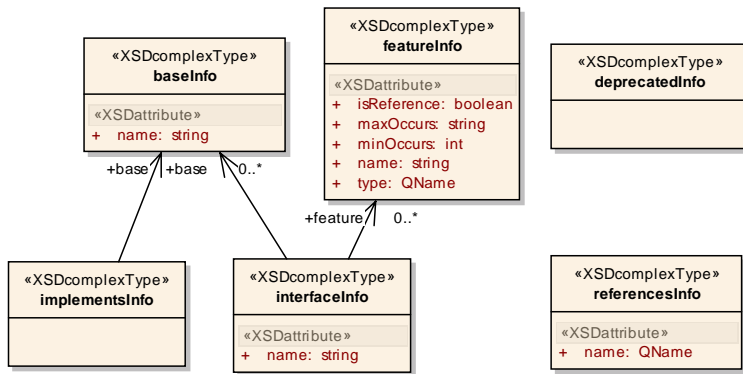


Figure 10.9 - Schema Info

### 10.4.9 Specific Queries

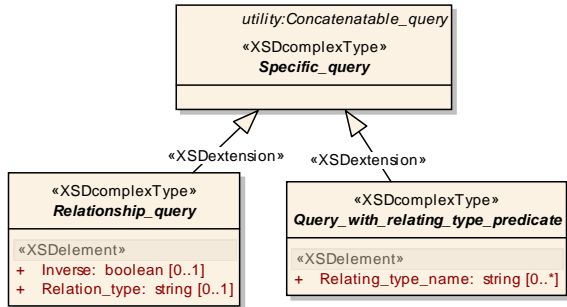


Figure 10.10 - Specific Queries

### 10.4.10 Utility Queries

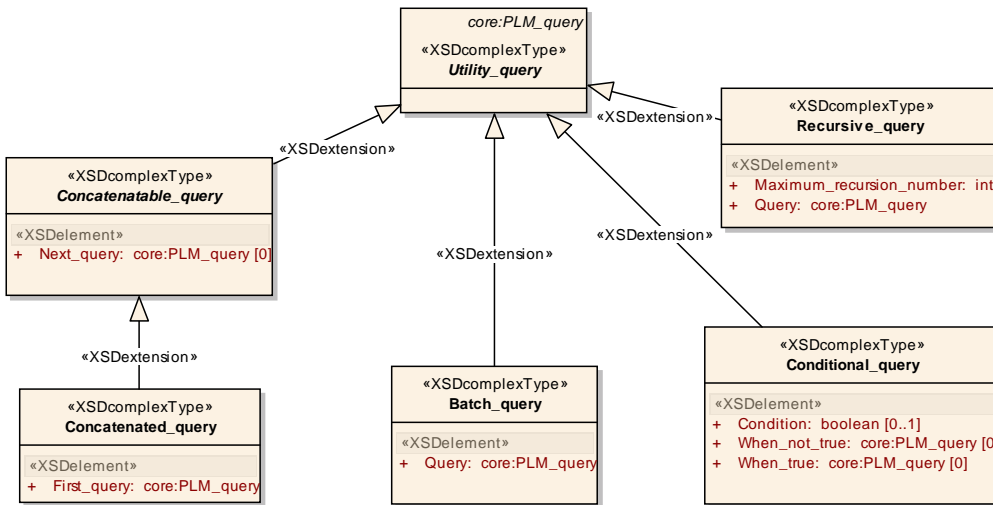


Figure 10.11 - Utility Queries

### 10.4.11 XPath Queries

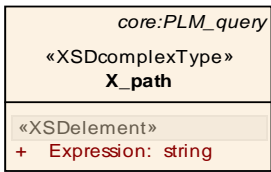


Figure 10.12 - XPath Queries

## 10.5 PLM Services Web services (normative)

The PLM Services Computational Viewpoint with applied stereotypes is given in a separate OMG document in XMI that is normative. Additional informative description is given below.

The Computational Viewpoint of the Web service PSM is defined in the Web Services Description Language (WSDL) 1.1. The WSDL imports the XML Schema defined by the Informational Viewpoint. The Web service PSM contains definitions of two ports: `PLM_connection_factory` and `PLM_connection`. Due to the fact that Web services cannot transfer object references as parameters or results of operations, the syntax and semantic of the operation `get_connection()` has changed in comparison with the PIM. In the Web service PSM `get_connection` returns a `PLM_session` instance that contains a `Session_context` and a `Location` element. The `Session_context` identifies a session and has to be added as a soap header element to each operation request to a `PLM_connection` port for this session. The optional `Location` element overrides the address element of the `PLM_connection` port in the WSDL. The PIM object types `PLM_resource_adapter` and `PLM_object_factory` have no counterpart in the Web service PSM.

### 10.5.1 Connection Factory

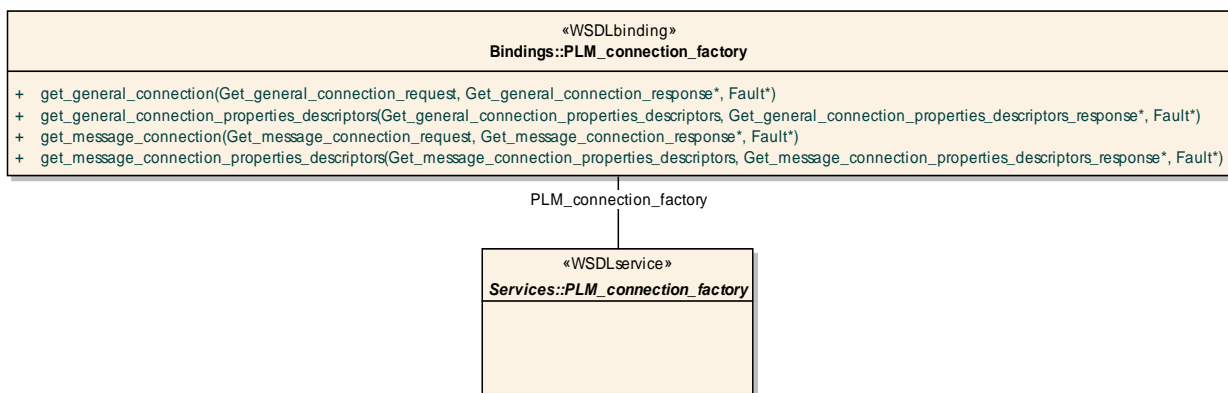


Figure 10.13 - Connection Factory

## 10.5.2 General Connection

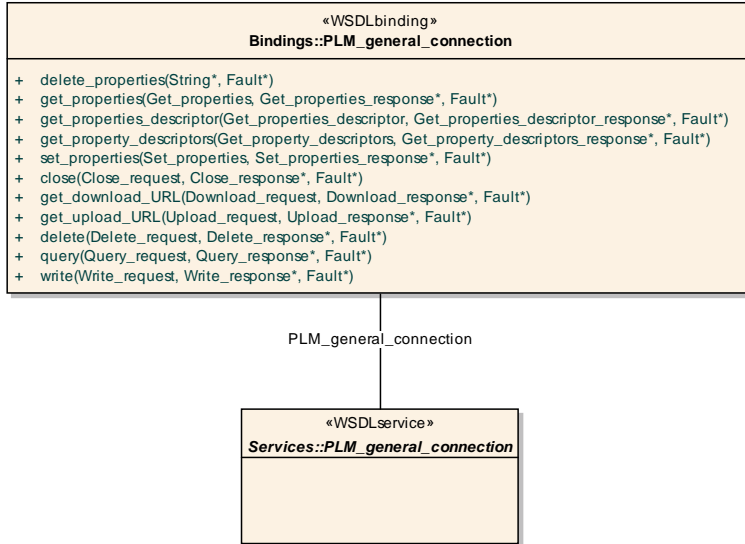


Figure 10.14 - General Connection WSDL Binding

## 10.5.3 Message Connection

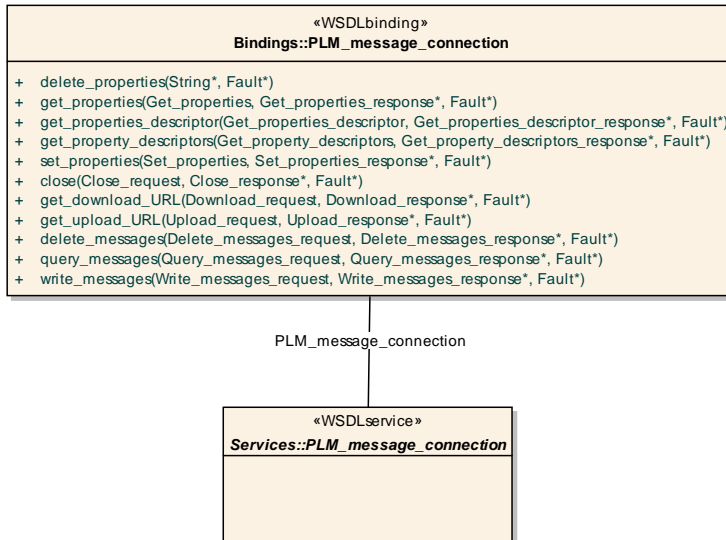


Figure 10.15 - Message Connection WSDL Binding



## 10.6 Query Examples (informative)

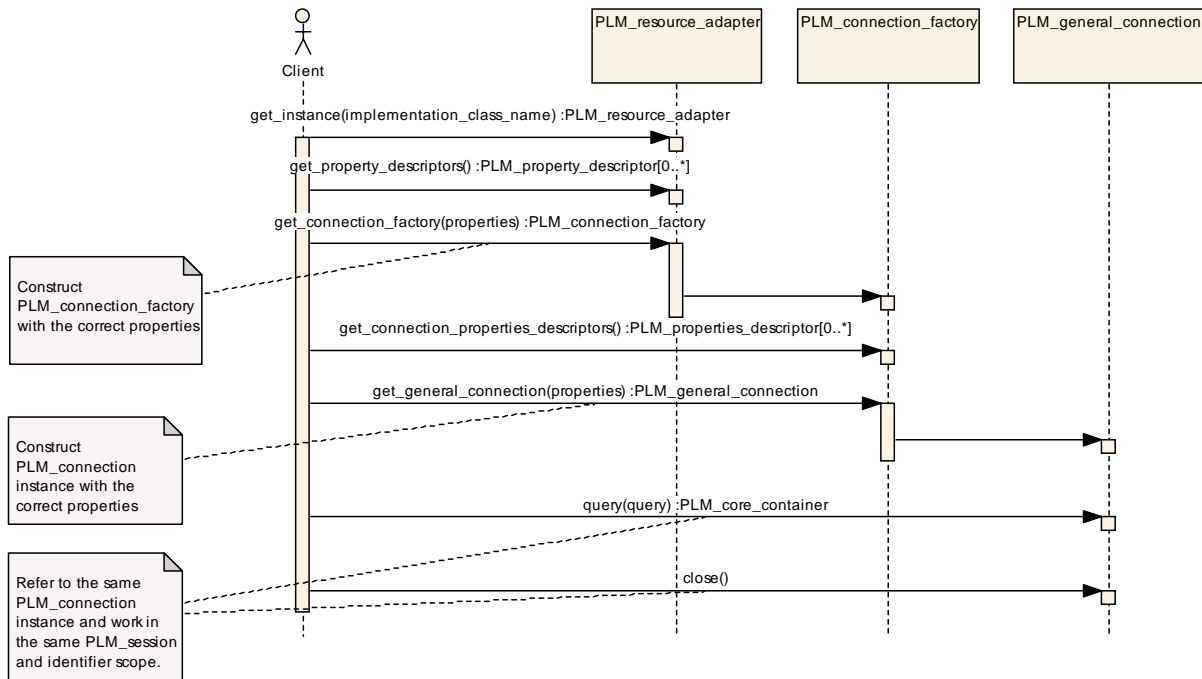


Figure 10.16 - Sequence diagram of a PLM session

### 10.6.1 Generic Queries Conformance Point Example

Query for all Item\_version objects with id='bar' of Item objects with Name='foo.'

```

<Query xsi:type="Location_path">
  <First_step>
    <Role_name>item</Role_name>
    <Predicate xsi:type="Attribute_predicate">
      <Attribute_name>name</Attribute_name>
      <Attribute_value>foo</Attribute_value>
    </Predicate>
    <Next_step>
      <Role_name>item_version</Role_name>
      <Predicate xsi:type="Attribute_predicate">
        <Attribute_name>id</Attribute_name>
        <Attribute_value>bar</Attribute_value>
      </Predicate>
    </Next_step>
  </First_step>
</Query>
  
```

### 10.6.2 XPath Queries Conformance Point Example

Query for all Item\_version objects of Item objects with id='foo.'

```

<Query xsi:type="X_path">
  <Expression> //Item[Id='foo']/Item_version </Expression>
</Query>
  
```

### 10.6.3 PDTnet Queries Conformance Point Examples

Assembly\_structure\_query for all Design\_discipline\_item\_definition objects of Item\_version objects with id='4711' of Item objects with name='bar' and name language='en-US'

```
<Query xsi:type="Item_query">
  <Name>bar</Name>
  <Name_language>en-US</Name_language>
  <Next_query xsi:type="Item_version_query">
    <Id>4711</Id>
    <Next_query xsi_type="Design_discipline_item_definition">
      <Next_query xsi:type="Assembly_structure_query"/>
    </Next_query>
  </Next_query>
</Query>
```

Assembly\_structure\_query for Design\_discipline\_item\_definition with an initial\_context with application\_domain='mechanical design' and life\_cycle\_stage='design' of all Item\_version objects of Item objects with id='foo'. The result is extended by associated Date\_time, Organization and Property\_value objects.

```
<Query xsi:type="Item_query">
  <Id>foo</Id>
  <Next_query xsi:type="Item_version_query">
    <Next_query xsi_type="Design_discipline_item_definition">
      <Application_domain>mechanical design</Application_domain>
      <Life_cycle_stage>mechanical design</Life_cycle_stage>
      <Next_query xsi:type="Assembly_structure_query">
        <Next_query xsi:type="Associated_date_time_query"/>
        <Next_query xsi:type="Associated_organization_query"/>
        <Next_query xsi:type="Associated_property_query"/>
      </Next_query>
    </Next_query>
  </Next_query>
</Query>
```

Assembly\_structure\_query for the PLM\_object with uid='assembly123' (which should be an Assembly\_definition).

```
<Query xsi:type="Object_by_uid_query">
  <uid>assembly123</uid>
  <Next_query xsi:type="Assembly_structure_query"/>
</Query>
```

## 10.7 Security Features (informative)

### 10.7.1 Introduction

The intention is to define the use of existing standards to realize security within SOAP communication and not the definition of a new security standard to enable secure communication between a sender and recipient.

### 10.7.2 Goals and Requirements

The security features defined here are based on the WS-Security Core Specification 1.1 OASIS (Organization for the Advancement of Structured Information Standards) standard [WS-SECURITY].

The WS-Security standard provides a flexible set of mechanisms to enable SOAP message security together with a high level of interoperability. Therefore this standard offers the possibility to integrate further WS-Security standards and other security standards like SAML etc., which are based on the WS-Security Header. The WS-Security standard also describes mechanisms to use XML-Encryption and XML-Signature, which are W3C (World Wide Web Consortium) standards to encrypt and sign XML-Documents.

### 10.7.3 Non-Goals

The following topics are outside the scope of this document:

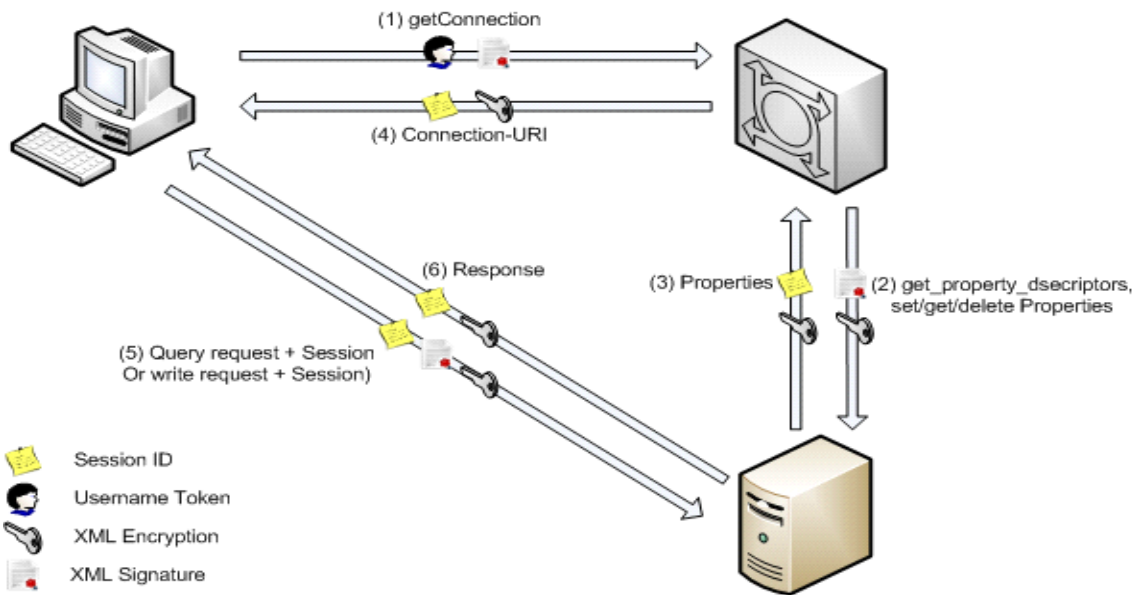
- Key derivation or key management
- Security policies

### 10.7.4 Message Protection Mechanisms

#### 10.7.4.1 Outline

The following figure gives an overview of the recommended security mechanisms. The figure shows the several topics for the SOAP security headers of the different calls.

1. **getConnection**  
This call has to deal with UsernameToken and Digital Signature at the authentication phase to get a “connection.” The Digital Signature is an additional security level to ensure the identity of the sender.
2. **get\_property\_descriptors**  
The Digital Signature performs the same task as in the “getConnection” call. The encryption is necessary to encrypt the properties send in the SOAP body.
3. **Properties**  
The response contains the generated Session ID (in the SOAP body or in the SOAP header) and additional data that should be encrypted.
4. **Connection URI**  
The same features as explained in point 3.
5. **General Client Request**  
The request calls the received URI with the given Session ID. The Digital Signature is again the assertion for the identity of the sender. Furthermore the request/query in the SOAP body could be encrypted if the request contains security sensitive data.
6. **General Server Response**  
The content of the response in the SOAP body should be encrypted in order to protect the information from unauthorized persons.



**Figure 10.17 - Overview of recommended security features for PLM-Services**

#### 10.7.4.2 Https

Https can be used for point to point security to enable security on transport layer. As mentioned this is only a point to point security and will not secure the data above intermediaries. Therefore xml-encryption and xml-signature should be used rather than https to ensure confidentiality of sensitive data.

#### 10.7.4.3 Security-Header

The <wsse:Security> header bloc provides a mechanism for attaching security related information targeted at a specific recipient in the form of a SOAP actor/role. This may be either the ultimate recipient of the message or an intermediary. Consequently, elements of this type may be present multiple times in a SOAP message. An active intermediary on the message path MAY add one or more new sub-elements to an existing <wsse:Security> header bloc they are targeted for its SOAP node or it MAY add one or more new headers for additional targets [“Web Services Security: SOAP Message Security 1.1”].

##### **SOAP Security Header**

```

<S11:Envelope>
  <S11:Header>
    ...
    <wsse:Security S11:actor="..." S11:mustUnderstand="...">
      ...
    </wsse:Security>
    ...
  </S11:Header>
  ...
</S11:Envelope>

```

The SOAP actor/role is an optional attribute; however, every <wsse:Security> header must define his own different actor/role. The mustUnderstand defines whether the recipient of the message must be able to process the security header.

### **Recommendation for the use of actor/role / mustUnderstand**

- mustUnderstand should be “true” in signed messages or in messages that contain a username token.
- mustUnderstand should be “false” for all other cases (i.e., for encryption).
- actor/role is only necessary if there are several security headers in the SOAP message which shall be processed by different receivers (e.g., intermediaries and end point).
- In the case of the PLM Services, there is usually only one security header that is processed by one recipient, so actor/role can be omitted.

### **Session ID**

- The Session ID (part of the SOAP header) identifies the Client for the Server: its use is strongly recommended for productive environments.
- On non-productive implementations (demonstrators, pilots), a stateless Web Service implementation is possible. In this case, the use of a session id is not obligatory, nor the call of get\_general/message\_connection (the URL of the connection may be called directly), or a dummy Factory (not returning any session id) may be used to call get\_general/message\_connection.
- The way how the session id is generated by the server is left free to the implementation: SiteMinder session, Jsession (for J2EE Application Servers), ...

## **10.7.5 Authentication and Authorization Security**

This describes the secure exchange of authentication and authorization information (rather than the mechanisms for authentication or authorization itself) through the use of security tokens described in the WS-Security specification. These security tokens in combination with WS-Security enable a standardized way to place security related data in the SOAP header. This data can be a username and password directly placed in the SOAP header or a reference to security related data in the SOAP body.

The security related data are placed within the <wsse:Security> element in the SOAP-Header. Doing so, all receivers can identify and handle this security information. For more details, refer to the “Web Services Security Username Token Profile 1.1” [TOKEN].

### **10.7.5.1 Usage of Security Tokens (UsernameToken)**

The UsernameToken transports standard data (like username and password) within the WS-Security header for user authentication. It shall be mandatory when the consumer of the “PLM\_connection\_factory” calls the method “get\_connection.” Doing so, the “PLM\_connection\_factory” can identify the requestor by username and password before returning a connection.

The following examples or descriptions are partially from the “Web Services Security Username Token Profile 1.1” [TOKEN].

#### **XML-Schema**

```
<xs:element name="UsernameToken">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Username"/>
      <xs:element ref="Password" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```

    </xs:sequence>
    <xs:attribute name="Id" type="xs:ID"/>
    <xs:anyAttribute namespace="##other"/>
  </xs:complexType>
</xs:element>

```

The password contains an attribute “type,” which specifies the type of password being provided. This can be plain text or digest.

### **Textformat**

```

<wsse:UsernameToken>
  <wsse:Username>Sven</wsse:Username>
  <wsse:Password Type="wsse:PasswordText">Kennwort</wsse:Password>
</wsse:UsernameToken>

```

It is also possible to encrypt the password or the whole UsernameToken by using the xml encryption mechanisms provided by WS-Security.

The “nonce” and “created” can be used to define the “PasswordDigest.” Proper instructions how to use the “nonce” and “created” in combination with the SHA1(Secure Hash Algorithm 1) to form the “PasswordDigest” can be found in the “Web Services Security Username Token Profile 1.1”.[reference 03].

### **Digestformat**

```

<wsse:UsernameToken>
  <wsse:Username>Sven</wsse:Username>
  <wsse:Password Type="wsse:PasswordDigest">
    KE6QugOpkPyT3Eo0SEgT30W4Keg=</wsse:Password>
  <wsse:Nonce>5uW4ABku/m6/S5rnE+L7vg==</wsse:Nonce>
  <wsu:Created xmlns:wsu=
    "http://schemas.xmlsoap.org/ws/2002/07/utility">
    2002-08-19T00:44:02Z
  </wsu:Created>
</wsse:UsernameToken>

```

### **Extract from [TOKEN]:**

/wsse:UsernameToken/wsse:Password/@{any }

This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the element.

/wsse:UsernameToken/wsse:Nonce

This optional element specifies a cryptographically random nonce. Each message including a <wsse:Nonce> element MUST use a new nonce value in order for web service producers to detect replay attacks.

/wsse:UsernameToken/wsse:Nonce/@EncodingType

his optional attribute URI specifies the encoding type of the nonce (see the definition of <wsse:BinarySecurityToken> for valid values). If this attribute isn't specified, then the default of Base64 encoding is used.

/wsse:UsernameToken/wsdu:Created

The optional <wsu:Created> element specifies a timestamp used to indicate the creation time. It is defined as part of the <wsu:Timestamp> definition.

## 10.7.5.2 Examples

The following example shows a full SOAP-Envelope with a <wsse:Security> element containing a UsernameToken. This security header shows the minimum required security header for PLM Services.

### *UsernameToken without password encryption*

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
      soapenv:mustUnderstand="1">
      <wsse:UsernameToken>
        <wsse:Username>test</wsse:Username>
        <wsse:Password Type=
          "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText">security
        </wsse:Password>
      </wsse:UsernameToken>
      ...
    </wsse:Security>
  </soapenv:Header>
  <soapenv:Body>
    ...
  </soapenv:Body>
</soapenv:Envelope>
```

The next example shows the same security header with an encrypted password within the UsernameToken.

### *UsernameToken with password encryption*

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
      soapenv:mustUnderstand="1">
      <xenc:EncryptedKey>
        <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"></xenc:EncryptionMethod>
        <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
          <wsse:SecurityTokenReference>
            <wsse:KeyIdentifier
              EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"
              ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3">
                MIIBYjCCAQygAwIBAgIQIWF09wjTxZJOxcgGtBqGVTANBqkqkhiG9w0BAQQFADAPMQ0wCwYDVQQDEwRkaW1zMB4XD
                TAzMDUxMjE2NDExN1oXDTM5MTIzMTIzNTk1OVowDzENMAsGA1UEAxMEZGltczBcMA0GCSqGSIb3DQEBAQUAA0sAM
                EgCQQDrmZ7T2MFQNWloGughSRoapkmbvtPAwBXt+21bFzqfXJ1SpliN6CCRczIfISQCCyBZ2j0dA51nZDWDizdNenAgMBA
                AGjRDBCMEAGA1UdAQQ5MDeAEBsIiVE Sxf6DrjkLYXayxmKhETAPMQ0wCwYDVQQDEwRkaW1zZgAhYU73CNPFk7FyAa
                0GoZVMA0GCSqGSIb3DQEBAUAA0EAXSGwjZFOScVLIVTxic1FKmPd8WTg1DrJFDWuxMTx6n0Zxn4N8ZxkAI7TNxJclIG+
                dlNyWZ0in3dOEtF0g5mA==
            </wsse:KeyIdentifier>
          </wsse:SecurityTokenReference>
        </ds:KeyInfo>
      <xenc:CipherData>
        <xenc:CipherValue>
          4eJCKfQxwrkYddfHcO0jZQ8q3pMbTk+Wo3AQSwNboDp7ej8kUSR6F4eM7Ld14lh1gNR7JrDKwE4fq2RPaNg==
        </xenc:CipherValue>
      </xenc:CipherData>
      <xenc:ReferenceList>
        <xenc:DataReference URI="#EncDataId-21721154"></xenc:DataReference>
      </xenc:ReferenceList>
    </xenc:EncryptedKey>
```

```

<wsse:UsernameToken>
  <wsse:Username>test</wsse:Username>
  <wsse:Password Type=
    "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText">
    <xenc:EncryptedData Id="EncDataId-21721154" Type="http://www.w3.org/2001/04/xmlenc#Content">
      <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripleDES-cbc"></xenc:EncryptionMethod>
      <xenc:CipherData>
        <xenc:CipherValue>YwFBdy5F2xFj5Y8wOYjmAB7vW8obeSml</xenc:CipherValue>
      </xenc:CipherData>
    </xenc:EncryptedData>
  </wsse:Password>
</wsse:UsernameToken>
</wsse:Security>
</soapenv:Header>
<soapenv:Body>
  ...
</soapenv:Body>
</soapenv:Envelope>

```

### 10.7.6 Data-Exchange Security

The WS-Security standard enables the sender to encrypt or sign the SOAP message, independently if the information that should be signed or encrypted is in the header or the body of the SOAP message.

The encryption or signing of data sent by the client or provider is optional. All information need by a receiver of an encrypted or signed message is contained in the security header. Only the key that was used for the encryption is not placed in the security header.

The key management between interoperating parties should be organized privately (not defined within this specification).

Any data within the SOAP body may be encrypted or signed. Which data shall be encrypted or signed depends on the concrete application scenario.

#### Example

```

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
      soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next" soapenv:mustUnderstand="1">
      ...
    </wsse:Security>
  </soapenv:Header>
  <soapenv:Body>
    ...
  </soapenv:Body>
</soapenv:Envelope>

```

For more details, refer to “Web Services Security: SOAP Message Security 1.1” [SEC].



### 10.7.6.1 XML-Encryption

The encryption of data secures the data over several intermediaries. This means the data sent from one Web Service to another shall not be readable for intermediate servers, but only for the receiver specified with the attribute 'role'/'actor' (depending on the used SOAP version). For more information about the XML-Encryption standard refer to the XML-Encryption specification at the W3C organization on <http://www.w3.org/Encryption/2001/> [ENC].

Recommendation for the use of Encryption:

- Should be used for the response ((3) + (6)) or for a write request (5)
- Several Security/SOAP-Frameworks cannot define the security settings based on a single method, but only for a complete web service. Optionally a customized handler can be implemented or namespaces can be used for specific encryption.

This is an additional reason why the request from the connection factory to the general connection (2) and the query request (5) should be encrypted, too.

- The increased size of the data to transfer due to encryption of (2) and (5) is rather (less than 2 kB) as that are small messages.
- The used encryption method (algorithm), keyInfo, ... should be negotiated between the sender and receiver.
- Also the type of encryption (e.g., hybrid) should be negotiated between the sender and receiver.

#### **Client - Connection Factory (getConnection)**

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xenc="http://www.w3.org/2001/04/
xmlenc#" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd" soap-
env:mustUnderstand="1">
      <xenc:EncryptedKey Id="EncKeyId-1432189">
        <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"></xenc:EncryptionMethod>
        <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
          <wsse:SecurityTokenReference>
            <ds:X509Data>
              <ds:X509IssuerSerial>
                <ds:X509IssuerName>CN=vn,OU=ou,O=o,C=de</ds:X509IssuerName>
                <ds:X509SerialNumber>1169805422</ds:X509SerialNumber>
              </ds:X509IssuerSerial>
            </ds:X509Data>
          </wsse:SecurityTokenReference>
        </ds:KeyInfo>
      <xenc:CipherData>
        <xenc:CipherValue>X8kT97Aobz+5mWyAsP0xlXZpbu8xtLYtEMqu03e3yUXsLyuo9o1UIUnFAnkyqvAmkrK8uNGB/
aaBuuLXmnLVuxmn5eM2CbfKfL0buKyMHPy4To6/
pPCAaw+jCtLXG8dysH5mMtlNKUYaCmYZ30ZckyHMjezJEDKle0DUqqrwEFH1Y=</xenc:CipherValue>
      </xenc:CipherData>
      <xenc:ReferenceList>
        <xenc:DataReference URI="#EncDataId-17972319"></xenc:DataReference>
      </xenc:ReferenceList>
    </xenc:EncryptedKey>
    <wsse11:SignatureConfirmation xmlns:wsse11="http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
Value="EdbPNg39FyAfJQeZ53/umQb0cXoqiiODTNERamjto22uaHBhSToxIP79vb/klzin7LjPtDr/
UOiHbFDDOPqxZXxRX3+rByDnhlx6TDOHXF4O9xD88jndw2dEsuK7UJUehw0FLU+9/AmXAcDKVMoe-
HmeD+QUC309H3KWUXrZrNt0=" wsu:Id="SigConf-6810664"></wsse11:SignatureConfirmation>
  </wsse:Security>
  <ns1:sessionId xmlns:ns1="http://xml.apache.org/axis/session" soapenv:actor="http://schemas.xmlsoap.org/soap/actor/
next" soapenv:mustUnderstand="0">F883E63F3BD61F22761B3890B641F495</ns1:sessionId>
```

```

</soapenv:Header>
<soapenv:Body>
  <xenc:EncryptedData Id="EncDataId-17972319" Type="http://www.w3.org/2001/04/xmlenc#Content">
    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"></xenc:EncryptionMethod>
    <xenc:CipherData>
      <xenc:CipherValue>neHaThSxwfRVexeHdPewYOLFDVPaHI9/
PTMds1VGuR+9YZzcCoHBtpBWIHe2t2V8CiFjvlUfZNUtN5Sz4emVF44up9+52JMp2qqJ0lc/xs8dYLBn7J7g1TuxFs2WAjbi/
0BWiQUYMHgNeWijSk4CR9prR6MrS2N92gvC9xTb+YZcZIEP3moN5HI+1yy7IcqHEDyZl0ZxA9wjVaOxnBt9WsFPRgZofzbW
QuUdKE87S5gSr4jBhIHnCcgs0lUow6HBlykQ7YLDIsTzINRHYyPVmee/Y08sPXnFQDI8+5m0ckMR/
ieIKkJivmKt2CFaFhEyNlzzV9tMHljW5UriJLN8BE7vUC4yAnr3KEWmXxKcwc/
ulp4u9Y50MUG4J3Y61liuE58AFKz7KbVmNykiUI2Bkv4C1MYAZWgDffc7diMkiU=</xenc:CipherValue>
    </xenc:CipherData>
  </xenc:EncryptedData>
</soapenv:Body>
</soapenv:Envelope>

```

### 10.7.6.2 XML-Signature (XMLDsig)

XML-Signature specification is also a W3C standard and defines an XML writing for digital signatures. It enables the sender to sign any data within an XML document. If embedded in the SOAP document, the XML Signature is called an enveloped signature. The hash code of the signature can be used to prove that the message hasn't been manipulated on his way from the sender to the receiver. So the receiver can check the authenticity of the sender and of the message. For more details about the XML-Signature standard refer to the XML- Signature specification at the W3C organization on <http://www.w3.org/TR/xmlldsig-core/> [SIGNATURE].

#### Client - Connection Factory (getConnection)

```

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd" soapenv:actor="security">
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmlldsig#" Id="Signature-19475367">
        <ds:SignedInfo>
          <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmlldsig#rsa-sha1"/>
          <ds:Reference URI="#id-4545192">
            <ds:Transforms>
              <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
            </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmlldsig#sha1"/>
          <ds:DigestValue>1Jc3W+ucn3epqx3SLu2xWtkDWFc=</ds:DigestValue>
        </ds:Reference>
      </ds:SignedInfo>
      <ds:SignatureValue>gMRwn8FDU0Cwg8f2793fqDdnyLzPe/
kGEMP2Bafm2txVBWau9NIh3m++R2vbCjdc84EMH2zWaH+R3eiMCEHViaO1iHHwhsWpxVoMwxyN6Y6JgaMJQcLSjHoGln1
pWCTs8iMqOpJtRBji4HQxLICtorK9lva78WxD9HTaY6g9mY=</ds:SignatureValue>
      <ds:KeyInfo Id="KeyId-15500127">
        <wsse:SecurityTokenReference xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd" wsu:Id="STRId-8859832">
          <ds:X509Data>
            <ds:X509IssuerSerial>
              <ds:X509IssuerName>CN=vn,OU=ou,O=o,C=de</ds:X509IssuerName>
              <ds:X509SerialNumber>1169805422</ds:X509SerialNumber>
            </ds:X509IssuerSerial>
          </ds:X509Data>
        </wsse:SecurityTokenReference>
      </ds:KeyInfo>
    </ds:Signature>
    <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecu-
rity-utility-1.0.xsd" wsu:Id="UsernameToken-15230484">
      <wsse:Username>test</wsse:Username>
    </wsse:UsernameToken>
  </soapenv:Header>
  <soapenv:Body>
    <test:MethodCall/>
  </soapenv:Body>
</soapenv:Envelope>

```

```

        <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordDigest">spPY5tvoDFJLawz1BOXsOtCTfT4=</wsse:Password>
        <wsse:Nonce>ZcFL2cKCn+dd6ld8CuVegw==</wsse:Nonce>
        <wsu:Created>2007-02-12T12:15:12.556Z</wsu:Created>
        </wsse:UsernameToken>
    </wsse:Security>
</soapenv:Header>
<soapenv:Body xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd" wsu:Id="id-4545192">
    <get_general_connection xmlns="urn:omg.org/plm20/services/parameter"/>
</soapenv:Body>
</soapenv:Envelope>

```

## 10.7.7 Attachments Exchange Security

The WS-Security standard enables the sender to encrypt or sign the SOAP message. The encryption allows encryption on any part of the SOAP message, the header, the body and although the possibly added attachments. PLM Services 2.0 does not describe how to secure an attachment transfer by a separate specified URL but there are some recommendations about this.

- use SSL,
- use basic authentication, or
- check transaction context to be sure requestor works within already secured PLM session.

### 10.7.7.1 SAML (Outlook)

SAML (Secure Assertion Markup Language) is also an OASIS standard. It enables the transport of authentication and authorization information in an SAML assertion. This information as well as the following is only a preview for the further security invention in PLM-Services. The usage of SAML is optional.

WS-Security outlines the basics for using further security standard in a SOAP message. The SAML assertions are also placed in the <wsse:Security> element. This assertion will be generated by an SAML authority that can be implemented by every company.

The process starts with an SAML request that asks for authentication or authorization. The SAML authority generates an SAML assertion with the given information. This requires that the SAML authority uses an existing system to do the authentication or authorization. As a result, the SAML authority generates the mentioned assertion that can contain information about a Subject (authentication, attributes, and authorization). This assertion can be sent to the client, within the security header of the SOAP message. The client can now use the SAML assertion in every request to assert his authentication or authorization over cross-company borders.

SAML offers therefore a possibility for single sign on and a standardized envelope for security related data.

For further details about SAML refer to the specification and documents at OASIS on [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=security](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security) [OASIS]

# A PIM and PSM for Product Lifecycle Management Services support information

## A.1 Mapping Specification (informative)

The mapping of the AIM representation of ISO 10303 AP214 CC21 to the PLM Equivalence Model defined in Section A.2 is given for informative purposes. It is defined in EXPRESS-X. It is informative and not included in this document. The machine readable file is to be found as indicated in Table 6.1.

## A.2 PIM Equivalence Model (informative)

The platform independent equivalence model is defined in EXPRESS. This model is produced by the EXPRESS-X mapping specification as defined in Section A.1 and is equivalent to the CC21 ARM model of ISO10303 AP214. For documentation see the explanation to the corresponding UML elements in Section 8. The model file is informative and not included in this document. The machine readable file is to be found as indicated in Table 6.1.

## A.3 PIM models (normative)

The PIM model for Product Lifecycle Management Services is defined in XMI and provided in extra OMG documents. The Informational Viewpoint and the Computational Viewpoint are provided in two separate XML documents. Both are normative and not included in this document. The machine readable files are to be found as indicated in Table 6.1.

## A.4 PSM models (normative)

The PSM models for Product Lifecycle Management Services are defined in XMI and provided in extra OMG documents. The Informational Viewpoint and the Computational Viewpoint are provided in two separate XML documents. Both are normative and not included in this document. The machine readable files are to be found as indicated in Table 6.1.

## A.5 PSM (normative)

The PSM for Product Lifecycle Management Services is defined in XML Schema and in WSDL and provided in extra OMG documents. The Informational Viewpoint (XML Schema) and the Computational Viewpoint (WSDL) are provided in two separate file systems. Both are normative and not included in this document. The machine readable files are to be found as indicated in Table 6.1.



## Index

### A

Acknowledgements 4  
Activity 54  
Activity\_query 262  
Activity\_authorization\_query 264  
Activity\_element 54  
Activity\_element\_query 265  
Activity\_relationship\_query 266  
Activity\_resolved\_request\_query 267  
Additional Information 3  
AIM  
    Application Interpreted Model 2  
Alias identification 159  
Alias\_identification\_query 268  
Alternative\_solution 52  
Alternative\_solution\_query 269  
AP  
    Application Protocol 2  
Application\_context\_query 270  
Approval\_relationship\_query 271  
ARM  
    Application Resource Model 2  
Assembly\_component\_placement\_query 272  
Associated\_activity\_query 273  
Associated\_approval\_query 274  
Associated\_classification\_query 275  
Associated\_date\_organization\_query 276  
Associated\_date\_time\_query 277  
Associated\_document\_query 278  
Associated\_effectivity\_query 279  
Associated\_file\_query 280  
Associated\_general\_classification\_query 281  
Associated\_item\_property\_query 282  
Associated\_person\_organization\_query 284  
Associated\_process\_property\_query 285  
Associated\_project\_query 287  
Associated\_property\_query 288  
Authentication/Start-Up of session 14  
Authorization 16  
Authorization - Approval 161  
Authorization - Date and time 162  
Authorization - Person and organization 161

### B

Base Model of the Computational Viewpoint 241  
Browsing down product structure data 19  
Browsing of Abstract Product Structures 48  
Browsing of Alternative Solutions within an Abstract Product Structure 50  
Browsing up product structure data 22

### C

CAD files 7

### CC

Conformance Class 2  
CC21  
    Conformance Class 21 2  
Change content of subscription list 45  
Change management 215  
Change notification 41  
Changes to Adopted OMG Specifications 3  
class Message\_by\_context\_query 353  
class Message\_by\_properties\_query 353  
Class\_specification\_association 52  
Class\_structure\_query 288  
Classification - General 128  
Classification - Item and document 129  
Complex\_product\_query 289  
Computational viewpoint 363  
Configuration 52  
Configuration management - Complex product 186  
Configuration management - Component and instance placement 184  
Configuration management - Effectivity 185  
Configuration management - manufacturing configuration 182  
Configuration management - Product class condition and specification 182  
Configuration management - Product identification 184  
Configuration management - specification category and inclusion 183  
Configuration\_query 290  
Conformance 1  
Conformance Class 2  
Connection Factory 370

### D

Data Exchange Tool 2  
Date\_organization\_assignment\_query 291  
DE tool 8  
Definitions 2  
Design\_discipline\_item\_definition\_query 292  
Display content of subscription list and confirm changes 43  
Document and file management 89  
Document properties 90  
Document\_classification\_hierarchy\_query 294  
Document\_classification\_query 293  
Document\_detail\_query 339  
Document\_property\_query 295  
Document\_query 296  
Document\_representation\_query 298  
Document\_selection\_query 339  
Document\_structure\_query 298  
Document\_traversal\_query 340  
Document\_version\_query 299  
Download a single digital file 29  
Download meta data including structures 26  
Download of product data 25

**E**  
ECM  
  Engineering Change Management 2  
ECM participant approval 58  
ECM participant comments 55  
ECM participant detailing and comments 60  
ECM participant proposal for a change 54  
ECO  
  Engineering Change Order 2  
ECR  
  Engineering Change Request 2  
EDI  
  electronic data interchange 2  
Effectivity 54  
Effectivity\_assignment 54  
Effectivity\_assignment\_query 300  
Effectivity\_query 301  
ENGDAT 7  
  Engineering Data Message 2  
ERP  
  Enterprise resource planning 3  
Event\_reference 54  
Event\_reference\_query 302  
Exceptions 364  
Export of product data 7  
EXPRESS-to-XMI mapping 1  
EXPRESS-X mapping 1  
External\_model\_query 304

**F**  
File\_property\_query 304

**G**  
General Connection WSDL Binding 371  
General\_detail\_query 337  
Generic object query 31  
Generic Queries 365  
Geometric\_model\_relationship\_query 305

**H**  
How to Read this Specification 3

**I**  
Import of product data 10  
Informational viewpoint 362  
Informations 366  
issues/problems viii  
Item\_classification\_hierarchy\_query 307  
Item\_classification\_query 307  
Item\_definition\_instance\_relationship\_query 308  
Item\_detail\_query 341  
Item\_instance 52  
Item\_instance\_query 310  
Item\_query 311  
Item\_selection\_query 342  
Item\_traversal\_query 343

Item\_use\_query 313  
Item\_version\_query 313  
Item\_version\_relationship\_query 314

**M**  
Material property 139  
Message Connection WSDL Binding 371  
Message Queries Conformance Point 352  
Model properties 140  
Multi-language support 238

**N**  
Normative References 1

**O**  
Object Management Group, Inc. (OMG) vii  
Object\_by\_uid\_query 315  
Objects\_by\_uids\_query 316  
Odette  
  Organization for Data exchange 3  
OFTP  
  Odette File Transport Protocol 3  
OMG specifications vii  
Organization\_query 317  
Organization\_relationship\_query 318

**P**  
Package Alias\_identification 159  
Package Authorization 161  
Package Change\_and\_work\_management 215  
Package Classification 128  
Package Configuration\_management 182  
Package Document\_and\_file\_management 89  
Package Multi\_language\_support 238  
Package Part\_identification 71  
Package Part\_structure 76  
Package PLM\_base 66  
Package Process\_planning 227  
Package Properties 138  
Package Shape\_definition\_and\_transformation 108  
Parameters 366  
Part identification 71  
Part structure - Assembly 76  
Part structure - Item instance 77  
PDM  
  Product data management 3  
PDTnet conformance point 336  
PDTnet Queries 367  
Person\_in\_organization\_query 319  
Person\_in\_organization\_relationship\_query 321  
Person\_organization\_assignment\_query 322  
Person\_query 322  
Platform Independent Model (PIM) 1  
Platform Specific Model (PSM) 1  
PLM  
  Product Lifecycle Management 3

- PLM base 66
- PLM Services Computational Viewpoint 370
- PLM tool 8
- PLM\_containment\_feature 260
- PLM\_proxy\_container 260
- PLM\_proxy\_feature 260
- PLM\_proxy\_object 259
- PLM\_reference\_feature 260
- PPS
  - Production planning systems 3
- Process planning 227
- Product Class Identification 47
- Product\_class 52
- Product\_class\_query 323
- Product\_detail\_query 344
- Product\_identification\_query 324
- Product\_selection\_query 345
- Product\_structure\_query 325
- Product\_traversal\_query 346
- Project\_assignment\_query 326
- Project\_query 327
- Properties 138
- Property value 139
- Proxy Queries 368
- Proxy\_query 261

**Q**

- Queries Conformance Point 1

**R**

- References 1
- Retrieve Configuration Data within an Abstract Product Structure 51

**S**

- Schema Info 368
- Scope 1
- Search in design space 34
- Shape definition and transformation 108
- Shape properties 140
- Simple property 139
- Simple\_property\_query 328
- Specific Queries 369
- Specification 52
- Specification\_category 52
- Start node identification 17
- STEP
  - Standard for the Exchange of product model data 3
- STEP PDM file
  - A data exchange file 3
- STEP PDM files 7
- Symbols 2

**T**

- Terms and definitions 2
- typographical conventions viii

**U**

- UML Attributes, Associations and Compositions 360
- UML Classes 358
- UML Interfaces 360
- UML Model 355
- UML Package 356
- Upload a single digital file (simple user interaction) 38
- Upload meta data including structures 39
- Upload of product data 37
- Use cases 7
- Utility Queries 369

**V**

- VDA (in German
  - Verband der Automobilindustrie) 3
- Viewing of Change Management Information 53

**W**

- Work\_order\_detail\_query 348, 351
- Work\_order\_is\_controlling\_query 332
- Work\_order\_query 329
- Work\_order\_selection\_query 347, 349
- Work\_request\_activity\_query 332
- Work\_request\_query 333
- Work\_request\_relationship\_query 335
- Work\_request\_scope\_query 336
- Work\_request\_traversal\_query 350
- WSDL 1

**X**

- XPath conformance point 258
- XPath Queries 369



