



OMG Systems Modeling Language™ (SysML®)

Version 2.0 Beta 2 (with change tracking)
(Revision 2024-02)

Part 2: SysML v1 to SysML v2 Transformation

OMG Document Number: None

Date: February 2024

Standard document URL: <https://www.omg.org/spec/SysML/2.0/Transformation/>

Machine Readable File(s): <https://www.omg.org/spec/SysML/20240201/>

Normative:

<https://www.omg.org/spec/SysML/20240201/SysMLv1Tov2.xmi>

Copyright © 2019-2024, 88solutions Corporation
Copyright © 2019-2024, Airbus
Copyright © 2019-2024, Aras Corporation
Copyright © 2019-2024, Association of Universities for Research in Astronomy (AURA)
Copyright © 2019-2024, BigLever Software
Copyright © 2019-2024, Boeing
Copyright © 2022-2024, Budapest University of Technology and Economics
Copyright © 2021-2024, Commissariat à l'énergie atomique et aux énergies alternatives (CEA)
Copyright © 2019-2024, Contact Software GmbH
Copyright © 2019-2024, Dassault Systèmes (No Magic)
Copyright © 2019-2024, DSC Corporation
Copyright © 2020-2024, DEKonsult
Copyright © 2020-2024, Delligatti Associates LLC
Copyright © 2019-2024, The Charles Stark Draper Laboratory, Inc.
Copyright © 2020-2024, ESTACA
Copyright © 2022-2024, Galois, Inc.
Copyright © 2019-2024, GfSE e.V.
Copyright © 2019-2024, George Mason University
Copyright © 2019-2024, IBM
Copyright © 2019-2024, Idaho National Laboratory
Copyright © 2019-2024, INCOSE
Copyright © 2019-2024, Intercax LLC
Copyright © 2019-2024, Jet Propulsion Laboratory (California Institute of Technology)
Copyright © 2019-2024, Kenntnis LLC
Copyright © 2020-2024, Kungliga Tekniska högskolan (KTH)
Copyright © 2019-2024, LightStreet Consulting LLC
Copyright © 2019-2024, Lockheed Martin Corporation
Copyright © 2019-2024, Maplesoft
Copyright © 2021-2024, MID GmbH
Copyright © 2020-2024, MITRE
Copyright © 2019-2024, Model Alchemy Consulting
Copyright © 2019-2024, Model Driven Solutions, Inc.
Copyright © 2019-2024, Model Foundry Pty. Ltd.
Copyright © 2023-2024, Object Management Group, Inc.
Copyright © 2019-2024, On-Line Application Research Corporation (OAC)
Copyright © 2019-2024, oose Innovative Informatik eG
Copyright © 2019-2024, Østfold University College
Copyright © 2019-2024, PTC
Copyright © 2020-2024, Qualtech Systems, Inc.
Copyright © 2019-2024, SAF Consulting
Copyright © 2019-2024, Simula Research Laboratory AS
Copyright © 2019-2024, System Strategy, Inc.
Copyright © 2019-2024, Thematix Partners, LLC
Copyright © 2019-2024, Tom Sawyer
Copyright © 2022-2024, Tucson Embedded Systems, Inc.
Copyright © 2019-2024, Universidad de Cantabria
Copyright © 2019-2024, University of Alabama in Huntsville
Copyright © 2019-2024, University of Detroit Mercy
Copyright © 2019-2024, University of Kaiserslauten
Copyright © 2020-2024, Willert Software Tools GmbH (SodiusWillert)

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, 9C Medway Road, PMB 274, Milford, MA 01757, U.S.A.

TRADEMARKS

CORBA®, CORBA logos®, FIBO®, Financial Industry Business Ontology®, Financial Instrument Global Identifier®, IIOP®, IMM®, Model Driven Architecture®, MDA®, Object Management Group®, OMG®, OMG Logo®, SoaML®, SOAML®, SysML®, UAF®, Unified Modeling Language™, UML®, UML Cube Logo®, VSIPL®, and XMI® are registered trademarks of the Object Management Group, Inc.

For a complete list of trademarks, see: https://www.omg.org/legal/tm_list.htm. 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 <https://www.omg.org>, under Documents, Report a Bug/Issue.

Table of Contents

0 Preface.....	23
1 Scope.....	1
2 Conformance.....	3
3 Normative References.....	5
4 Terms and Definitions.....	7
5 Symbols	9
6 Introduction.....	11
6.1 Mapping Approach	11
6.2 Acknowledgements.....	11
7 Mappings.....	13
7.1 Overview	13
7.2 Foundations	13
7.2.1 Overview	13
7.2.2 Foundational class specifications.....	14
7.2.2.1 UniqueMapping.....	14
7.2.2.2 Factory.....	14
7.2.2.3 Mapping	14
7.2.2.4 MainMapping.....	15
7.2.2.5 Initializer	16
7.3 Mapping Helper and Library.....	16
7.3.1 Helper.....	16
7.3.2 SysML v1 Library	22
7.4 Initializers.....	25
7.4.1 Overview	25
7.4.2 Mapping Specifications.....	25
7.4.2.1 KerML Initializers.....	25
7.4.2.1.1 AnnotatingElement_Init	25
7.4.2.1.2 Annotation_Init	26
7.4.2.1.3 Association_Init.....	26
7.4.2.1.4 Behavior_Init.....	26
7.4.2.1.5 Classifier_Init	27
7.4.2.1.6 Comment_Init.....	27
7.4.2.1.7 Conjugation_Init.....	27
7.4.2.1.8 Connector_Init.....	28
7.4.2.1.9 Documentation_Init.....	28
7.4.2.1.10 Element_Init	28
7.4.2.1.11 EndFeatureMembership_Init	29
7.4.2.1.12 Expression_Init	29
7.4.2.1.13 Feature_Init	30
7.4.2.1.14 FeatureChainExpression_Init	31
7.4.2.1.15 FeatureChaining_Init	31
7.4.2.1.16 FeatureMembership_Init	31
7.4.2.1.17 FeatureReferenceExpression_Init	32
7.4.2.1.18 FeatureTyping_Init	32
7.4.2.1.19 FeatureValue_Init	32
7.4.2.1.20 Function_Init	33
7.4.2.1.21 Import_Init	33
7.4.2.1.22 Interaction_Init	34
7.4.2.1.23 InvocationExpression_Init	34
7.4.2.1.24 ItemFlow_Init	34
7.4.2.1.25 Membership_Init	34

7.4.2.1.26 MembershipImport_Init	35
7.4.2.1.27 Namespace_Init	35
7.4.2.1.28 NamespaceImport_Init	36
7.4.2.1.29 OperatorExpression_Init	36
7.4.2.1.30 OwningMembership_Init	36
7.4.2.1.31 Package_Init	37
7.4.2.1.32 ParameterMembership_Init	37
7.4.2.1.33 Predicate_Init.....	37
7.4.2.1.34 Redefinition_Init	38
7.4.2.1.35 ReferenceSubsetting_Init	38
7.4.2.1.36 Relationship_Init	38
7.4.2.1.37 ReturnParameterMembership_Init	39
7.4.2.1.38 Specialization_Init.....	39
7.4.2.1.39 Step_Init	40
7.4.2.1.40 Subclassification_Init	40
7.4.2.1.41 Subsetting_Init	40
7.4.2.1.42 Succession_Init.....	41
7.4.2.1.43 SuccessionItemFlow_Init	41
7.4.2.1.44 TextualRepresentation_Init	41
7.4.2.1.45 Type_Init	41
7.4.2.1.46 TypeFeaturing_Init.....	42
7.4.2.2 System Initializers	42
7.4.2.2.1 ActionUsage_Init.....	42
7.4.2.2.2 ActorMembership_Init	43
7.4.2.2.3 AssignmentActionUsage_Init	43
7.4.2.2.4 ConjugatedPortDefinition_Init	43
7.4.2.2.5 ConjugatedPortTyping_Init.....	43
7.4.2.2.6 ConnectionUsage_Init	44
7.4.2.2.7 ConstraintDefinition_Init	44
7.4.2.2.8 ConstraintUsage_Init.....	44
7.4.2.2.9 Definition_Init.....	45
7.4.2.2.10 EventOccurrenceUsage_Init	45
7.4.2.2.11 FlowConnectionUsage_Init	45
7.4.2.2.12 ItemDefinition_Init.....	45
7.4.2.2.13 ItemFeature_Init	46
7.4.2.2.14 MetadataUsage_Init	46
7.4.2.2.15 ObjectiveMembership_Init	46
7.4.2.2.16 OccurrenceDefinition_Init	47
7.4.2.2.17 OccurrenceUsage_Init	47
7.4.2.2.18 PartUsage_Init	47
7.4.2.2.19 PortConjugation_Init	48
7.4.2.2.20 PortDefinition_Init	48
7.4.2.2.21 ReferenceUsage_Init	48
7.4.2.2.22 RequirementUsage_Init	48
7.4.2.2.23 StateUsage_Init	49
7.4.2.2.24 SubjectMembership_Init	49
7.4.2.2.25 Usage_Init	49
7.5 Factories	50
7.5.1 Overview	50
7.5.2 Mapping Specifications.....	50
7.5.2.1 LiteralString_Factory	50
7.5.2.2 StringParameterFeature_Factory	50
7.5.2.3 StringParameterFeatureValue_Factory	51
7.5.2.4 StringParameterMembership_Factory	51

7.5.2.5 SubjectMembership_Factory	52
7.5.2.6 AssignmentActionUsage_Factory.....	52
7.5.2.7 AssignmentActionUsageFeatureMembership2_Factory	52
7.5.2.8 AssignmentActionUsageFeatureMembership3_Factory	53
7.5.2.9 AssignmentActionUsageOwningMembership_Factory.....	53
7.5.2.10 AssignmentActionUsageParameterMembership_Factory	54
7.5.2.11 AssignmentActionUsageReferenceUsageIn1_Factory	54
7.5.2.12 AssignmentActionUsageTargetReferenceUsageIn2_Factory.....	54
7.5.2.13 AssignmentActionUsageTargetReferenceUsageIn3_Factory.....	55
7.5.2.14 DirectedReferenceUsage_Factory.....	55
7.5.2.15 DirectedReferenceUsageParameterMembership_Factory	56
7.5.2.16 EmptyObjectiveMembership_Factory	56
7.5.2.17 EmptyRequirementUsage_Factory	56
7.5.2.18 EmptySubject_Factory	57
7.5.2.19 EmptySubjectMembership_Factory.....	57
7.5.2.20 FeatureTyping_Factory	58
7.5.2.21 FlowConnectionUsage_Factory	58
7.5.2.22 FlowConnectionUsageFeatureMembership_Factory	59
7.5.2.23 FlowEndParameterMembership_Factory	59
7.5.2.24 FlowItem_Factory	60
7.5.2.25 FlowItemFeatureMembership_Factory	61
7.5.2.26 InformationFlowEventOccurrenceUsage_Factory	61
7.5.2.27 InformationFlowReferenceSubsetting_Factory	61
7.5.2.28 LiteralBoolean_Factory.....	62
7.5.2.29 LiteralNull_Factory	62
7.5.2.30 LiteralRational_Factory	63
7.5.2.31 ObjectFlowItemFlowEndRedefinition_Factory	63
7.5.2.32 ReferenceSubsetting_Factory.....	64
7.5.2.33 ReturnParameterFeature_Factory.....	64
7.5.2.34 ReturnParameterFeatureMembership_Factory	64
7.5.2.35 Subsetting_Factory	65
7.6 Generic Mappings	65
7.6.1 Overview	65
7.6.2 Common Mappings	66
7.6.2.1 CommonFeatureReferenceExpression_Mapping.....	66
7.6.2.2 CommonMembership_Mapping	66
7.6.2.3 CommonParameterReferenceUsageInMembership_Mapping.....	67
7.6.2.4 CommonParameterReferenceUsageIn_Mapping	68
7.6.2.5 CommonParameterReferenceUsageInFeatureTyping_Mapping	68
7.6.2.6 CommonParameterReferenceUsageInUntyped_Mapping	69
7.6.2.7 CommonReturnParameterFeature_Mapping	70
7.6.2.8 CommonReturnParameterFeatureTyping_Mapping	70
7.6.2.9 CommonReturnParameterFeatureUntyped_Mapping	71
7.6.2.10 CommonReturnParameterFeatureMembership_Mapping	72
7.6.2.11 CommonReturnParameterReferenceUsageMembership_Mapping	72
7.6.2.12 CommonReturnParameterReferenceUsage_Mapping	73
7.6.2.13 CommonReturnParameterReferenceUsageFeatureTyping_Mapping	74
7.6.2.14 CommonReturnParameterReferenceUsageUntyped_Mapping	75
7.6.2.15 CommonReferenceUsageIn_Mapping	75
7.6.2.16 CommonReferenceUsageInFeatureMembership_Mapping	76
7.6.2.17 CommonReferenceUsageInFeatureTyping_Mapping	77
7.6.2.18 CommonReferenceUsageInUntyped_Mapping	77
7.6.3 Generic Mappings To KerML	78
7.6.3.1 GenericToAnnotatingElement_Mapping	78

7.6.3.2 GenericToAnnotation_Mapping	79
7.6.3.3 GenericToAssociation_Mapping	79
7.6.3.4 GenericToBehavior_Mapping	80
7.6.3.5 GenericToClassifier_Mapping	80
7.6.3.6 GenericToComment_Mapping	81
7.6.3.7 GenericToConjugation_Mapping	81
7.6.3.8 GenericToConnector_Mapping	82
7.6.3.9 GenericToDocumentation_Mapping	83
7.6.3.10 GenericToElement_Mapping	83
7.6.3.11 GenericToEndFeatureMembership_Mapping	84
7.6.3.12 GenericToExpression_Mapping	85
7.6.3.13 GenericToFeature_Mapping	85
7.6.3.14 GenericToFeatureChainExpression_Mapping	86
7.6.3.15 GenericToFeatureChaining_Mapping	87
7.6.3.16 GenericToFeatureMembership_Mapping	87
7.6.3.17 GenericToFeatureReferenceExpression_Mapping	88
7.6.3.18 GenericToFeatureTyping_Mapping	88
7.6.3.19 GenericToFeatureValue_Mapping	89
7.6.3.20 GenericToFunction_Mapping	90
7.6.3.21 GenericToImport_Mapping	90
7.6.3.22 GenericToInvocationExpression_Mapping	91
7.6.3.23 GenericToInteraction_Mapping	92
7.6.3.24 GenericToItemFlow_Mapping	92
7.6.3.25 GenericToMembership_Mapping	93
7.6.3.26 GenericToMembershipImport_Mapping	93
7.6.3.27 GenericToNamespace_Mapping	94
7.6.3.28 GenericToNamespaceImport_Mapping	94
7.6.3.29 GenericToOperatorExpression_Mapping	95
7.6.3.30 GenericToOwningMembership_Mapping	96
7.6.3.31 GenericToPackage_Mapping	96
7.6.3.32 GenericToParameterMembership_Mapping	97
7.6.3.33 GenericToPredicate_Mapping	98
7.6.3.34 GenericToRedefinition_Mapping	98
7.6.3.35 GenericToReferenceSubsetting_Mapping	99
7.6.3.36 GenericToRelationship_Mapping	99
7.6.3.37 GenericToReturnParameterMembership_Mapping	100
7.6.3.38 GenericToSpecialization_Mapping	101
7.6.3.39 GenericToStep_Mapping	102
7.6.3.40 GenericToSubclassification_Mapping	102
7.6.3.41 GenericToSubsetting_Mapping	103
7.6.3.42 GenericToSuccession_Mapping	104
7.6.3.43 GenericToSuccessionItemFlow_Mapping	104
7.6.3.44 GenericToTextualRepresentation_Mapping	104
7.6.3.45 GenericToType_Mapping	105
7.6.3.46 GenericToTypeFeaturing_Mapping	106
7.6.4 Generic Mappings to Systems	107
7.6.4.1 GenericToActionUsage_Mapping	107
7.6.4.2 GenericToActorMembership_Mapping	107
7.6.4.3 GenericToAssignmentActionUsage_Mapping	108
7.6.4.4 GenericToConnectionUsage_Mapping	108
7.6.4.5 GenericToConjugatedPortDefinition_Mapping	109
7.6.4.6 GenericToConjugatedPortTyping_Mapping	109
7.6.4.7 GenericToConstraintDefinition_Mapping	110
7.6.4.8 GenericToConstraintUsage_Mapping	110

7.6.4.9 GenericToDefinition_Mapping	111
7.6.4.10 GenericToEventOccurrenceUsage_Mapping	111
7.6.4.11 GenericToItemDefinition_Mapping	112
7.6.4.12 GenericToItemUsage	112
7.6.4.13 GenericToMetadataUsage_Mapping	113
7.6.4.14 GenericToObjectMembership_Mapping	113
7.6.4.15 GenericToOccurrenceDefinition_Mapping	113
7.6.4.16 GenericToOccurrenceUsage_Mapping	114
7.6.4.17 GenericToPartUsage_Mapping	115
7.6.4.18 GenericToPortConjugation_Mapping	115
7.6.4.19 GenericToPortDefinition_Mapping	116
7.6.4.20 GenericToReferenceUsage_Mapping	116
7.6.4.21 GenericToRequirementUsage_Mapping	117
7.6.4.22 GenericToStateUsage_Mapping	117
7.6.4.23 GenericToSubjectMembership_Mapping	118
7.6.4.24 GenericToTransitionUsage_Mapping	118
7.6.4.25 GenericToUsage_Mapping	118
7.7 Mappings from UML4SysML metaclasses	119
7.7.1 Overview	119
7.7.2 Actions	119
7.7.2.1 Overview	119
7.7.2.2 UML4SysML::Actions elements not mapped	121
7.7.2.3 Mapping Specifications	122
7.7.2.3.1 Accept Event Actions	122
7.7.2.3.1.1 AcceptCallAction_Mapping	122
7.7.2.3.1.2 AcceptEventAction_Mapping	123
7.7.2.3.1.3 AEAChangeExpressionMembership_Mapping	124
7.7.2.3.1.4 AEAChangeParameter_Mapping	124
7.7.2.3.1.5 AEAChangeParameterValue_Mapping	125
7.7.2.3.1.6 AEAChangeParameterTrigger_Mapping	126
7.7.2.3.1.7 AEAChangeParameterTriggerExpression_Mapping	126
7.7.2.3.1.8 AEAChangeParameterResultExpressionMembership_Mapping	127
7.7.2.3.1.9 AEAChangeParameterFeatureChainExpression_Mapping	128
7.7.2.3.1.10 AEAChangeParameterFeature_Mapping	128
7.7.2.3.1.11 AEAChangeParameterExpressionFeatureValue_Mapping	129
7.7.2.3.1.12 AEAChangeParameterFeatureReferenceExpression_Mapping	130
7.7.2.3.1.13 AEAChangeParameterMembership_Mapping	130
7.7.2.3.1.14 AEAChangeParameterParameterMembership_Mapping	131
7.7.2.3.1.15 AEAReceiverParameter_Mapping	131
7.7.2.3.1.16 AEAReceiverParameterMembership_Mapping	132
7.7.2.3.1.17 AEAReceiverFeatureValue_Mapping	133
7.7.2.3.1.18 AEASignalParameter_Mapping	134
7.7.2.3.1.19 AEASignalParameterFeatureTyping_Mapping	134
7.7.2.3.1.20 AEAParameterMembership_Mapping	135
7.7.2.3.1.21 AEAReceiverFeatureReferenceExpression_Mapping	136
7.7.2.3.1.22 AEAReceiverFeatureReferenceExpressionMembership_Mapping	136
7.7.2.3.1.23 ReplyAction_Mapping	137
7.7.2.3.1.24 UnmarshallAction_Mapping	137
7.7.2.3.2 Actions	138
7.7.2.3.2.1 CommonAction_Mapping	138
7.7.2.3.2.2 OpaqueAction_Mapping	139
7.7.2.3.2.3 OABody_Mapping	140
7.7.2.3.2.4 OABodyMembership_Mapping	140
7.7.2.3.2.5 Pin_Mapping	141

7.7.2.3.2.6 ValuePin_Mapping	142
7.7.2.3.2.7 ValuePinFeatureValue_Mapping	143
7.7.2.3.2.8 ValuePinUntyped_Mapping	144
7.7.2.3.3 Invocation Actions	145
7.7.2.3.3.1 BroadcastSignalAction_Mapping	145
7.7.2.3.3.2 CallBehaviorAction_Mapping	145
7.7.2.3.3.3 CBAFeatureTyping_Mapping	146
7.7.2.3.3.4 CallOperationAction_Mapping	146
7.7.2.3.3.5 COAOutputPinFeature_Mapping	147
7.7.2.3.3.6 COAOutputPinFeatureChainExpression_Mapping	148
7.7.2.3.3.7 COAOutputPinFeatureChainExpressionMembership_Mapping	149
7.7.2.3.3.8 COAOutputPinFeatureFeature_Mapping	149
7.7.2.3.3.9 COAOutputPinFeatureFeatureMembership_Mapping	150
7.7.2.3.3.10 COAOutputPinFeatureFeatureValue_Mapping	150
7.7.2.3.3.11 COAOutputPinFeatureMembership_Mapping	151
7.7.2.3.3.12 COAOutputPinFeatureReferenceExpression_Mapping	152
7.7.2.3.3.13 COAOutputPinFeatureReferenceExpressionMembership_Mapping	152
7.7.2.3.3.14 COAOutputPinParameterMembership_Mapping	153
7.7.2.3.3.15 COAOutputPinReferenceUsage_Mapping	153
7.7.2.3.3.16 COAOutputPinReferenceUsageFeatureValue_Mapping	154
7.7.2.3.3.17 COAPerformAction_Mapping	155
7.7.2.3.3.18 COAPerformActionFeatureMembership_Mapping	155
7.7.2.3.3.19 COAPerformActionReferenceSubsetting_Mapping	156
7.7.2.3.3.20 COAPerformActionFeature_Mapping	157
7.7.2.3.3.21 COAPerformActionFeatureChainingOperation_Mapping	157
7.7.2.3.3.22 COAPerformActionFeatureChainingTarget_Mapping	158
7.7.2.3.3.23 SendObjectAction_Mapping	158
7.7.2.3.3.24 SendSignalAction_Mapping	159
7.7.2.3.3.25 SSAFeatureMembership_Mapping	160
7.7.2.3.3.26 SSAParameterMembership_Mapping	160
7.7.2.3.3.27 SSAReferenceUsage_Mapping	161
7.7.2.3.3.28 SSAItemParameterMembership_Mapping	162
7.7.2.3.3.29 SSAItemReferenceUsage_Mapping	162
7.7.2.3.3.30 SSAItemReferenceUsageFeatureValue_Mapping	163
7.7.2.3.3.31 SSAItemReferenceUsageFeatureTyping_Mapping	164
7.7.2.3.3.32 SSAItemReferenceUsageInvocationExpression_Mapping	164
7.7.2.3.3.33 SSATargetParameterMembership_Mapping	165
7.7.2.3.3.34 SSATargetReferenceUsage_Mapping	166
7.7.2.3.3.35 SSATargetReferenceUsageFeatureValue_Mapping	166
7.7.2.3.3.36 SSATargetReferenceUsageFeatureValueMembership_Mapping	167
7.7.2.3.3.37 SSATargetReferenceUsageFeatureValueExpression_Mapping	167
7.7.2.3.3.38 SSASendActionUsage_Mapping	168
7.7.2.3.3.39 StartClassifierBehaviorAction_Mapping	169
7.7.2.3.3.40 StartObjectBehaviorAction_Mapping	169
7.7.2.3.4 Link Actions	170
7.7.2.3.4.1 ClearAssociationAction_Mapping	170
7.7.2.3.4.2 CreateLinkAction_Mapping	170
7.7.2.3.4.3 CreateLinkObjectAction_Mapping	171
7.7.2.3.4.4 DestroyLinkAction_Mapping	171
7.7.2.3.4.5 ReadLinkAction_Mapping	172
7.7.2.3.4.6 ReadLinkObjectEndAction_Mapping	173
7.7.2.3.4.7 ReadLinkObjectEndQualifierAction_Mapping	173
7.7.2.3.5 Object Actions	174
7.7.2.3.5.1 CreateObjectAction_Mapping	174

7.7.2.3.5.2 COAInvocationExpressionFeatureTyping_Mapping	174
7.7.2.3.5.3 COAInvocationExpression_Mapping	175
7.7.2.3.5.4 COAPin_Mapping	176
7.7.2.3.5.5 COAPinFeatureValue_Mapping	176
7.7.2.3.5.6 DestroyObjectAction_Mapping	177
7.7.2.3.5.7 DOADestroyActionUsage_Mapping	178
7.7.2.3.5.8 DOADestroyActionUsageFeatureMembership_Mapping	179
7.7.2.3.5.9 DOADestroyActionUsageFeatureReferenceExpression_Mapping	179
7.7.2.3.5.10 DOADestroyActionUsageMembership_Mapping	180
7.7.2.3.5.11 DOADestroyActionUsageFeatureTyping_Mapping	180
7.7.2.3.5.12 DOADestroyActionUsageFeatureValue_Mapping	181
7.7.2.3.5.13 DOADestroyActionUsageReferenceUsage_Mapping	182
7.7.2.3.5.14 DOADestroyFeatureMembership_Mapping	182
7.7.2.3.5.15 ReadIsClassifiedObjectAction_Mapping	183
7.7.2.3.5.16 RICOAFeatureValue_Mapping	184
7.7.2.3.5.17 RICOAFeatureValueOperatorExpression_Mapping	184
7.7.2.3.5.18 RICOAFeatureValueOperatorExpressionFeature_Mapping	185
7.7.2.3.5.19 RICOAFeatureValueOperatorExpressionFeatureValue_Mapping	186
7.7.2.3.5.20 RICOAFeatureValueOperatorFeatureReferenceExpression_Mapping	186
7.7.2.3.5.21 RICOAFeatureValueOperatorMembership_Mapping	187
7.7.2.3.5.22 RICOAFeatureValueOperatorParameterMembership_Mapping	187
7.7.2.3.5.23 RICOAOutputPin_Mapping	188
7.7.2.3.5.24 ReadExtentAction_Mapping	189
7.7.2.3.5.25 REAFeatureValue_Mapping	189
7.7.2.3.5.26 REAFeatureValueOperatorExpression_Mapping	190
7.7.2.3.5.27 REAFeatureValueOperatorExpressionFeature_Mapping	191
7.7.2.3.5.28 REAFeatureValueOperatorExpressionFeatureTyping_Mapping	191
7.7.2.3.5.29 REAFeatureValueOperatorExpressionMembership_Mapping	192
7.7.2.3.5.30 REAOutputPin_Mapping	193
7.7.2.3.5.31 ReadSelfAction_Mapping	193
7.7.2.3.5.32 RSAFeatureValue_Mapping	194
7.7.2.3.5.33 RSAFeatureValueFeatureReferenceExpression_Mapping	195
7.7.2.3.5.34 RSAFeatureValueMembership_Mapping	195
7.7.2.3.5.35 RSAOutputPin_Mapping	196
7.7.2.3.5.36 ReclassifyObjectAction_Mapping	197
7.7.2.3.5.37 TestIdentityAction_Mapping	197
7.7.2.3.5.38 TIAOperatorExpression_Mapping	198
7.7.2.3.5.39 TIAResultExpressionMembership_Mapping	199
7.7.2.3.5.40 ValueSpecificationAction_Mapping	199
7.7.2.3.5.41 VSAOutputPin_Mapping	201
7.7.2.3.5.42 VSAOutputPinFeatureValue_Mapping	201
7.7.2.3.6 Other Actions	202
7.7.2.3.6.1 RaiseExceptionAction_Mapping	202
7.7.2.3.6.2 ReduceAction_Mapping	202
7.7.2.3.7 Structural Feature Actions	203
7.7.2.3.7.1 AddStructuralFeatureValueAction_Mapping	203
7.7.2.3.7.2 ASFVAFeatureTyping_Mapping	204
7.7.2.3.7.3 ASFVAObjectFeatureMembership_Mapping	205
7.7.2.3.7.4 ASFVAObjectReferenceUsage_Mapping	205
7.7.2.3.7.5 ASFVAObjectReferenceUsageFeatureTyping_Mapping	206
7.7.2.3.7.6 ASFVAObjectReferenceUsageRedefinition_Mapping	207
7.7.2.3.7.7 ASFVATargetFeatureChainExpression_Mapping	207
7.7.2.3.7.8 ASFVATargetFeatureMembership_Mapping	208
7.7.2.3.7.9 ASFVATargetFeatureValue_Mapping	209

7.7.2.3.7.10 ASFVATargetParameterExpressionFeature_Mapping	209
7.7.2.3.7.11 ASFVATargetParameterExpressionFeatureMembership_Mapping	210
7.7.2.3.7.12 ASFVATargetParameterExpressionMembership_Mapping	210
7.7.2.3.7.13 ASFVATargetParameterFeature_Mapping	211
7.7.2.3.7.14 ASFVATargetParameterFeatureExpressionMembership_Mapping	212
7.7.2.3.7.15 ASFVATargetParameterFeatureReferenceExpression_Mapping	212
7.7.2.3.7.16 ASFVATargetParameterFeatureValue_Mapping	213
7.7.2.3.7.17 ASFVATargetParameterMembership_Mapping	214
7.7.2.3.7.18 ASFVATargetReferenceUsage_Mapping	214
7.7.2.3.7.19 ASFVATargetReferenceUsageRedefinition_Mapping	215
7.7.2.3.7.20 ClearStructuralFeatureAction_Mapping	216
7.7.2.3.7.21 ReadStructuralFeatureAction_Mapping	216
7.7.2.3.7.22 RSFAResourceUsage_Mapping	217
7.7.2.3.7.23 RSFAResourceUsageExpressionFeature_Mapping	218
7.7.2.3.7.24 RSFAResourceUsageExpressionFeatureMembership_Mapping	218
7.7.2.3.7.25 RSFAResourceUsageExpressionFeatureReferenceExpression_Mapping	219
7.7.2.3.7.26 RSFAResourceUsageExpressionFeatureValue_Mapping	219
7.7.2.3.7.27 RSFAResourceUsageFeatureChainExpression_Mapping	220
7.7.2.3.7.28 RSFAResourceUsageFeatureChainExpressionFeature_Mapping	221
7.7.2.3.7.29 RSFAResourceUsageFeatureChainExpressionMembership_Mapping	221
7.7.2.3.7.30 RSFAResourceUsageFeatureMembership_Mapping	222
7.7.2.3.7.31 RSFAResourceUsageFeatureValue_Mapping	222
7.7.2.3.7.32 RSFAResourceUsageMembership_Mapping	223
7.7.2.3.7.33 RSFAResourceUsageParameterMembership_Mapping	224
7.7.2.3.7.34 RemoveStructuralFeatureValueAction_Mapping	224
7.7.2.3.8 Structured Actions	225
7.7.2.3.8.1 LoopNode_Mapping	225
7.7.2.3.8.2 SequenceNode_Mapping	225
7.7.2.3.8.3 StructuredActivityNode_Mapping	226
7.7.2.3.9 Variable Actions	227
7.7.2.3.9.1 AddVariableValueAction_Mapping	227
7.7.2.3.9.2 AVVAFeatureTyping_Mapping	228
7.7.2.3.9.3 AVVAFeatureValue_Mapping	228
7.7.2.3.9.4 AVVAIsReplaceAll_Mapping	229
7.7.2.3.9.5 AVVAIsReplaceAllFeatureMembership_Mapping	230
7.7.2.3.9.6 AVVAIsReplaceAllRedefinition_Mapping	231
7.7.2.3.9.7 AVVAIsReplaceAllValue_Mapping	231
7.7.2.3.9.8 AVVAValueExpressionMembership_Mapping	232
7.7.2.3.9.9 AVVAValueFeatureReferenceExpression_Mapping	233
7.7.2.3.9.10 AVVAVariable_Mapping	233
7.7.2.3.9.11 AVVAVariableFeatureMembership_Mapping	234
7.7.2.3.9.12 AVVAVariableRedefinition_Mapping	235
7.7.2.3.9.13 ClearVariableAction_Mapping	235
7.7.2.3.9.14 CVAFeatureMembership_Mapping	236
7.7.2.3.9.15 CVAResourceUsage_Mapping	237
7.7.2.3.9.16 CVAResourceUsageFeatureValue_Mapping	237
7.7.2.3.9.17 ReadVariableAction_Mapping	238
7.7.2.3.9.18 RVAFeatureMembership_Mapping	239
7.7.2.3.9.19 RVAReferenceUsage_Mapping	239
7.7.2.3.9.20 RVAReferenceUsageFeatureReferenceExpression_Mapping	240
7.7.2.3.9.21 RVAReferenceUsageFeatureTyping_Mapping	241
7.7.2.3.9.22 RVAReferenceUsageFeatureValue_Mapping	241
7.7.2.3.9.23 RVAReferenceUsageExpressionMembership_Mapping	242
7.7.2.3.9.24 RemoveVariableValueAction_Mapping	242

7.7.2.3.9.25 RVVAFEATURETyping_Mapping.....	243
7.7.2.3.9.26 RVVAVARIABLE_Mapping.....	244
7.7.2.3.9.27 RVVAVARIABLEExpressionMembership_Mapping.....	245
7.7.2.3.9.28 RVVAVARIABLEFeatureMembership_Mapping	245
7.7.2.3.9.29 RVVAVARIABLEFeatureReferenceExpression_Mapping.....	246
7.7.2.3.9.30 RVVAVARIABLEFeatureValue_Mapping	247
7.7.2.3.9.31 RVVAVARIABLERedefinition_Mapping.....	247
7.7.3 Activities	248
7.7.3.1 Overview	248
7.7.3.2 UML4SysML::Activities elements not mapped	249
7.7.3.3 Mapping Specifications.....	249
7.7.3.3.1 ActivityAsDefinition_Mapping.....	249
7.7.3.3.2 ActivityEdgeInitialNodeFeatureMembership_Mapping.....	250
7.7.3.3.3 ActivityEdgeMetadata_Mapping	251
7.7.3.3.4 ActivityEdgeMetadataFeatureMembership_Mapping	252
7.7.3.3.5 ActivityEdgeMetadataFeatureTyping_Mapping	252
7.7.3.3.6 ActivityEdgeMetadataFeatureValue_Mapping	253
7.7.3.3.7 ActivityEdgeMetadataOwningMembership_Mapping	253
7.7.3.3.8 ActivityEdgeMetadataRedefinition_Mapping	254
7.7.3.3.9 ActivityEdgeMetadataReferenceUsage_Mapping	255
7.7.3.3.10 ActivityEdgeSourceEndFeature_Mapping	255
7.7.3.3.11 ActivityEdgeSourceInitialNode_Mapping	256
7.7.3.3.12 ActivityEdgeSourceEndFeatureMembership_Mapping	257
7.7.3.3.13 ActivityEdgeSourceInitialNodeSubsetting_Mapping	257
7.7.3.3.14 ActivityEdgeSourceEndSubsetting_Mapping	258
7.7.3.3.15 ActivityEdgeTransitionUsageSourceMembership_Mapping	259
7.7.3.3.16 CentralBufferNode_Mapping.....	260
7.7.3.3.17 CommonActivityEdgeSuccessionAsUsage_Mapping	260
7.7.3.3.18 CommonVariable_Mapping.....	261
7.7.3.3.19 ControlFlowTransitionUsage_Mapping.....	262
7.7.3.3.20 ControlFlowFinalNodeFeatureMembership_Mapping	263
7.7.3.3.21 ControlFlowTargetFinalNodeSubsetting_Mapping	264
7.7.3.3.22 ControlFlowSuccessionAsUsage_Mapping	265
7.7.3.3.23 ControlFlowTargetFinalNode_Mapping	266
7.7.3.3.24 ControlFlowTargetEndFeature_Mapping	267
7.7.3.3.25 ControlFlowTargetFeatureMembership_Mapping	268
7.7.3.3.26 ControlFlowTargetEndSubsetting_Mapping	269
7.7.3.3.27 ControlFlowTransitionUsageFeatureMembership_Mapping	269
7.7.3.3.28 DataStoreNode_Mapping	270
7.7.3.3.29 DecisionNode_Mapping.....	270
7.7.3.3.30 FlowFinalNodeMembership_Mapping	271
7.7.3.3.31 ForkNode_Mapping	272
7.7.3.3.32 InitialNodeMembership_Mapping	273
7.7.3.3.33 JoinNode_Mapping	274
7.7.3.3.34 MergeNode_Mapping	274
7.7.3.3.35 ObjectFlow_Mapping.....	275
7.7.3.3.36 ObjectFlowFeatureMembership_Mapping	276
7.7.3.3.37 ObjectFlowGuardFeatureMembership_Mapping	277
7.7.3.3.38 ObjectFlowGuard_Mapping.....	278
7.7.3.3.39 ObjectFlowGuardSuccessionTargetEndFeature_Mapping	279
7.7.3.3.40 ObjectFlowGuardSuccessionTargetEndFeatureMembership_Mapping	280
7.7.3.3.41 ObjectFlowGuardSuccessionTargetEndSubsetting_Mapping	280
7.7.3.3.42 ObjectFlowItemFeature_Mapping	281
7.7.3.3.43 ObjectFlowItemFeatureMembership_Mapping	282

7.7.3.3.44 ObjectFlowItemFeatureTyping_Mapping	282
7.7.3.3.45 ObjectFlowItemFeatureUntyped_Mapping	283
7.7.3.3.46 ObjectFlowEndFeatureMembership_Mapping	283
7.7.3.3.47 ObjectFlowItemFlowEnd_Mapping	284
7.7.3.3.48 ObjectFlowItemFlowEndReferenceUsage_Mapping	285
7.7.3.3.49 ObjectFlowItemFlowEndFeatureMembership_Mapping	286
7.7.3.3.50 ObjectFlowItemFlowEndRedefinition_Mapping	286
7.7.3.3.51 ObjectFlowItemFlowEndSubsetting_Mapping	287
7.7.3.3.52 ObjectFlowTransitionUsageFeatureMembership_Mapping	288
7.7.3.3.53 VariableAttribute_Mapping	289
7.7.3.3.54 VariableFeatureTyping_Mapping	289
7.7.3.3.55 VariableItem_Mapping	290
7.7.3.3.56 VariableMembership_Mapping	291
7.7.4 Classification	291
7.7.4.1 Overview	291
7.7.4.2 Mapping Specifications	292
7.7.4.2.1 BehavioralFeature_Mapping	292
7.7.4.2.2 Classifier_Mapping	292
7.7.4.2.3 DefaultLowerBound_Mapping	293
7.7.4.2.4 DefaultMultiplicityBoundFeatureMembership_Mapping	294
7.7.4.2.5 DefaultMultiplicityElement_Mapping	295
7.7.4.2.6 DefaultMultiplicityLowerBoundFeatureMembership_Mapping	295
7.7.4.2.7 DefaultMultiplicityMembership_Mapping	296
7.7.4.2.8 DefaultMultiplicityUpperBoundFeatureMembership_Mapping	297
7.7.4.2.9 DefaultUpperBound_Mapping	297
7.7.4.2.10 DefaultValue_Mapping	298
7.7.4.2.11 ElementFeatureMembership_Mapping	299
7.7.4.2.12 Generalization_Mapping	299
7.7.4.2.13 InstanceSpecificationLink_Mapping	300
7.7.4.2.14 InstanceSpecification_Mapping	301
7.7.4.2.15 InstanceSpecificationFeatureTyping_Mapping	302
7.7.4.2.16 InstanceValue_Mapping	303
7.7.4.2.17 InstanceValueMembership_Mapping	304
7.7.4.2.18 LowerBoundValueFeatureMembership_Mapping	305
7.7.4.2.19 MultiplicityElement_Mapping	305
7.7.4.2.20 MultiplicityLowerBoundOwningMembership_Mapping	306
7.7.4.2.21 MultiplicityMembership_Mapping	307
7.7.4.2.22 MultiplicityUpperBoundOwningMembership_Mapping	307
7.7.4.2.23 Operation_Mapping	308
7.7.4.2.24 Parameter_Mapping	309
7.7.4.2.25 ParameterDefaultValue_Mapping	310
7.7.4.2.26 ParameterMembership_Mapping	311
7.7.4.2.27 ParameterSet_Mapping	312
7.7.4.2.28 ParameterSetMembership_Mapping	313
7.7.4.2.29 ParameterSetParameterFeatureMembership_Mapping	313
7.7.4.2.30 ParameterSetParameterReferenceUsage_Mapping	314
7.7.4.2.31 ParameterSetParameterReferenceUsageFeatureValue_Mapping	315
7.7.4.2.32 ParameterSetParameterReferenceUsageFeatureValueExpression_Mapping	315
7.7.4.2.33 ParameterSetParameterReferenceUsageMembership_Mapping	316
7.7.4.2.34 ParameterToFeatureTyping_Mapping	317
7.7.4.2.35 PropertyCommon_Mapping	317
7.7.4.2.36 PropertySubsetting_Mapping	318
7.7.4.2.37 PropertyTypedByClassInterface_Mapping	319
7.7.4.2.38 PropertyUntyped_Mapping	320

7.7.4.2.39 Realization_Mapping	321
7.7.4.2.40 Slot_Mapping	321
7.7.4.2.41 SlotMembership_Mapping	322
7.7.4.2.42 SlotFeatureTyping_Mapping	322
7.7.4.2.43 SlotValue_Mapping	323
7.7.4.2.44 StructuralFeature_Mapping	324
7.7.4.2.45 StructuralFeatureMembership_Mapping	325
7.7.4.2.46 StructuralFeatureToFeatureTyping_Mapping	326
7.7.4.2.47 TypedElementFeatureTyping_Mapping	326
7.7.4.2.48 UpperBoundValueFeatureMembership_Mapping	327
7.7.5 CommonBehavior	328
7.7.5.1 Overview	328
7.7.5.2 UML4SysML::CommonBehavior elements not mapped	328
7.7.5.3 Mapping Specifications	328
7.7.5.3.1 Behavior_Mapping	329
7.7.5.3.2 ChangeEvent_Mapping	329
7.7.5.3.3 OpaqueBehavior_Mapping	330
7.7.5.3.4 OpaqueBehaviorMembership_Mapping	332
7.7.5.3.5 OpaqueBehaviorSpecification_Mapping	332
7.7.5.3.6 TimeEvent_Mapping	333
7.7.5.3.7 Trigger_Mapping	334
7.7.6 CommonStructure	334
7.7.6.1 Overview	334
7.7.6.2 Mapping Specifications	334
7.7.6.2.1 Abstraction_Mapping	334
7.7.6.2.2 Comment_Mapping	335
7.7.6.2.3 CommentAnnotation_Mapping	336
7.7.6.2.4 CommentOwnership_Mapping	337
7.7.6.2.5 Constraint_Mapping	337
7.7.6.2.6 ConstrainedElementFeatureMembership_Mapping	338
7.7.6.2.7 ConstraintUsageFeatureTyping_Mapping	339
7.7.6.2.8 ConstraintUsage_Mapping	340
7.7.6.2.9 Dependency_Mapping	340
7.7.6.2.10 DirectedRelationship_Mapping	341
7.7.6.2.11 ElementMain_Mapping	342
7.7.6.2.12 ElementMembership_Mapping	343
7.7.6.2.13 ElementOwnership_Mapping	343
7.7.6.2.14 ElementOwningMembership_Mapping	344
7.7.6.2.15 NamedElementMain_Mapping	345
7.7.6.2.16 Namespace_Mapping	346
7.7.6.2.17 Relationship_Mapping	346
7.7.6.2.18 Usage_Mapping	347
7.7.7 InformationFlows	347
7.7.7.1 Overview	348
7.7.7.2 Mapping Specifications	348
7.7.7.2.1 InformationFlow_Mapping	348
7.7.7.2.2 InformationFlowConveyedFeatureMembership_Mapping	349
7.7.7.2.3 InformationFlowEnd_Mapping	350
7.7.7.2.4 InformationFlowEndFeatureMembership_Mapping	351
7.7.7.2.5 InformationFlowFeatureTyping_Mapping	352
7.7.7.2.6 InformationFlowSubclassification_Mapping	352
7.7.7.2.7 InformationItem_Mapping	353
7.7.7.2.8 InformationItemFlowConveyedItemUsage_Mapping	354
7.7.7.2.9 InformationItemFlowConveyedItemUsageFeatureTyping_Mapping	354

7.7.8 Interactions	355
7.7.8.1 Overview	355
7.7.8.2 UML4SysML::Interactions elements not mapped	356
7.7.8.3 Mapping Specifications.....	356
7.7.8.3.1 ActionExecutionSpecification_Mapping	356
7.7.8.3.2 BehaviorExecutionSpecification_Mapping.....	357
7.7.8.3.3 CombinedFragment_Mapping.....	357
7.7.8.3.4 CombinedFragmentMembership_Mapping	358
7.7.8.3.5 ExecutionSpecificationMembership_Mapping	359
7.7.8.3.6 Interaction_Mapping	359
7.7.8.3.7 InteractionOperand_Mapping	361
7.7.8.3.8 InteractionOperandMembership_Mapping	362
7.7.8.3.9 InteractionUse_Mapping	362
7.7.8.3.10 InteractionUseMembership_Mapping	363
7.7.8.3.11 InteractionUseFeatureTyping_Mapping	364
7.7.8.3.12 LifelineMembership_Mapping.....	364
7.7.8.3.13 LifelinePartUsage_Mapping	365
7.7.8.3.14 LifelineFeatureTyping_Mapping	366
7.7.8.3.15 Message_Mapping.....	366
7.7.8.3.16 MessageMembership_Mapping	367
7.7.8.3.17 StateInvariant_Mapping	367
7.7.8.3.18 StateInvariantMembership_Mapping	368
7.7.8.3.19 StateInvariantFeatureTyping_Mapping	369
7.7.9 Packages	369
7.7.9.1 Overview	370
7.7.9.2 UML4SysML::Packages elements not mapped	370
7.7.9.3 Mapping Specifications.....	370
7.7.9.3.1 ElementImport_Mapping	370
7.7.9.3.2 Model_Mapping	372
7.7.9.3.3 ModelViewpointMetadataUsage_Mapping	372
7.7.9.3.4 ModelViewpointMetadataFeatureMembership_Mapping	373
7.7.9.3.5 ModelViewpointMetadataReferenceUsage_Mapping	373
7.7.9.3.6 ModelViewpointMetadataFeatureTyping_Mapping	374
7.7.9.3.7 ModelViewpointMetadataMembership_Mapping	374
7.7.9.3.8 ModelViewpointMetadataFeatureValue_Mapping	375
7.7.9.3.9 ModelViewpointMetadataRedefinition_Mapping	376
7.7.9.3.10 ModelViewpointValue_Mapping	376
7.7.9.3.11 Package_Mapping	377
7.7.9.3.12 PackageImport_Mapping	378
7.7.9.3.13 PackageURIMetadataUsage_Mapping	379
7.7.9.3.14 PackageURIFeatureMembership_Mapping	379
7.7.9.3.15 PackageURIFeatureTyping_Mapping	380
7.7.9.3.16 PackageURIMetadataReferenceUsage_Mapping	381
7.7.9.3.17 PackageURIMetadataFeatureValue_Mapping	381
7.7.9.3.18 PackageURIMetadataMembership_Mapping	382
7.7.9.3.19 PackageURIRedefinition_Mapping	383
7.7.9.3.20 PackageURIValue_Mapping	384
7.7.9.3.21 Profile_Mapping	384
7.7.9.3.22 ProfileMetadataMembership_Mapping	385
7.7.9.3.23 ProfileMetadataUsage_Mapping	386
7.7.9.3.24 StereotypeMetadataDefinition_Mapping	386
7.7.9.3.25 StereotypeMetadataDefinitionMembership_Mapping	387
7.7.9.3.26 StereotypeOccurrenceUsage_Mapping	387
7.7.9.3.27 StereotypeOccurrenceUsageFeatureTyping_Mapping	388

7.7.9.3.28 StereotypeOccurrenceUsageMembership_Mapping	388
7.7.9.3.29 StereotypeOccurrenceUsageMultiplicityMembership_Mapping	389
7.7.9.3.30 StereotypeOccurrenceUsageMultiplicityRange_Mapping	390
7.7.9.3.31 StereotypeOccurrenceUsageMultiplicityRangeInfinity_Mapping	390
7.7.9.3.32 StereotypeOccurrenceUsageInfinityReturnParameter_Mapping	391
7.7.9.3.33 StereotypeOccurrenceUsageInfinityReturnParameterMembership_Mapping	392
7.7.9.3.34 StereotypeOccurrenceUsageMultiplicityRangeMembership_Mapping	392
7.7.10 SimpleClassifiers.....	393
7.7.10.1 Overview	393
7.7.10.2 Mapping Specifications.....	394
7.7.10.2.1 Attribute_Mapping	394
7.7.10.2.2 AttributeRedefined_Mapping.....	395
7.7.10.2.3 AttributeRedefinedRedefinition_Mapping.....	396
7.7.10.2.4 AttributeRedefinedMembership_Mapping	396
7.7.10.2.5 AttributeRedefinedFeatureTyping_Mapping	397
7.7.10.2.6 BehavioredClassifier_Mapping.....	397
7.7.10.2.7 BehavioredClassifierFeatureMembership_Mapping.....	399
7.7.10.2.8 BehavioredClassifierFeatureTyping_Mapping	399
7.7.10.2.9 BehavioredClassifierActionUsage_Mapping.....	400
7.7.10.2.10 DataType_Mapping.....	401
7.7.10.2.11 Enumeration_Mapping	401
7.7.10.2.12 EnumerationLiteral_Mapping	402
7.7.10.2.13 EnumerationVariantMembership_Mapping.....	402
7.7.10.2.14 Interface_Mapping	403
7.7.10.2.15 InterfaceConjugatedPortDefinition_Mapping	404
7.7.10.2.16 InterfaceConjugatedPortDefinitionMembership_Mapping.....	405
7.7.10.2.17 InterfacePortConjugation_Mapping	405
7.7.10.2.18 InterfaceRealization_Mapping	406
7.7.10.2.19 PrimitiveType_Mapping	407
7.7.10.2.20 Reception_Mapping	407
7.7.10.2.21 ReceptionFeatureTyping_Mapping	408
7.7.10.2.22 Signal_Mapping	409
7.7.11 StateMachines	409
7.7.11.1 Overview	409
7.7.11.2 Mapping Specifications.....	409
7.7.11.2.1 ConnectionPointReference_Mapping	410
7.7.11.2.2 FinalState_Mapping	410
7.7.11.2.3 PseudoState_Mapping	411
7.7.11.2.4 Region_Mapping	412
7.7.11.2.5 State_Mapping.....	413
7.7.11.2.6 StateDefinition_Mapping	413
7.7.11.2.7 Transition_Mapping	414
7.7.11.2.8 TransitionSuccession_Mapping	415
7.7.11.2.9 TransitionSourceToSubsetting_Mapping	416
7.7.11.2.10 TransitionSuccessionSource_Mapping	417
7.7.11.2.11 TransitionSuccessionSourceMembership_Mapping	417
7.7.11.2.12 TransitionSuccessionTarget_Mapping	418
7.7.11.2.13 TransitionSuccessionTargetMembership_Mapping	419
7.7.11.2.14 TransitionTargetToSubsetting_Mapping	420
7.7.12 StructuredClassifiers	420
7.7.12.1 Overview	420
7.7.12.2 Mapping Specifications.....	421
7.7.12.2.1 AssociationClass_Mapping	421
7.7.12.2.2 AssociationCommon_Mapping	422

7.7.12.2.3 AssociationMetadataUsage_Mapping	423
7.7.12.2.4 AssociationMetadataUsageFeatureMembership_Mapping	423
7.7.12.2.5 AssociationMetadataUsageFeatureTyping_Mapping	424
7.7.12.2.6 AssociationMetadataUsageFeature_Mapping	425
7.7.12.2.7 AssociationMetadataUsageFeatureValue_Mapping	425
7.7.12.2.8 AssociationMetadataUsageMembership_Mapping	426
7.7.12.2.9 AssociationMetadataUsageRedefinition_Mapping	427
7.7.12.2.10 Class_Mapping	427
7.7.12.2.11 ConnectionEndToSubsetting_Mapping	428
7.7.12.2.12 Connector_Mapping	429
7.7.12.2.13 ConnectorEndToFeatureCommon_Mapping	430
7.7.12.2.14 ConnectorEndToMembership_Mapping	431
7.7.12.2.15 ConnectorEndToOwnedFeature_Mapping	431
7.7.12.2.16 ConnectorEndToSubsettedFeature_Mapping	432
7.7.12.2.17 ConnectorEndToSubsettedFeatureMembership_Mapping	433
7.7.12.2.18 ConnectorMultiplicityMembership_Mapping	434
7.7.12.2.19 ConnectorType_Mapping	434
7.7.12.2.20 ConnectorTypeDerived_Mapping	435
7.7.12.2.21 End_Mapping	436
7.7.12.2.22 EndMembership_Mapping	437
7.7.12.2.23 EndToSubsettedFeature_Mapping	437
7.7.12.2.24 EndToSubsettedFeatureChaining_Mapping	438
7.7.12.2.25 NonOwnedEndSubsetting_Mapping	438
7.7.12.2.26 NonOwnedEndToSubsettedFeatureMembership_Mapping	439
7.7.12.2.27 NonOwnedEnd_Mapping	440
7.7.12.2.28 NonOwnedEndMembership_Mapping	441
7.7.12.2.29 NonOwnedEndSubsettingMembership_Mapping	441
7.7.12.2.30 NonOwnedEndFeatureTyping_Mapping	442
7.7.12.2.31 OwnedEnd_Mapping	442
7.7.12.2.32 OwnedEndMembership_Mapping	444
7.7.12.2.33 Port_Mapping	445
7.7.12.2.34 PortUntyped_Mapping	445
7.7.12.2.35 PropertyToFeatureChaining_Mapping	446
7.7.12.2.36 QualifierMembership_Mapping	447
7.7.13 UseCases	447
7.7.13.1 Overview	447
7.7.13.2 UML4SysML::UseCases elements not mapped	448
7.7.13.3 Mapping Specifications	448
7.7.13.3.1 Actor_Mapping	448
7.7.13.3.2 Include_Mapping	448
7.7.13.3.3 IncludeFeatureTyping_Mapping	449
7.7.13.3.4 UseCase_Mapping	450
7.7.13.3.5 UseCaseActor_Mapping	451
7.7.13.3.6 UseCaseActorFeatureTyping_Mapping	452
7.7.13.3.7 UseCaseActorMembership_Mapping	452
7.7.13.3.8 UseCaseEmptySubjectReferenceUsage_Mapping	453
7.7.13.3.9 UseCaseObjectiveMembership_Mapping	453
7.7.13.3.10 UseCaseObjectiveRequirementUsage_Mapping	454
7.7.13.3.11 UseCaseObjectiveSubjectMembership_Mapping	455
7.7.13.3.12 UseCaseSubjectFeatureTyping_Mapping	455
7.7.13.3.13 UseCaseSubjectMembership_Mapping	456
7.7.13.3.14 UseCaseSubjectReferenceUsage_Mapping	457
7.7.14 Values	457
7.7.14.1 Overview	458

7.7.14.2 UML4SysML::Values elements not mapped	458
7.7.14.3 Mapping Specifications	459
7.7.14.3.1 EqualOperatorExpressionFeature_Mapping	459
7.7.14.3.2 EqualOperatorExpressionFeatureValue_Mapping	460
7.7.14.3.3 EqualOperatorExpressionOperandParameterMembership_Mapping	460
7.7.14.3.4 Expression_Mapping	461
7.7.14.3.5 ExpressionElse_Mapping	462
7.7.14.3.6 ExpressionElseMembership_Mapping	462
7.7.14.3.7 ExpressionElseSpecification_Mapping	463
7.7.14.3.8 LiteralBoolean_Mapping	464
7.7.14.3.9 LiteralInteger_Mapping	464
7.7.14.3.10 LiteralNull_Mapping	465
7.7.14.3.11 LiteralReal_Mapping	465
7.7.14.3.12 LiteralSpecificationCommon_Mapping	466
7.7.14.3.13 LiteralSpecificationFeatureTyping_Mapping	467
7.7.14.3.14 LiteralString_Mapping	467
7.7.14.3.15 LiteralUnlimitedUnbounded_Mapping	468
7.7.14.3.16 LiteralUnlimitedInteger_Mapping	468
7.7.14.3.17 OpaqueExpressionAsValue_Mapping	469
7.7.14.3.18 OpaqueExpression_Mapping	470
7.7.14.3.19 OpaqueExpressionFeature_Mapping	470
7.7.14.3.20 OpaqueExpressionFeatureFeature_Mapping	471
7.7.14.3.21 OpaqueExpressionFeatureFeatureMembership_Mapping	471
7.7.14.3.22 OpaqueExpressionFeatureValue_Mapping	472
7.7.14.3.23 OpaqueExpressionFeatureValueExpression_Mapping	473
7.7.14.3.24 OpaqueExpressionFeatureValueExpressionMembership_Mapping	473
7.7.14.3.25 OpaqueExpressionMembership_Mapping	474
7.7.14.3.26 OpaqueExpressionParameterMembership_Mapping	475
7.7.14.3.27 OpaqueExpressionReferenceUsageReturnParameterMembership_Mapping	475
7.7.14.3.28 OpaqueExpressionReferenceUsage_Mapping	476
7.7.14.3.29 OpaqueExpressionReferenceUsageFeatureTyping_Mapping	477
7.7.14.3.30 OpaqueExpressionReferenceUsageUntyped_Mapping	477
7.7.14.3.31 OpaqueExpressionSpecification_Mapping	478
7.7.14.3.32 TimeExpression_Mapping	478
7.7.14.3.33 ValueSpecification_Mapping	479
7.8 Mappings from SysML v1.7 stereotypes	480
7.8.1 Overview	480
7.8.2 Activities	480
7.8.2.1 Overview	480
7.8.2.2 SysML::Activities elements not mapped	481
7.8.2.3 Mapping Specifications	481
7.8.2.3.1 ProbabilityMetadataUsage_Mapping	481
7.8.2.3.2 ProbabilityMetadataUsageFeatureMembership_Mapping	482
7.8.2.3.3 ProbabilityMetadataUsageFeatureTyping_Mapping	483
7.8.2.3.4 ProbabilityMetadataUsageReferenceUsage_Mapping	483
7.8.2.3.5 ProbabilityMetadataUsageReferenceUsageFeatureValue_Mapping	484
7.8.2.3.6 ProbabilityMetadataUsageReferenceUsageRedefinition_Mapping	485
7.8.2.3.7 ProbabilityOwningMembership_Mapping	486
7.8.2.3.8 RateMetadataUsage_Mapping	486
7.8.2.3.9 RateMetadataUsageContinuousFeatureMembership_Mapping	488
7.8.2.3.10 RateMetadataUsageFeatureValue_Mapping	488
7.8.2.3.11 RateMetadataUsageContinuousReferenceUsage_Mapping	489
7.8.2.3.12 RateMetadataUsageContinuousReferenceUsageRedefinition_Mapping	490
7.8.2.3.13 RateMetadataUsageDiscreteFeatureMembership_Mapping	491

7.8.2.3.14 RateMetadataUsageDiscreteReferenceUsage_Mapping	491
7.8.2.3.15 RateMetadataUsageDiscreteReferenceUsageRedefinition_Mapping	492
7.8.2.3.16 RateMetadataUsageFeatureTyping_Mapping	493
7.8.2.3.17 RateOwningMembership_Mapping	494
7.8.2.3.18 Model Libraries	494
7.8.2.3.18.1 ControlValues	494
7.8.2.3.18.1.1 ControlValueKind	494
7.8.3 Allocations	494
7.8.3.1 Overview	495
7.8.3.2 SysML::Allocations elements not mapped	495
7.8.3.3 Mapping Specifications	495
7.8.3.3.1 Allocation_Mapping	495
7.8.3.3.2 AllocationFeatureMembership_Mapping	497
7.8.3.3.3 AllocationFeatureTyping_Mapping	497
7.8.3.3.4 AllocationReferenceUsage_Mapping	498
7.8.3.3.5 AllocationSourceReferenceUsageRedefinition_Mapping	499
7.8.3.3.6 AllocationTargetFeatureMembership_Mapping	500
7.8.3.3.7 AllocationTargetReferenceUsage_Mapping	500
7.8.3.3.8 AllocationTargetReferenceUsageRedefinition_Mapping	501
7.8.3.3.9 AllocationUsage_Mapping	502
7.8.3.3.10 AllocationUsageEndFeatureMembership_Mapping	502
7.8.3.3.11 AllocationUsageFeature_Mapping	503
7.8.3.3.12 AllocationUsageFeatureChaining_Mapping	504
7.8.3.3.13 AllocationUsageFeatureChainingChainedFeature_Mapping	505
7.8.3.3.14 AllocationUsageFeatureMembership_Mapping	505
7.8.3.3.15 AllocationUsageFeatureSubsetting_Mapping	506
7.8.3.3.16 AllocationUsageFeatureSubsettingFeature_Mapping	507
7.8.3.3.17 AllocationUsageTargetEndFeatureMembership_Mapping	507
7.8.3.3.18 AllocationUsageTargetFeature_Mapping	508
7.8.3.3.19 AllocationUsageTargetFeatureChaining_Mapping	509
7.8.3.3.20 AllocationUsageTargetFeatureSubsetting_Mapping	509
7.8.3.3.21 AllocationUsageTargetFeatureSubsettingFeature_Mapping	510
7.8.4 Blocks	511
7.8.4.1 Overview	511
7.8.4.2 SysML::Blocks elements not mapped	512
7.8.4.3 Mapping Specifications	513
7.8.4.3.1 AssociationBlock_Mapping	513
7.8.4.3.2 BindingConnector_Mapping	514
7.8.4.3.3 Block_Mapping	514
7.8.4.3.4 EncapsulatedBlock_Mapping	515
7.8.4.3.5 EncapsulatedBlockMetadataMembership_Mapping	517
7.8.4.3.6 EncapsulatedBlockMetadata_Mapping	517
7.8.4.3.7 EncapsulatedBlockMetadataFeatureMembership_Mapping	518
7.8.4.3.8 EncapsulatedBlockMetadataFeatureTyping_Mapping	518
7.8.4.3.9 EncapsulatedBlockMetadataReferenceUsage_Mapping	519
7.8.4.3.10 EncapsulatedBlockMetadataFeatureValue_Mapping	520
7.8.4.3.11 EncapsulatedBlockMetadataRedefinition_Mapping	520
7.8.4.3.12 PartProperty_Mapping	521
7.8.4.3.13 Model Libraries	522
7.8.4.3.13.1 PrimitiveValueTypes	522
7.8.4.3.13.1.1 Boolean	522
7.8.4.3.13.1.2 Complex	522
7.8.4.3.13.1.3 Integer	522
7.8.4.3.13.1.4 Number	522

7.8.4.3.13.1.5 Real	522
7.8.4.3.13.1.6 String	522
7.8.4.3.13.2 UnitAndQuantityKind	522
7.8.4.3.13.2.1 QuantityKind	523
7.8.4.3.13.2.2 Unit	523
7.8.4.3.14 ValueType_Mapping	523
7.8.5 ConstraintBlocks	523
7.8.5.1 Overview	524
7.8.5.2 Mapping Specifications	524
7.8.5.2.1 ConstraintBlock_Mapping	524
7.8.5.2.2 ConstraintParameter_Mapping	525
7.8.6 Model Elements	526
7.8.6.1 Overview	526
7.8.6.2 SysML::ModelElements elements not mapped	526
7.8.6.3 Mapping Specifications	527
7.8.6.3.1 ProblemRationaleMetadataFeatureMembership_Mapping	527
7.8.6.3.2 ProblemRationaleMetadataFeatureTyping_Mapping	527
7.8.6.3.3 ProblemRationaleMetadataReferenceUsage_Mapping	528
7.8.6.3.4 ProblemRationaleMetadataFeatureValue_Mapping	529
7.8.6.3.5 ProblemRationaleMetadataMembership_Mapping	529
7.8.6.3.6 Concern_Mapping	530
7.8.6.3.7 ConcernDocumentation_Mapping	531
7.8.6.3.8 ConcernOwningMembership_Mapping	532
7.8.6.3.9 ConcernStakeholderMembership_Mapping	533
7.8.6.3.10 ConcernStakeholderPartUsage_Mapping	533
7.8.6.3.11 ConcernStakeholderPartUsageFeatureTyping_Mapping	534
7.8.6.3.12 ConcernStakeholderPartUsageOwningMembership_Mapping	535
7.8.6.3.13 ConcernStakeholderPartUsageFeature_Mapping	535
7.8.6.3.14 ElementGroup_Mapping	536
7.8.6.3.15 ElementGroupMetadataMembership_Mapping	537
7.8.6.3.16 ElementGroupMetadataFeatureMembership_Mapping	537
7.8.6.3.17 ElementGroupMetadataFeatureTyping_Mapping	538
7.8.6.3.18 ElementGroupMetadataFeatureValue_Mapping	539
7.8.6.3.19 ElementGroupMetadataRedefinition_Mapping	539
7.8.6.3.20 ElementGroupMetadataReferenceUsage_Mapping	540
7.8.6.3.21 ElementGroupMetadataUsage_Mapping	541
7.8.6.3.22 ProblemRationale_Mapping	541
7.8.6.3.23 ProblemRationaleMetadataRedefinition_Mapping	542
7.8.6.3.24 ProblemRationaleMetadataUsage_Mapping	543
7.8.6.3.25 Stakeholder_Mapping	544
7.8.6.3.26 StakeholderMetadataUsage_Mapping	545
7.8.6.3.27 StakeholderMetadataFeatureMembership_Mapping	546
7.8.6.3.28 StakeholderMetadataFeatureTyping_Mapping	547
7.8.6.3.29 StakeholderMetadataOwningMembership	547
7.8.6.3.30 StakeholderMetadataReferenceUsage_Mapping	548
7.8.6.3.31 StakeholderMetadataReferenceUsageFeatureValue_Mapping	548
7.8.6.3.32 StakeholderMetadataReferenceUsageRedefinition_Mapping	549
7.8.6.3.33 Viewpoint_Mapping	550
7.8.6.3.34 ViewpointConcernReferenceSubsetting_Mapping	552
7.8.6.3.35 ViewpointConcernUsage_Mapping	552
7.8.6.3.36 ViewpointConstraintUsage_Mapping	553
7.8.6.3.37 ViewpointConstraintUsageDocumentation_Mapping	553
7.8.6.3.38 ViewpointConstraintUsageOwningMembership_Mapping	554
7.8.6.3.39 ViewpointFramedConcernMembership_Mapping	555

7.8.6.3.40 ViewpointLanguagesMetadataFeatureMembership_Mapping	555
7.8.6.3.41 ViewpointLanguagesMetadataFeatureValue_Mapping	556
7.8.6.3.42 ViewpointLanguagesMetadataRedefinition_Mapping	557
7.8.6.3.43 ViewpointLanguagesMetadataReferenceUsage_Mapping	557
7.8.6.3.44 ViewpointMetadataFeatureTyping_Mapping	558
7.8.6.3.45 ViewpointLanguagesMetadataOperatorExpression_Mapping	558
7.8.6.3.46 ViewpointMetadataOwningMembership_Mapping	559
7.8.6.3.47 ViewpointMetadataUsage_Mapping	560
7.8.6.3.48 ViewpointPresentationsMetadataFeatureMembership_Mapping	560
7.8.6.3.49 ViewpointPresentationsMetadataFeatureValue_Mapping	561
7.8.6.3.50 ViewpointPresentationsMetadataOperatorExpression_Mapping	562
7.8.6.3.51 ViewpointPresentationsMetadataRedefinition_Mapping	562
7.8.6.3.52 ViewpointPresentationsMetadataReferenceUsage_Mapping	563
7.8.6.3.53 ViewpointRenderingFeatureMembership_Mapping	564
7.8.6.3.54 ViewpointRenderingUsage_Mapping	564
7.8.6.3.55 ViewpointRenderingUsageActionUsage_Mapping	565
7.8.6.3.56 ViewpointRenderingUsageActionUsageFeatureMembership_Mapping	566
7.8.6.3.57 ViewpointRenderingUsageActionUsageFeatureTyping_Mapping	566
7.8.6.3.58 ViewpointRequirementConstraintMembership_Mapping	567
7.8.6.3.59 ViewpointSatisfyFeatureMembership_Mapping	567
7.8.6.3.60 ViewpointSatisfyRequirementUsage_Mapping	568
7.8.6.3.61 ViewpointSatisfyRequirementUsageReferenceSubsetting_Mapping	569
7.8.6.3.62 ViewpointViewpointUsage_Mapping	569
7.8.6.3.63 ViewpointViewpointUsageFeatureMembership_Mapping	570
7.8.7 PortsAndFlows	571
7.8.7.1 Overview	571
7.8.7.2 SysML::Ports&Flows elements not mapped	571
7.8.7.3 Mapping Specifications	572
7.8.7.3.1 AcceptChangeStructuralFeatureEventAction_Mapping	572
7.8.7.3.2 CommonFullPort_Mapping	572
7.8.7.3.3 FeatureDirectionKind	573
7.8.7.3.4 FlowDirectionKind	573
7.8.7.3.5 FullPort_Mapping	574
7.8.7.3.6 FullPortMetadata_Mapping	574
7.8.7.3.7 FullPortMetadataFeatureMembership_Mapping	575
7.8.7.3.8 FullPortMetadataFeatureTyping_Mapping	576
7.8.7.3.9 FullPortMetadataOwningMembership_Mapping	576
7.8.7.3.10 FullPortMetadataReferenceUsage_Mapping	577
7.8.7.3.11 FullPortMetadataReferenceUsageFeatureValue_Mapping	578
7.8.7.3.12 FullPortMetadataReferenceUsageRedefinition_Mapping	578
7.8.7.3.13 FullPortUntyped_Mapping	579
7.8.7.3.14 InterfaceBlock_Mapping	580
7.8.7.3.15 InterfaceBlockConjugated_Mapping	580
7.8.7.3.16 OperationDirectedFeature_Mapping	581
7.8.8 Requirements	582
7.8.8.1 Overview	582
7.8.8.2 SysML::Requirements elements not mapped	583
7.8.8.3 Mapping Specifications	583
7.8.8.3.1 DeriveReqt_Mapping	583
7.8.8.3.2 DeriveReqtFeatureTyping_Mapping	584
7.8.8.3.3 DeriveReqtSourceEndFeatureMembership_Mapping	585
7.8.8.3.4 DeriveReqtSourceFeature_Mapping	585
7.8.8.3.5 DeriveReqtSourceFeatureReferenceSubsetting_Mapping	586
7.8.8.3.6 DeriveReqtTargetEndFeatureMembership_Mapping	586

7.8.8.3.7 DeriveReqtTargetFeature_Mapping.....	587
7.8.8.3.8 DeriveReqtTargetFeatureReferenceSubsetting_Mapping.....	588
7.8.8.3.9 Refine_Mapping.....	588
7.8.8.3.10 RefineAnnotation_Mapping.....	589
7.8.8.3.11 RefineMetadataFeatureMembership_Mapping.....	590
7.8.8.3.12 RefineMetadataReferenceUsage_Mapping.....	591
7.8.8.3.13 RefineMetadataReferenceUsageFeatureValue_Mapping	591
7.8.8.3.14 RefineMetadataReferenceUsageRedefinition_Mapping.....	592
7.8.8.3.15 RefineMetadataUsage_Mapping	593
7.8.8.3.16 RefineMetadataUsageFeatureTyping_Mapping	593
7.8.8.3.17 Requirement_Mapping.....	594
7.8.8.3.18 RequirementDocumentation_Mapping	595
7.8.8.3.19 RequirementDocumentationMembership_Mapping	596
7.8.8.3.20 RequirementSubject_Mapping	596
7.8.8.3.21 RequirementSubjectMembership_Mapping.....	597
7.8.8.3.22 Satisfy_Mapping	598
7.8.8.3.23 SatisfyReferenceUsage_Mapping	599
7.8.8.3.24 SatisfyReferenceUsageFeatureMembership_Mapping	600
7.8.8.3.25 SatisfySubjectReferenceUsage_Mapping	600
7.8.8.3.26 SatisfySubjectReferenceUsageValue_Mapping.....	601
7.8.8.3.27 SatisfySubjectReferenceUsageValueFeature_Mapping	602
7.8.8.3.28 SatisfySubjectReferenceUsageFeatureChaining_Mapping.....	602
7.8.8.3.29 SatisfySubjectReferenceUsageValueFeatureChainingProperty_Mapping	603
7.8.8.3.30 SatisfySubjectReferenceUsageFeatureValue_Mapping	604
7.8.8.3.31 SatisfySubjectReferenceUsageValueOwningMembership_Mapping.....	604
7.8.8.3.32 SatisfySubjectSubjectMembership_Mapping	605
7.8.8.3.33 SatisfyFeatureTyping_Mapping	606
7.8.8.3.34 SatisfyReferenceUsageFeatureTyping_Mapping	606
7.8.8.3.35 TestCaseActivity_Mapping.....	607
7.8.8.3.36 TestCaseActivityReturnParameterMembership_Mapping	608
7.8.8.3.37 TestCaseVerifyObjectiveMembership_Mapping	608
7.8.8.3.38 TestCaseVerifyObjectiveRequirementUsage_Mapping	609
7.8.8.3.39 TestCaseVerifyRequirementUsageReferenceSubsetting_Mapping	609
7.8.8.3.40 TestCaseVerifyRequirementUsage_Mapping	610
7.8.8.3.41 Trace_Mapping	611
7.8.8.3.42 TraceAnnotation_Mapping	612
7.8.8.3.43 TraceMetadataFeatureMembership_Mapping	612
7.8.8.3.44 TraceMetadataReferenceUsage_Mapping	613
7.8.8.3.45 TraceMetadataReferenceUsageFeatureValue_Mapping	614
7.8.8.3.46 TraceMetadataReferenceUsageRedefinition_Mapping	614
7.8.8.3.47 TraceMetadataUsage_Mapping	615
7.8.8.3.48 TraceMetadataUsageFeatureTyping_Mapping	616
7.8.8.3.49 Verify_Mapping	616
7.8.8.3.50 Model Libraries	617
7.8.8.3.50.1 Verdicts.....	617
7.8.8.3.50.1.1 VerdictKind	617

List of Tables

1. List of all mappings	119
2. List of SysML v1 elements not mapped of this section.....	121
3. List of all mappings	248
4. List of SysML v1 elements not mapped of this section.....	249
5. List of all mappings	291
6. List of all mappings	328
7. List of SysML v1 elements not mapped of this section.....	328
8. List of all mappings	334
9. List of all mappings	334
10. List of all mappings	348
11. List of all mappings	355
12. List of SysML v1 elements not mapped of this section.....	356
13. List of all mappings	370
14. List of SysML v1 elements not mapped of this section.....	370
15. List of all mappings	393
16. List of all mappings	409
17. List of all mappings	421
18. List of all mappings	447
19. List of SysML v1 elements not mapped of this section.....	448
20. List of all mappings	458
21. List of SysML v1 elements not mapped of this section.....	459
22. List of all mappings	480
23. List of SysML v1 elements not mapped of this section.....	481
24. List of all mappings	495
25. List of SysML v1 elements not mapped of this section.....	495
26. List of all mappings	511
27. List of SysML v1 elements not mapped of this section.....	512
28. List of all mappings	524
29. List of all mappings	526
30. List of SysML v1 elements not mapped of this section.....	527
31. List of all mappings	571
32. List of SysML v1 elements not mapped of this section.....	572
33. List of all mappings	582
34. List of SysML v1 elements not mapped of this section.....	583

0 Preface

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 <https://www.omg.org/>.

OMG Specifications

As noted, OMG specifications address middleware, modeling, and vertical domain frameworks. All OMG Specifications are available from the OMG website at: <https://www.omg.org/spec>

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
9C Medway Road, PMB 274
Milford, MA 01757
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320

Email: pubs@omg.org

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

Issues

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 <https://www.omg.org>, under Specifications, Report an Issue.

1 Scope

This specification describes a transformation for a semantic translation from SysML v1 [SysMLv1] to SysML v2 [SysMLv2] in a precise way. (In this document, "SysML v1" refers to SysML v1.7, the last version of SysML prior to v2.0, and "SysML v2" refers to SysML v2.0, or whatever version corresponds to the current version of this specification.)

The main intent is to provide the rules on which automated conversions of SysML v1 models to the SysML v2 standard can be developed. In addition, this annex can be considered an educational document that provides useful information for people who would like to compare using SysML v2 and using SysML v1.

More sophisticated applications of this transformation can also be envisaged. For instance, a SysML v1 conformant tool could use this transformation to implement a limited subset of the SysML v2 API that will provide "SysMLv2-like" read-only access to its SysMLv1 models for external applications.

2 Conformance

A tool shall demonstrate *conformance* with this specification by meeting all of the following requirements.

1. The tool shall implement the UML4SysML abstract syntax and SysML v1 profile conformant with [SysMLv1]. The tool should, but is not required, to provide the ability to import a SysML v1 model using standard XMI Model Interchange format [XMI].
2. The tool shall implement the SysML v2 abstract syntax conformant with [SysML v2]. The tool should, but is not required, to provide the ability to export a SysML v2 model KerML-standard model interchange project (see [KerML], Clause 10; see also [SysML v2], Clause 2).
3. The tool shall implement a transformation from an abstract syntax representation of an input SysML v1 model to the abstract syntax representation of an output SysML v2, as specified in view link does not exist of this specification.

A tool may claim *partial conformance* with this specification by satisfying the first two requirements above, but only implementing an identified subset of the mappings specified in view link does not exist and view link does not exist . (Note that care must also be taken that certain mappings depend on other mappings, and so cannot reasonably be implemented separately.)

Note. A tool that conforms to [SysMLv2] is not required to necessarily implement a transformation conformant with this specification, or it may implement a SysML v1 to v2 transformation that is not claimed to conform with the transformation defined in this specification.

3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification.

[KerML] *Kernel Modeling Language (KerML)*, Version 1.0
<https://www.omg.org/spec/KerML/1.0>

[MOF] *Meta Object Facility*, Version 2.5.1
<https://www.omg.org/spec/MOF/2.5.1>

[OCL] *Object Constraint Language*, Version 2.4
<https://www.omg.org/spec/OCL/2.4>

[SysML v1] *OMG Systems Modeling Language (SysML)*, Version 1.7
<https://www.omg.org/spec/SysML/1.7>

[SysML v2] *OMG Systems Modeling Language (SysML)*, Version 2.0
<https://www.omg.org/spec/SysML/2.0>

[UML] *Unified Modeling Language (UML)*, Version 2.5.1
<https://www.omg.org/spec/UML/2.5.1>

[XMI] *XML Metadata Interchange*, Version 2.5.1
<https://www.omg.org/spec/XMI/2.5.1>

4 Terms and Definitions

Various terms and definitions are specified throughout the body of this specification.

5 Symbols

No special symbols are defined in this specification.

6 Introduction

6.1 Mapping Approach

The SysML v1 to v2 transformation is specified by directional mappings between UML metaclasses or stereotypes that are part of the SysML v1 specification [SysMLv1] (referenced below as the "SysML v1 scope") on the one hand, and the set of the metaclasses defined in the KerML [KerML] and SysMLv2 [SysMLv2] specifications (referenced below as "SysML v2") in the other hand. Some library classes are also involved.

Each mapping is a directed relationship that reifies a semantic link between a concept belonging to the SysML v1 scope on the source side and one concept belonging to SysML v2 (or one conforming library element) on the target side. As a set, those mappings constitute a declarative specification of a formal transformation that describes how the information encoded by the SysML v1 concepts can be reliably represented using constructs of SysML v2 metaclass instances.

In this approach, a mapping is represented by a UML class that has a pair of associations. One provides the `from` end that designates the source SysML v1 concept, while the other provides the `to` end that designates the target SysML v2 metaclass.

In addition to those associations, a mapping class provides a set of operations defining how the values of non-derived properties of the target metaclass instance have to be computed based on property values reachable from the source object. The computation algorithm is provided by the body condition of those operations and expressed using OCL code.

Note that the values assigned to the properties of the target object shall be instances of SysML v2 metaclasses, coming themselves from transformations of SysMLv1 objects to SysMLv2 objects. Since the specification is declarative, the order in which the individual transformations shall happen is not imposed. It is up to a conforming implementation to deal with this. Instead, the `getMapped` static operation is provided for referring to the result of a transformation from within an OCL rule. It returns a (possibly undefined) value, that is typed by the target metaclass of the mapping class from which it is invoked.

Each mapping class enables the transformation of any object that has the type specified by the `from` role to an object of the type specified by the `to` role, as long as it is not overloaded by a more specific mapping definition. In other words, assume a mapping is specified for the class `A` (i.e., it has `A` typing its `from` property), then it applies to any instance of a class `B` if `B` is a subclass of `A` and if there is no specialization of that mapping class specified for `B` (i.e., that has `B` typing its `from` property).

It is possible to restrict the applicability of a mapping specification to a specific subset of objects. This is achieved by the `filter` static operation that is evaluated against each candidate object. Only objects of the appropriate type for which this `filter` operation returns `true` shall be translated according to the specifications of that mapping class. The default `filter` operation always returns `true`.

Some mapping classes have one or more qualifiers for their `to` attribute. In such a case, each of those qualifiers reflects the specific property of the source type (i.e. the type of the `from` attribute) that has the same name and the same type. For those specific mappings, it is expected to get one instance of the target class (as specified by the type of the `to` attribute") for each actual combination of value of those properties for a given instance of object of the source type, assuming they pass the applicability filter as described above.

6.2 Acknowledgements

The primary authors of this specification document (and also developers of a proof-of-concept implementation of it) are:

- Yves Bernard, Airbus
- Tim Weilkiens, oose

The specification was formally submitted for standardization by the following organizations:

- 88solutions Corporation
- Dassault Systèmes
- GfSE e.V.
- IBM
- INCOSE
- Intercax LLC
- Lockheed Martin Corporation
- MITRE
- Model Driven Solutions, Inc.
- PTC
- Simula Research Laboratory AS
- Thematix Partners LLC

However, work on the specification was also supported by over 200 people in over 80 organizations that participated in the SysML v2 Submission Team (SST), by contributing use cases, providing critical review and comment, and validating the language design. The following individuals had leadership roles in the SST:

- Manas Bajaj, Intercax LLC (API and services development lead)
- Yves Bernard, Airbus (v1 to v2 transformation co-lead)
- Bjorn Cole, Lockheed Martin Corporation (metamodel development co-lead)
- Sanford Friedenthal, SAF Consulting (SST co-lead, requirements V&V lead)
- Charles Galey, Lockheed Martin Corporation (metamodel development co-lead)
- Karen Ryan, Siemens (metamodel development co-lead)
- Ed Seidewitz, Model Driven Solutions (SST co-lead, pilot implementation lead)
- Tim Weilkiens, oose (v1 to v2 transformation co-lead)

The specification was prepared using CATIA No Magic modeling tools and the OpenMBEE system for model publication (<http://www.openmbee.org>), with the invaluable support of the following individuals:

- Tyler Anderson, No Magic/Dassault Systèmes
- Christopher Delp, Jet Propulsion Laboratory
- Ivan Gomes, Twingineer
- Doris Lam, Jet Propulsion Laboratory
- Robert Karban, Jet Propulsion Laboratory
- Christopher Klotz, No Magic/Dassault Systèmes
- John Watson, Lightstreet Consulting

7 Mappings

7.1 Overview

This Clause is organized in order to match the packages that subdivide the model of the transformation. The Foundations package gathers the abstract classes that represent the concepts on top of which the mapping approach is built. The next subclause presents a utility class named `Helper` that provides reusable operations that simplify the OCL statements defining the computation rules of target properties and make them more readable. Libraries play an important role in SysML v2, and a specific one has been created in order to represent semantics equivalent to those of UML/SysML concepts, where needed. It is presented in this subclause as well.

The three next subclauses are dedicated to initializers, factories and generic mappings, respectively. They do not specify mappings, strictly speaking. Instead, they factorize more or less advanced OCL code that will be reused by the actual mapping specifications that are contained in the two last subclauses. The first of them is dedicated to UML metaclass from the UML4SYSML scope, while the second deals with SysML stereotypes more specifically.

7.2 Foundations

7.2.1 Overview

The concepts defined by KerML/SysML v2 are relatively similar to those of UML/SysML v1, but the ways they are built are different. This makes the specification of the global transformation quite complex. In order to keep it manageable, specific kinds of foundational classes are provided. They represent concepts on which classical "model to model" transformation technologies rely:

- The mappings built on top of the abstract class `Mapping` shall be executed only when they are explicitly called. Each call shall produce a new target element, whatever the source element. It specifies a `from` property typed by the `UML::CommonStructure::Element` metaclass that shall be redefined by any of its subclass for specifying the convenient type of source element. Also it specifies a default (neutral) filter and a set of `getMapped` operations for various purposes: regular mapping result, qualified mapping result and mapping result for a collection of elements.
- The mappings built on top of the abstract class `UniqueMapping`, specified as a specialization of the `Mapping` class, shall produce only one target element for a given source element, whatever the number of time they are called.
- The mappings built on top of the abstract class `MainMapping`, specified as a specialization of the `UniqueMapping` class, shall be systematically executed (i.e. implicitly called) for all the elements that match both theirs source type and filter. There can be at most one main mapping for a given source type and only one target element shall be produced for a given source element.

The corresponding classes are located the the Foundations package.

Sometimes, it is necessary to be able to generate elements in the target model without having to provide an explicit link with a source element. In such a case, a mapping class is not appropriate. Instead the mapping framework provides the concept of a `Factory`.

Last, the concept of an `Initializer` allows the factorization of the specification of properties' default values that can be inherited by mappings and factories, as convenient.

In the model of the transformation that is specified here, all of the abstract classes of this Foundations package are subject to direct or indirect subclassing. In other words, this specification is built as a set of interrelated initializers, factories, regular, unique and main mappings, where the initializers' operation factorizes the specification of default

values for their target element, wherever possible. Those "default operations" are either used as-is or redefined by mappings or factories that can inherit for a specific initializer, as appropriate.

7.2.2 Foundational class specifications

7.2.2.1 UniqueMapping

Description

The mappings built on top of the abstract class UniqueMapping are a specific kind of Mappings that are intended to produce only one target element for a given source element, whatever the number of time they are called. If a getMapped is called several time with the same source element, the target element returned shall always be the same.

Generalizations

- Mapping (from Foundations)

7.2.2.2 Factory

Description

Similarly to the well-known to the homonyms software design pattern, a Factory can be used for specifying the production of a target element without any link with a source element. Factories have in common with mapping classes the operations that specify how the properties of the target element shall be computed and the "to" property that specifies the type of the target element. However factories do not define source element. Instead, they can have parameters. Those parameters, if any, shall be specified by properties with appropriate types and multiplicities. Factories are expected to provide a "create" operation with parameters matching in type and multiplicity the properties that are intended to specify them.

Generalizations

- Initializer (from Foundations)

7.2.2.3 Mapping

Description

This is the generic abstract class that provides the basic features of any mapping class mapping. The mappings built on top of the abstract class Mapping are intended to be executed only when explicitly called (e.g. by the rule of another mapping class). It specifies a "from" property typed by the UML::CommonStructure::Element metaclass that shall be redefined by any of its subclass for specifying the convenient type of source element. Also it specifies a default (neutral) filter and a set of getMapped operations for various purposes: regular mapping result, qualified mapping result and mapping result for a collection of elements. Each call to the getMapped operation shall produce a new target element, whatever the source element provided. Instances of Mapping class are represent a link between one source element and the target element produced by the transformation specified by that mapping class.

Generalizations

- Initializer (from Foundations)

Association Ends

- from : Element [1]

Operations

- filter (in src : Element) : Boolean [1]
returns "true" if the element provided as the actual parameter value can have a mapping to an instance of the type specified by the "to" attribute (i.e. can be used as a value for the "from" attribute)

true

- getMapped (in fromVar : Element) : Element [1]

postConditions:

```
self.filter(fromVar) and
self.to.allFeatures()->selectByKind(UML::Property)->reject(isDerived)
->forAll(p | let ops: Operation = self.allFeatures()
    ->selectByKind(UML::Operation)->any(o | o.name = p.name) in
    p = ops()) and
result = self.to
```

- getMapped (in fromVar : Element, in qual : Element) : Element [1]

postConditions:

```
self.filter(fromVar) and
self.to.allFeatures()->selectByKind(UML::Property)->reject(isDerived)
->forAll(p | let ops: Operation = self.allFeatures()
    ->selectByKind(UML::Operation)->any(o | o.name = p.name) in
    if ops.ownedParameter
        ->select(p | p.direction = UML:: ParameterDirectionKind::_'in')
        ->size()=1 then
        p = ops.qual
    else if ops.ownedParameter
        ->select(p | p.direction = UML:: ParameterDirectionKind::_'in')
        ->size()=0 then
        p = ops()
    else
        invalid
    endif endif) and
result = self.to
```

- getMappedColl (in fromColl : Element) : Element [0..*]

postConditions:

```
result = fromColl->collect(e | self.getMapped(e))
```

7.2.2.4 MainMapping

Description

The mappings built on top of the abstract class MainMapping are a specific kind of UniqueMappings class that are always implicitly called for any element in the source model that match both their source type (as specified by their

"from" property) and their filter condition. If more than one main mapping is specified for a given source type, they shall have filters that specify mutually exclusive conditions. Also, as with any unique mapping, only one target element shall be produced for a given source element.

Generalizations

- UniqueMapping (from Foundations)

7.2.2.5 Initializer

Description

The abstract class Initializer is the common ancestor of Mapping and Factory. It specifies a "to" property typed by the KerML::Root::Element metaclass that shall be redefined by any of its subclass for specifying the convenient type of target element. Initializers are intended to specify reusable properties' computation rules, mainly for initializing them with default values. Those rules will be inherited or redefined by the sub-classes, as appropriate.

Attributes

- /inputs [0..*]

Association Ends

- to : Element [1]

7.3 Mapping Helper and Library

7.3.1 Helper

[**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**](#)

[**SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct**](#)

[**SYSML2-238: ObjectFlows targeting a final node or a activity parameter node cannot be mapped**](#)

[**SYSML2-228: Helpers::activityOwnedRelationships mixes up FinalNodes and FlowFinalNodes**](#)

[**SYSML2-280: ElementMain_Mapping::ownedRelationship is wrong**](#)

[**SYSML2-178: ClassifierBehaviorFeatureMembership_Mapping does not exist**](#)

[**SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct**](#)

Description

The Helper class contains operations that are used by multiple mapping classes. The specification is in the bodyCondition.

Operations

- actionOwnedRelationship (in src : Element) : Relationship [0..*]
Reusable mapping rule for owned relationships of a UML4SysML::Action mapping.

```
let actionInputPin: Set(UML::Element) =
    src.ownedElement->select(e | e.oclIsTypeOf(UML::ActionInputPin)) in
let triggers: Set(UML::Element) =
    src.ownedElement->select(e | e.oclIsKindOf(UML::Trigger)) in
let toElementFMS: Set(UML::Element) =
```

```

src.ownedElement->select(e | e.oclIsKindOf(UML::Pin)) in
let toElementOMS: Set(UML::Element) =
  (((src.ownedElement - toElementFMS) - actionInputPin) - triggers) in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))

```

- activityOwnedRelationship (in src : Element) : Relationship [0..*]

Reusable mapping rule for owned relationships of a UML4SysML::Activity mapping.

```

let initialNodes : Set(UML::Element) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::InitialNode)) in
let flowFinalNodes : Set(UML::Element) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::FlowFinalNode)) in
let ignoreActivityFinalNodes : Set(UML::Element) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::ActivityFinalNode)) in
let ignoreEdgesToActivityFinalNodes : Set(UML::Element) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::ActivityEdge)
    and e.oclaType(UML::ActivityEdge).target.oclIsTypeOf(UML::ActivityFinalNode)) in
let elementsFMS : Set(UML::Element) =
  (((src.ownedElement->select(e | e.oclIsKindOf(UML::ControlNode) or
    e.oclIsKindOf(UML::Action) or e.oclIsKindOf(UML::ControlFlow) or
    e.oclIsKindOf(UML::ObjectFlow) or e.oclIsKindOf(UML::Property)))
    - initialNodes) - flowFinalNodes) - ignoreActivityFinalNodes)
    - ignoreEdgesToActivityFinalNodes in
let parameters: Set(UML::Parameter) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::Parameter)) in
let ignoreParameterNodes: Set(UML::ActivityParameterNode) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::ActivityParameterNode)) in
let ignoreActivityPartition: Set(UML::ActivityPartition) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::ActivityPartition)) in
let ignoreInterruptibleActivityRegion: Set(UML::InterruptibleActivityRegion) =
  src.ownedElement
    ->select(e | e.oclIsKindOf(UML::InterruptibleActivityRegion)) in
let ownedClassifier: Sequence(UML::Classifier) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::Classifier)) in
let variables: Sequence(UML::Variable) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::Variable)) in
let parameterSets: Set(UML::ParameterSet) =
  src.ownedElement->select(e | e.oclIsKindOf(UML::ParameterSet)) in
let elementsOMS: Set(UML::Element) =
  (((((((((src.ownedElement-initialNodes)-flowFinalNodes)-
    ignoreActivityFinalNodes)-ignoreEdgesToActivityFinalNodes)-
    elementsFMS)-parameters)-ignoreParameterNodes)-
    ignoreActivityPartition)-ignoreInterruptibleActivityRegion)-
    ownedClassifier)-variables)-parameterSets)-
    Set{from.classifierBehavior}) in
let memberships : Sequence(UML::Element) =
  elementsOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
  ->union(initialNodes->collect(e | InitialNodeMembership_Mapping.getMapped(e)))
  ->union(flowFinalNodes->collect(e | FlowFinalNodeMembership_Mapping.getMapped(e)))
  ->union(elementsFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
  ->union(variables->collect(e | VariableMembership_Mapping.getMapped(e)))
  ->union(parameterSets->collect(e | ParameterSetMembership_Mapping.getMapped(e)))
  ->union(ownedClassifier
    ->collect(e | ElementOwningMembership_Mapping.getMapped(e))) in
if src.classifierBehavior.oclIsUndefined() then
  memberships
else

```

```

memberships
->append(BehavioredClassifierFeatureMembership_Mapping.getMapped(src))
endif

```

- **createUUID () : String [1]**

Creates a UUID. The specification is implementation-specific and therefore cannot be provided here.

- **excludedPin (in pin : Pin) : Boolean [1]**

Checks if a pin is excluded from the transformation, because it is already defined as a parameter in the SysMLv1Library.

```

if (pin.owner.oclIsTypeOf(UML::AddVariableValueAction) and
    (pin.name = 'value' or pin.name = 'insertAt')) then
    true
else if (pin.owner.oclIsTypeOf(UML::AddStructuralFeatureValueAction) and
    (pin.name = 'value' or pin.name = 'insertAt' or pin.name = 'object')) then
    true
else
    false
endif endif

```

- **getAppliedStereotypes (in element : Element) : Stereotype [0..*]**

Returns the list of applied stereotypes. The specification is implementation-specific and therefore cannot be provided here.

- **getEnumerationType (in t : Enumeration) : EnumerationDefinition [1]**

Maps a given UML4SysM::Enumeration to the appropriate SysML v2 EnumerationDefinition.

```

let enum: SYSML2::EnumerationDefinition =
    Enumeration_Mapping.getMapped(t) in
if enum.oclIsKindOf(SYSML2::EnumerationDefinition) then
    enum
else if t.name = 'VerdictKind' then
    SYSML2::EnumerationDefinition.allInstances()
        ->any(e | e.qualifiedName = 'VerificationCases::VerdictKind')

else if t = UML:: ParameterDirectionKind then
    KerML::FeatureDirectionKind

    else if t.qualifiedName =
        'SysML::Libraries::ControlValues::ControlValueKind' then
        SYSML2::EnumerationDefinition.allInstances()
            ->any(e | e.qualifiedName =
                'SysMLv1Library::Enumerations::ControlValueKind')

    else
        SYSML2::EnumerationDefinition.allInstances()
            ->any(e | e.qualifiedName =
                'SysMLv1Library::Enumerations:' + t.name)
    endif
endif
endif
endif

```

- `getFlowDirectionKind` (in `v : EnumerationLiteral`) : `FeatureDirectionKind` [1]
Maps a given SysMLv1 feature direction enumeration literal to a SysML v2 `FeatureDirectionKind` enumeration literal.

```

if v.enumeration.qualifiedName =
  'SysML::Ports&Flows::FlowDirectionKind' then
  if v = SysML::FlowDirectionKind::_'out' then
    KerML::FeatureDirectionKind::_'out'
  else if (v = SysML::FlowDirectionKind::_'in') then
    KerML::FeatureDirectionKind::_'in'
  else if (v = SysML::FlowDirectionKind::inout) then
    KerML::FeatureDirectionKind::inout
  else
    invalid
  endif endif endif
else
  invalid
endif

```

- `getID` (in `src : Element`) : `String` [1]
Returns the identifier of a UML4SysML::Element. The specification is implementation-specific and therefore cannot provided here.
- `getKerMLFeatureDirectionKind` (in `v : EnumerationLiteral`) : `FeatureDirectionKind` [1]
Maps a given SysMLv1 feature direction enumeration literal to a SysML v2 `FeatureDirectionKind` enumeration literal.

```

if v.enumeration.qualifiedName =
  'SysML::Ports&Flows::FeatureDirectionKind' or
  v.enumeration.qualifiedName = 'SysML::Ports&Flows::FeatureDirection' then
  if v = SysML::FeatureDirectionKind::provided then
    KerML::FeatureDirectionKind::_'out'
  else if (v = SysML::FeatureDirectionKind::required) then
    KerML::FeatureDirectionKind::_'in'
  else if (v = SysML::FeatureDirectionKind::providedRequired) then
    KerML::FeatureDirectionKind::inout
  else
    invalid
  endif endif endif
else
  invalid
endif

```

- `getKerMLParameterDirectionKind` (in `v : ParameterDirectionKind`) : `FeatureDirectionKind` [1]
Maps a given SysMLv1 parameter direction enumeration literal to a SysML v2 `FeatureDirectionKind` enumeration literal.

```

if v = UML::ParameterDirectionKind::_'in' then
  KerML::FeatureDirectionKind::_'in'
else if (v = UML::ParameterDirectionKind::return) then
  KerML::FeatureDirectionKind::out
else if (v = UML::ParameterDirectionKind::out) then
  KerML::FeatureDirectionKind::out
else if (v = UML::ParameterDirectionKind::inout) then
  KerML::FeatureDirectionKind::inout

```

```

        else
            invalid
        endif endif endif
    
```

- **getKerMLVisibilityKind** (in v : VisibilityKind) : VisibilityKind [1]

Maps a given UML4SysML::VisibilityKind enumeration literal to a SysML v2 VisibilityKind enumeration literal.

```

if (v = UML::VisibilityKind::public) then
    KerML::VisibilityKind::public
else if (v = UML::VisibilityKind::protected) then
    KerML::VisibilityKind::protected
else if (v = UML::VisibilityKind::private) then
    KerML::VisibilityKind::private
else if (v = UML::VisibilityKind::package) then
    KerML::VisibilityKind::public
else
    invalid
endif endif endif
    
```

- **getMetadataByName** (in mdName : String) : AttributeDefinition [1]

Returns the metadata attribute definition element for a given metadata name.

```
SYSML2::AttributeDefiniton.allInstances() ->any(e | e.name = mdName)
```

- **getRequirementStereotype** (in element : NamedElement) : Stereotype [0..1]

Returns the requirement stereotype for a given element.

```

let stereotypes: Set(UML::Stereotype) =
    Helper.getAppliedStereotypes(element) in
stereotypes->any(s | s.general->collect(g | g.qualifiedName)
->includes('SysML::Requirements::AbstractRequirement'))
    
```

- **getScalarValueType** (in t : DataType) : DataType [1]

Maps a given SysMLv1 primitive type to a SysMLv2 scalar value type.

```

if t.name = 'UnlimitedNatural' then
    SYSML2::DataType.allInstances()
    ->any(e | e.qualifiedName = 'ScalarValues::Natural')
else
    SYSML2::DataType.allInstances()
    ->any(e | e.qualifiedName = 'ScalarValues::' + t.name)
endif
    
```

- **getScalarValueTypeByName** (in ptName : String) : DataType [1]

Maps a given SysMLv1 primitive type name string to a SysMLv2 scalar value type.

```
SYSML2::DataType.allInstances()
->any(e | e.qualifiedName = 'ScalarValues::' + ptName)
```

- **getTagValue** (in element : Element, in stereotypeName : String, in tagValueName : String) [1]

Returns the value of a stereotype property. The specification is implementation-specific and therefore

cannot provided here.

- `getTagValueAsElement` (in `element : Element`, in `stereotypeName : String`, in `tagValueName : String`) : `Element [1]`
Returns the value of a stereotype property. The specification is implementation-specific and therefore cannot provided here.
- `getTagValueAsElementColl` (in `element : Element`, in `stereotypeName : String`, in `tagValueName : String`) : `Element [0..*]`
Returns the value of a stereotype property as a collection. The specification is implementation-specific and therefore cannot provided here.
- `getTagValueAsString` (in `element : Element`, in `stereotypeName : String`, in `tagValueName : String`) : `String [1]`
Returns the value of a stereotype property as a string. The specification is implementation-specific and therefore cannot provided here.
- `getTagValueAsStringColl` (in `element : Element`, in `stereotypeName : String`, in `tagValueName : String`) : `String [0..*]`
Returns the value of a stereotype property as a string collection. The specification is implementation-specific and therefore cannot provided here.
- `globalNamespace ()` : `Namespace [1]`

```
KerML::Package.allInstances () ->any(p | p.owningNamespace->isEmpty())
```

- `hasMainMapping` (in `element : Element`) : `Boolean [1]`
- `hasStereotypeApplied` (in `element : Element`, in `stereotypeName : String`) : `Boolean [1]`
Returns true if the given stereotype is applied to the element. The specification is implementation-specific and therefore cannot provided here.
- `isConnectionDef` (in `association : Association`) : `Boolean [1]`
Checks if a UML4SysML::Association is mapped to a SysML v2 ConnectionDefinition.

```
-- Case 1: composite association with
-- multiplicity 1..1 on owner side
let case1: Boolean = association.memberEnd
->exists(e | not e.isComposite and e.lower=1) and
association.memberEnd->exists(e | e.isComposite) in

-- Case 2: association is not composite and
-- there is no owned end with multiplicity 0..*
let case2: Boolean = not association.memberEnd
->exists(e | e.isComposite) and
not association.ownedEnd
->exists(e | e.lower = 0 and e.upper = -1) in

association.oclIsTypeOf(UML::AssociationClass) or
case1 or
case2
```

- `isInScope` (in `element : Element`) : `Boolean [1]`
The `isInScope` operation is intended to define the scope on which the transformation will apply. If the

isInScope operation return "true" for a given model element, this element shall be consider by the transformation. Especially, main mappings - if any - will apply to it. It shall be ignored otherwise.

- isRequirement (in element : Element) : Boolean [1]
Checks whether the stereotype AbstractRequirement is applied to the given element.

```
let stereotypes: Set(UML::Stereotype) =
    Helper.getAppliedStereotypes(element) in
stereotypes->exists(s | s.general->collect(g | g.qualifiedName)
->includes('SysML::Requirements::AbstractRequirement'))
```

- packageOwnedRelationship (in src : Element) : Relationship [0..*]
Reusable mapping rule for owned relationships of a UML4SysML::Package mapping.

```
let useCaseAssociations : Set(UML::Association) =
    src.ownedType->select(e | e.oclIsKindOf(UML::Association))
    ->select(a | a.memberEnd->exists(e | e.type.oclIsKindOf(UML::UseCase))) in
let unmappedAssociations : Set(UML::Association) =
    src.ownedType->select(e | e.oclIsKindOf(UML::Association))
    ->reject(a | Helper.isConnectionDef(a)) in
let imports: Set(UML::PackageImport) =
    src.packageImport->select(pi | Helper.isInScope(pi.importedPackage)) in
let relationships: Set(SysMLv2::Relationship) =
    src.ownedComment->reject(c | c.annotatedElement->includes(src))->collect(c| CommentOwners
->union(((src.ownedType-useCaseAssociations)-unmappedAssociations)->collect(e | ElementOwning
->union(imports->collect(i | PackageImport_Mapping.getMapped(i))->asSet())
->union(src.ownedElement->select(e | e.oclIsKindOf(UML::Dependency) or
e.oclIsKindOf(UML::InformationFlow) or e.oclIsKindOf(UML::Package)
or (e.oclIsKindOf(UML::InstanceSpecification) and
e.oclAsType(UML::InstanceSpecification).classifier->notEmpty()))
->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()) in

if src.URI.oclIsUndefined() or src.URI = '' then
    relationships
else
    relationships->including(PackageURIMetadataMembership_Mapping.getMapped(src))
endif
```

- stateOwnedRelationship (in src : Element) : Relationship [0..*]
Reusable mapping rule for owned relationships of a UML4SysML::State mapping.

```
let initialState : Set(UML::Element) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::Pseudostate) and
e.oclAsType(UML::Pseudostate).kind = UML::PseudostateKind::initial) in
let toElementOMS : Set(UML::Element) = from.ownedElement - initialState in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(initialState->collect(e | InitialStateMembership_Mapping.getMapped(e)))
```

7.3.2 SysML v1 Library

The SysML v1 library is a SysML v2 model library with metadata definitions for annotating some model elements resulting from a transformation from a SysML v1 model using the SysML v1 to SysML v2 transformation.

```

package SysMLv1Library {

    doc /*
     * The SysMLv1Library defines library elements and metadata for
     * SysML elements which cannot mapped to a SysML v2 element.
     */

    // Library elements

    action def AddValueAction {
        in insertAt : ScalarValues::Natural [0..1];
        in value : ScalarValues::Integer;
        in isReplaceAll : ScalarValues::Boolean = false;
        in target;

        if not isReplaceAll {
            if insertAt == * {
                assign target := SequenceFunctions:::including(target, value);
            }
            else {
                assign target :=
                    SequenceFunctions:::includingAt(target, value, insertAt);
            }
        } else {
            target := value;
        }
    }

    action def AddStructuralFeatureValueAction :> AddValueAction {
        in object;
    }

    action def RemoveVariableValueAction :> Actions:::AssignmentAction {
        in removeAt: ScalarValues::Natural [0..1];
        in value : ScalarValues::Integer;
        in isRemoveDuplicates : ScalarValues::Boolean = false;
        in variable;

        // isRemoveDuplicates not covered yet

        if removeAt {
            assign variable :=
                SequenceFunctions:::excludingAt(variable, value, removeAt);
        } else {
            assign variable := SequenceFunctions:::excluding(variable, value);
        }
    }

    // Metadata

    metadata def ActivityEdgeData {
        doc /* Metadata definition for UML::ActivityEdge::weight property */
        attribute weight : ScalarValues::Natural;
    }

    metadata def AssociationData {
        doc /* Metadata definition for
         * UML::StructuredClassifiers::Association::isDerived property mapping
         */
    }
}

```

```

        attribute isDerived : ScalarValues::Boolean;
    }

metadata def BlockData {
    doc /* Metadata definition for
        * SysML::Blocks::Block::isEncapsulated property
        */
    attribute isEncapsulated : ScalarValues::Boolean;
}

metadata def ElementGroupData {
    doc /* Metadata definition for the criterion
        * of a SysML::ModelElements::ElementGroup
        */
    attribute criterion : ScalarValues::String;
}

metadata def ModelData :> PackageData {
    doc /* Metadata definition for the UML::Model::viewpoint property */
    :> annotatedElement : SysML::Package;
    attribute 'viewpoint' : ScalarValues::String;
}

metadata def PackageData {
    doc /* Metadata definition for the UML::Package::URI property */
    :> annotatedElement : SysML::Package;
    attribute URI : ScalarValues::String;
}

metadata def ParametersetData {
    doc /* Metadata definition for tagging parameters
        * mapped from a UML::ParameterSet
        */
    attribute isParameterSet : ScalarValues::Boolean;
}

metadata def PortData {
    doc /* Metadata definition for tagging SysML v2 ports
        * mapped from a SysML::Ports&Flows::FullPort element
        */
    :> annotatedElement : SysML::PartUsage;
    attribute isFullPort : ScalarValues::Boolean;
}

metadata def ProbabilityData {
    doc /* Metadata definition for SysML::Activities::Probability stereotype */
    attribute probability : ScalarValues::Real;
}

metadata def RateData {
    doc /* Metadata definition for SysML::Activities::Rate and
        * specialized Discrete and Continuous stereotypes
        */
    :> annotatedElement : SysML::PartUsage;
    part rate;
    attribute isDiscrete : ScalarValues::Boolean;
    attribute isConcrete : ScalarValues::Boolean;
}

metadata def RefineData {

```

```

doc /* Metadata definition for tagging SysML v2 dependencies
 * mapped from a SysML::Requirements::Refine relationship
 */
:> annotatedElement : SysML::Dependency;
attribute isRefine : ScalarValues::Boolean;
}

metadata def StakeholderData {
    doc /* Metadata definition for tagging SysML v2 item definitions
     * mapped from a SysML::ModelElements::Stakeholder element
     */
:> annotatedElement : SysML::ItemDefinition;
attribute isStakeholder : ScalarValues::Boolean;
}

metadata def traceData {
    doc /* Metadata definition for tagging SysML v2 dependencies
     * mapped from a SysML::Requirements::Trace relationship
     */
:> annotatedElement : SysML::Dependency;
attribute isTrace : ScalarValues::Boolean;
}

metadata def ViewpointData {
    doc /* Metadata definition for SysML::ModelElements::Viewpoint properties */
    attribute languages [0..*] : ScalarValues::String;
    attribute presentations [0..*] : ScalarValues::String;
}

package Enumerations {
    enum def ControlValueKind {
        doc /* The ControlValueKind enumeration is a type for
         * treating control values as data and for UML control pins.
         */
        enum disable;
        enum enable;
    }
}
}

```

7.4 Initializers

7.4.1 Overview

The classes presented in this subclause provide set of rules that provide default values for all non-derived features of their target metaclasses. Intentionally, initializers do not specify any "source" element. This makes them easier to specialize but prevents them from being able to provide a computation algorithm for some target features. In such a case, the operation matching the feature will be specified as abstract.

7.4.2 Mapping Specifications

7.4.2.1 KerML Initializers

7.4.2.1.1 AnnotatingElement_Init

Description

Initializes the properties of the SysML v2 element AnnotatingElement.

Generalizations

- Element_Init (from KerMLInitializers)

Association Ends

- to : AnnotatingElement [1]
(redefines: Element_Init::to)

Operations

- annotation () : Annotation [0..*]

Set { }

7.4.2.1.2 Annotation_Init

Description

Initializes the properties of the SysML v2 element Annotation.

Generalizations

- Relationship_Init (from KerMLInitializers)

Attributes

- to : Annotation [1]

Operations

- annotatedElement () : Element [1] {redefines target, abstract}
- annotatingElement () : AnnotatingElement [1] {redefines source, abstract}
- owningAnnotatedElement () : Element [0..1]

null

7.4.2.1.3 Association_Init

Description

Initializes the properties of the SysML v2 element Association.

Generalizations

- Classifier_Init (from KerMLInitializers)
- Relationship_Init (from KerMLInitializers)

Attributes

- to : Association [1]

7.4.2.1.4 Behavior_Init

Description

Initializes the properties of the SysML v2 element Behavior.

Generalizations

- Classifier_Init (from KerMLInitializers)

Attributes

- to : Behavior [1]

7.4.2.1.5 Classifier_Init

Description

Initializes the properties of the SysML v2 element Classifier.

Generalizations

- Type_Init (from KerMLInitializers)

Attributes

- to : Classifier [1]

7.4.2.1.6 Comment_Init

Description

Initializes the properties of the SysML v2 element Comment.

Generalizations

- AnnotatingElement_Init (from KerMLInitializers)

Association Ends

- to : Comment [1]
(redefines: AnnotatingElement_Init::to)

Operations

- body () : String [1]{abstract}
- locale () : String [1]

null

7.4.2.1.7 Conjugation_Init

Description

Initializes the properties of the SysML v2 element Conjugation.

Generalizations

- Relationship_Init (from KerMLInitializers)

Attributes

- to : Conjugation [1]

Operations

- conjugatedType () : Type [1] {redefines source, abstract}
- originalType () : Type [1] {redefines target, abstract}

7.4.2.1.8 Connector_Init

Description

Initializes the properties of the SysML v2 element Connector.

Generalizations

- Feature_Init (from KerMLInitializers)
- Relationship_Init (from KerMLInitializers)

Attributes

- to : Connector [1]

Operations

- isDirected () : Boolean [1]

false

7.4.2.1.9 Documentation_Init

Description

Initializes the properties of the SysML v2 element Documentation.

Generalizations

- Comment_Init (from KerMLInitializers)

Attributes

- to : Documentation [1]

7.4.2.1.10 Element_Init

Description

This is the general abstract class to be used as an ancestor for any class mapping specification.

Generalizations

- Initializer (from Foundations)

Association Ends

- to : Element [1]
(redefines: Initializer::to)

Operations

- aliasId () : String [0..*]

Set{ }

- declaredName () : String [0..1]

null

- elementId () : String [1]

Helper.createUUID()

- ownedRelationship () : Relationship [0..*]

Set{ }

- shortName () : String [0..1]

null

7.4.2.1.11 EndFeatureMembership_Init

Description

Initializes the properties of the SysML v2 element EndFeatureMembership.

Generalizations

- FeatureMembership_Init (from KerMLInitializers)

Attributes

- to : EndFeatureMembership [1]

7.4.2.1.12 Expression_Init

Description

Initializes the properties of the SysML v2 element Expression.

Generalizations

- Step_Init (from KerMLInitializers)

Attributes

- to : Expression [1]

7.4.2.1.13 Feature_Init

Description

Initializes the properties of the SysML v2 element Feature.

Generalizations

- Type_Init (from KerMLInitializers)

Attributes

- to : Feature [1]

Operations

- direction () : FeatureDirectionKind [0..1]

 null

- isComposite () : Boolean [1]

 false

- isDerived () : Boolean [1]

 false

- isEnd () : Boolean [1]

 false

- isOrdered () : Boolean [1]

 false

- isPortion () : Boolean [1]

 false

- isReadOnly () : Boolean [1]

 false

- isUnique () : Boolean [1]

true

7.4.2.1.14 FeatureChainExpression_Init

Description

Initializes the properties of the SysML v2 element FeatureChainExpression.

Generalizations

- OperatorExpression_Init (from KerMLInitializers)

Attributes

- to : FeatureChainExpression [1]

7.4.2.1.15 FeatureChaining_Init

Description

Initializes the properties of the SysML v2 element FeatureChaining.

Generalizations

- Relationship_Init (from KerMLInitializers)

Attributes

- to : FeatureChaining [1]

Operations

- chainingFeature () : Feature [1] {redefines target, abstract}

7.4.2.1.16 FeatureMembership_Init

Description

Initializes the properties of the SysML v2 element FeatureMembership.

Generalizations

- OwningMembership_Init (from KerMLInitializers)
- TypeFeaturing_Init (from KerMLInitializers)

Attributes

- to : FeatureMembership [1]

Operations

- ownedMemberFeature () : Feature [1] {redefines ownedMemberElement, abstract}

- ownedRelatedElement () : Element [0..*] {redefines ownedRelatedElement}


```
Set{self.ownedMemberFeature () }
```

7.4.2.1.17 FeatureReferenceExpression_Init

Description

Initializes the properties of the SysML v2 element FeatureReferenceExpression.

Generalizations

- Expression_Init (from KerMLInitializers)

Attributes

- to : FeatureReferenceExpression [1]

7.4.2.1.18 FeatureTyping_Init

Description

Initializes the properties of the SysML v2 element FeatureTyping.

Generalizations

- Specialization_Init (from KerMLInitializers)

Attributes

- to : FeatureTyping [1]

Operations

- type () : Type [1] {redefines general, abstract}
- typedFeature () : Feature [1] {redefines specific, abstract}

7.4.2.1.19 FeatureValue_Init

Description

Initializes the properties of the SysML v2 element FeatureValue.

Generalizations

- OwningMembership_Init (from KerMLInitializers)

Attributes

- to : FeatureValue [1]

Operations

- featureWithValue () : Feature [1] {redefines ownedMemberElement, abstract}

- `isDefault ()` : Boolean [1]
 - `false`
- `isInitial ()` : Boolean [1]
 - `false`
- `ownedRelatedElement ()` : Element [0..*] {redefines `ownedRelatedElement`}
 - `Set{self.value ()}`
- `value ()` : Expression [1] {redefines `ownedMemberElement`, abstract}
 - `Set{self.value ()}`

7.4.2.1.20 Function_Init

Description

Initializes the properties of the SysML v2 element Function.

Generalizations

- `Behavior_Init` (from KerMLInitializers)

Attributes

- `to` : Function [1]

7.4.2.1.21 Import_Init

Description

Initializes the properties of the SysML v2 element Import.

Generalizations

- `Relationship_Init` (from KerMLInitializers)

Attributes

- `to` : Import [1]

Operations

- `importedMemberName ()` : String [0..1]

`null`

- `isImportAll ()` : Boolean [1]

`false`

- isRecursive () : Boolean [1]
- false
- source () : Element [1] {redefines source, abstract}
 - target () : Element [1] {redefines target, abstract}
 - visibility () : VisibilityKind [1]

KerML::VisibilityKind::public

7.4.2.1.22 Interaction_Init

Description

Initializes the properties of the SysML v2 element Interaction.

Generalizations

- Association_Init (from KerMLInitializers)
- Behavior_Init (from KerMLInitializers)

Attributes

- to : Interaction [1]

7.4.2.1.23 InvocationExpression_Init

Description

Initializes the properties of the SysML v2 element InvocationExpression.

Generalizations

- Expression_Init (from KerMLInitializers)

Attributes

- to : InvocationExpression [1]

7.4.2.1.24 ItemFlow_Init

Description

Initializes the properties of the SysML v2 element ItemFlow.

Generalizations

- Connector_Init (from KerMLInitializers)

Attributes

- to : ItemFlow [1]

7.4.2.1.25 Membership_Init

Description

Initializes the properties of the SysML v2 element Membership.

Generalizations

- Relationship_Init (from KerMLInitializers)

Attributes

- to : Membership [1]

Operations

- memberElement () : Element [1] {redefines target, abstract}
- memberName () : String [0..1]

null

- memberShortName () : String [0..1]

null

- membershipOwningNamespace () : Element [0..*] {redefines source, abstract}
- visibility () : VisibilityKind [1]

KerML::VisibilityKind::public

7.4.2.1.26 MembershipImport_Init

Description

Initializes the properties of the SysML v2 element MembershipImport.

Generalizations

- Import_Init (from KerMLInitializers)

Attributes

- to : MembershipImport [1]

Operations

- importedMembership () : Namespace [1] {redefines target, abstract}

7.4.2.1.27 Namespace_Init

Description

Initializes the properties of the SysML v2 element Namespace.

Generalizations

- Element_Init (from KerMLInitializers)

Association Ends

- to : Namespace [1]
(redefines: Element_Init::to)

7.4.2.1.28 NamespaceImport_Init

Description

Initializes the properties of the SysML v2 element NamespaceImport.

Generalizations

- Import_Init (from KerMLInitializers)

Attributes

- to : NamespaceImport [1]

Operations

- importedNamespace () : Namespace [1] {redefines target, abstract}

7.4.2.1.29 OperatorExpression_Init

Description

Initializes the properties of the SysML v2 element OperatorExpression.

Generalizations

- Expression_Init (from KerMLInitializers)

Attributes

- to : OperatorExpression [1]

Operations

- operator () : String [1]{abstract}

7.4.2.1.30 OwningMembership_Init

Description

Initializes the properties of the SysML v2 element OwningMembership.

Generalizations

- Membership_Init (from KerMLInitializers)

Attributes

- to : OwningMembership [1]

Operations

- ownedMemberElement () : Element [1] {redefines memberElement, abstract}
- ownedRelatedElement () : Element [0..*] {redefines ownedRelatedElement}

```
Set{self.ownedMemberElement() }
```

7.4.2.1.31 Package_Init

Description

Initializes the properties of the SysML v2 element Package.

Generalizations

- Namespace_Init (from KerMLInitializers)

Attributes

- to : Package [1]

7.4.2.1.32 ParameterMembership_Init

Description

Initializes the properties of the SysML v2 element ParameterMembership.

Generalizations

- FeatureMembership_Init (from KerMLInitializers)

Attributes

- to : ParameterMembership [1]

Operations

- ownedMemberParameter () : Feature [1] {redefines ownedMemberFeature, abstract}
- ownedRelatedElement () : Element [0..*] {redefines ownedRelatedElement}

```
Set{self.ownedMemberParameter() }
```

7.4.2.1.33 Predicate_Init

Description

Initializes the properties of the SysML v2 element Predicate.

Generalizations

- Function_Init (from KerMLInitializers)

Attributes

- to : Predicate [1]

7.4.2.1.34 Redefinition_Init

Description

Initializes the properties of the SysML v2 element Redefinition.

Generalizations

- Subsetting_Init (from KerMLInitializers)

Attributes

- to : Redefinition [1]

Operations

- redefinedFeature () : Feature [1] {redefines subsettedFeature, abstract}
- redefiningFeature () : Feature [1] {redefines subsettingFeature, abstract}

7.4.2.1.35 ReferenceSubsetting_Init

Description

Initializes the properties of the SysML v2 element ReferenceSubsetting.

Generalizations

- Subsetting_Init (from KerMLInitializers)

Attributes

- to : ReferenceSubsetting [1]

Operations

- referencedFeature () : Feature [1] {redefines subsettedFeature, abstract}

7.4.2.1.36 Relationship_Init

Description

Initializes the properties of the SysML v2 element Relationship.

Generalizations

- Element_Init (from KerMLInitializers)

Association Ends

- to : Relationship [1]
(redefines: Element_Init::to)

Operations

- ownedRelatedElement () : Element [0..*]

Set{ }

- source () : Element [0..*]

Set{ }

- target () : Element [0..*]

Set{ }

7.4.2.1.37 ReturnParameterMembership_Init

Description

Initializes the properties of the SysML v2 element ReturnParameterMembership.

Generalizations

- ParameterMembership_Init (from KerMLInitializers)

Attributes

- to : ReturnParameterMembership [1]

Operations

- isComposite (in src : Element) : Boolean [1]
returns "true" if the element provided as the actual parameter value can have a mapping to an instance of the type specified by the "to" attribute (i.e. can be used as a value for the "from" attribute)

false

7.4.2.1.38 Specialization_Init

Description

Initializes the properties of the SysML v2 element Specialization.

Generalizations

- Relationship_Init (from KerMLInitializers)

Attributes

- to : Specialization [1]

Operations

- general () : Type [1] {redefines target, abstract}
- specific () : Type [1] {redefines source, abstract}

7.4.2.1.39 Step_Init

Description

Initializes the properties of the SysML v2 element Step.

Generalizations

- Feature_Init (from KerMLInitializers)

Attributes

- to : Step [1]

7.4.2.1.40 Subclassification_Init

Description

Initializes the properties of the SysML v2 element Subclassification.

Generalizations

- Specialization_Init (from KerMLInitializers)

Attributes

- to : Subclassification [1]

Operations

- subclassifier () : Classifier [1]{abstract}
- superclassifier () : Classifier [1]{abstract}

7.4.2.1.41 Subsetting_Init

Description

Initializes the properties of the SysML v2 element Subsetting.

Generalizations

- Specialization_Init (from KerMLInitializers)

Attributes

- to : Subsetting [1]

Operations

- subsettedFeature () : Feature [1] {redefines general, abstract}

- subsettingFeature () : Feature [1] {redefines specific, abstract}

7.4.2.1.42 Succession_Init

Description

Initializes the properties of the SysML v2 element Succession.

Generalizations

- Connector_Init (from KerMLInitializers)

Attributes

- to : Succession [1]

7.4.2.1.43 SuccessionItemFlow_Init

Description

Initializes the properties of the SysML v2 element SuccessionItemFlow.

Generalizations

- ItemFlow_Init (from KerMLInitializers)
- Succession_Init (from KerMLInitializers)

Attributes

- to : SuccessionItemFlow [1]

7.4.2.1.44 TextualRepresentation_Init

Description

Initializes the properties of the SysML v2 element TextualRepresentation.

Generalizations

- AnnotatingElement_Init (from KerMLInitializers)

Attributes

- to : TextualRepresentation [1]

Operations

- body () : String [1]{abstract}
- language () : String [1]{abstract}

7.4.2.1.45 Type_Init

Description

Initializes the properties of the SysML v2 element Type.

Generalizations

- Namespace_Init (from KerMLInitializers)

Attributes

- to : Type [1]

Operations

- isAbstract () : Boolean [1]

false

- isSufficient () : Boolean [1]

false

7.4.2.1.46 TypeFeaturing_Init

Description

Initializes the properties of the SysML v2 element TypeFeaturing.

Generalizations

- Relationship_Init (from KerMLInitializers)

Attributes

- to : TypeFeaturing [1]

Operations

- featureOfType () : Feature [1] {redefines source, abstract}
- featuringType () : Type [1] {redefines target, abstract}

7.4.2.2 System Initializers

7.4.2.2.1 ActionUsage_Init

Description

Initializes the properties of the SysML v2 element ActionUsage.

Generalizations

- Step_Init (from KerMLInitializers)
- Usage_Init (from SystemInitializers)

Attributes

- to : ActionUsage [1]

Operations

- isComposite () : Boolean [1] {redefines isComposite}

true

7.4.2.2.2 ActorMembership_Init

Description

Initializes the properties of the SysML v2 element ActorMembership.

Generalizations

- ParameterMembership_Init (from KerMLInitializers)

Attributes

- to : ActorMembership [1]

7.4.2.2.3 AssignmentActionUsage_Init

Description

Initializes the properties of the SysML v2 element AssignmentActionUsage.

Generalizations

- ActionUsage_Init (from SystemInitializers)

Attributes

- to : AssignmentActionUsage [1]

7.4.2.2.4 ConjugatedPortDefinition_Init

Description

Initializes the properties of the SysML v2 element ConjugatedPortDefinition.

Generalizations

- PortDefinition_Init (from SystemInitializers)

Attributes

- to : ConjugatedPortDefinition [1]

7.4.2.2.5 ConjugatedPortTyping_Init

Description

Initializes the properties of the SysML v2 element ConjugatedPortTyping.

Generalizations

- FeatureTyping_Init (from KerMLInitializers)

Attributes

- to : ConjugatedPortTyping [1]

Operations

- conjugatedPortDefinition () : ConjugatedPortDefinition [1] {redefines type, abstract}
- portDefinition () : PortDefinition [1]{abstract}

7.4.2.2.6 ConnectionUsage_Init

Description

Initializes the properties of the SysML v2 element ConnectionUsage.

Generalizations

- PartUsage_Init (from SystemInitializers)

Attributes

- to : ConnectionUsage [1]

7.4.2.2.7 ConstraintDefinition_Init

Description

Initializes the properties of the SysML v2 element ConstraintDefinition.

Generalizations

- Definition_Init (from SystemInitializers)

Attributes

- to : ConstraintDefinition [1]

7.4.2.2.8 ConstraintUsage_Init

Description

Initializes the properties of the SysML v2 element ConstraintUsage.

Generalizations

- Usage_Init (from SystemInitializers)

Attributes

- to : ConstraintUsage [1]

7.4.2.2.9 Definition_Init

Description

Initializes the properties of the SysML v2 element Definition.

Generalizations

- Classifier_Init (from KerMLInitializers)

Attributes

- to : Definition [1]

Operations

- isVariation () : Boolean [1]

false

7.4.2.2.10 EventOccurrenceUsage_Init

Description

Initializes the properties of the SysML v2 element EventOccurrenceUsage.

Generalizations

- OccurrenceUsage_Init (from SystemInitializers)

Attributes

- to : EventOccurrenceUsage [1]

7.4.2.2.11 FlowConnectionUsage_Init

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Initializes the properties of the SysML v2 element FlowConnectionUsage.

Generalizations

- ConnectionUsage_Init (from SystemInitializers)

Association Ends

- to : FlowConnectionUsage [1]
(redefines: ConnectionUsage_Init::to)

7.4.2.2.12 ItemDefinition_Init

Description

Initializes the properties of the SysML v2 element ItemDefinition.

Generalizations

- Definition_Init (from SystemInitializers)

Attributes

- to : ItemDefinition [1]

7.4.2.2.13 ItemFeature_Init

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Initializes the properties of the SysML v2 element ItemFeature.

Generalizations

- Feature_Init (from KerMLInitializers)

Association Ends

- to : ItemFeature [1]
(redefines: Feature_Init::to)

7.4.2.2.14 MetadataUsage_Init

Description

Initializes the properties of the SysML v2 element MetadataUsage.

Generalizations

- Usage_Init (from SystemInitializers)

Attributes

- to : MetadataUsage [1]

7.4.2.2.15 ObjectiveMembership_Init

Description

Initializes the properties of the SysML v2 element ObjectiveMembership.

Generalizations

- FeatureMembership_Init (from KerMLInitializers)

Attributes

- to : ObjectiveMembership [1]

7.4.2.2.16 OccurrenceDefinition_Init

Description

Initializes the properties of the SysML v2 element OccurrenceDefinition.

Generalizations

- Definition_Init (from SystemInitializers)

Attributes

- to : OccurrenceDefinition [1]

Operations

- isIndividual () : Boolean [1]

false

7.4.2.2.17 OccurrenceUsage_Init

Description

Initializes the properties of the SysML v2 element OccurrenceUsage.

Generalizations

- Usage_Init (from SystemInitializers)

Attributes

- to : OccurrenceUsage [1]

Operations

- isIndividual () : Boolean [1]

false

- portionKind () : PortionKind [1]{abstract}

7.4.2.2.18 PartUsage_Init

Description

Initializes the properties of the SysML v2 element PartUsage.

Generalizations

- Usage_Init (from SystemInitializers)

Attributes

- to : PartUsage [1]

7.4.2.2.19 PortConjugation_Init

Description

Initializes the properties of the SysML v2 element PortConjugation.

Generalizations

- Conjugation_Init (from KerMLInitializers)

Attributes

- to : PortConjugation [1]

Operations

- originalPortDefinition () : PortDefinition [1] {redefines originalType, abstract}

7.4.2.2.20 PortDefinition_Init

Description

Initializes the properties of the SysML v2 element PortDefinition.

Generalizations

- Definition_Init (from SystemInitializers)

Attributes

- to : PortDefinition [1]

7.4.2.2.21 ReferenceUsage_Init

Description

Provides the basic features to map to a ReferenceUsage element.

Generalizations

- Usage_Init (from SystemInitializers)

Attributes

- to : ReferenceUsage [1]

7.4.2.2.22 RequirementUsage_Init

Description

Initializes the properties of the SysML v2 element RequirementUsage.

Generalizations

- Usage_Init (from SystemInitializers)

Attributes

- to : RequirementUsage [1]

7.4.2.2.23 StateUsage_Init

Description

Initializes the properties of the SysML v2 element StateUsage.

Generalizations

- ActionUsage_Init (from SystemInitializers)

Attributes

- to : StateUsage [1]

7.4.2.2.24 SubjectMembership_Init

Description

Initializes the properties of the SysML v2 element SubjectMembership.

Generalizations

- ParameterMembership_Init (from KerMLInitializers)

Attributes

- to : SubjectMembership [1]

7.4.2.2.25 Usage_Init

Description

Initializes the properties of the SysML v2 element Usage.

Generalizations

- Feature_Init (from KerMLInitializers)

Attributes

- to : Usage [1]

Operations

- isVariation () : Boolean [1]

false

7.5 Factories

7.5.1 Overview

The classes presented in this subclause specify facilities for creating elements in the target model from an arbitrary set of zero to many input parameters. After the target element is created, no link between it and an the value of inputs parameter (if any) will be preserved.

7.5.2 Mapping Specifications

7.5.2.1 LiteralString_Factory

Description

Factory class to create a LiteralString element.

Generalizations

- Expression_Init (from KerMLInitializers)
- Factory (from Foundations)

Association Ends

- string : String [1]
- to : LiteralString [1]
(redefines: Expression_Init::to)

Operations

- create (in string : String) : LiteralString [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

Set{ReturnParameterFeatureMembership_Factory.create() }

7.5.2.2 StringParameterFeature_Factory

Description

Factory class to create a feature element representing a string.

Generalizations

- Factory (from Foundations)
- Feature_Init (from KerMLInitializers)

Association Ends

- string : String [1]

Operations

- create (in string : String) : Feature [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set{StringParameterFeatureValue_Factory.create(string)}
```

7.5.2.3 StringParameterFeatureValue_Factory

Description

Factory class to create a string feature value relationship for a feature element.

Generalizations

- Factory (from Foundations)
- FeatureValue_Init (from KerMLInitializers)

Association Ends

- string : String [1]

Operations

- create (in string : String) : FeatureValue [1]
- value () : Expression [1] {redefines value}

```
LiteralString_Factory.create(string)
```

7.5.2.4 StringParameterMembership_Factory

Description

Factory class to create a parameter membership relationship for a feature element representing a string.

Generalizations

- Factory (from Foundations)
- ParameterMembership_Init (from KerMLInitializers)

Association Ends

- string : String [1]

Operations

- create (in string : String) : ParameterMembership [1]
- ownedMemberParameter () : Feature [1] {redefines ownedMemberParameter}

```
StringParameterFeature_Factory.create(string)
```

7.5.2.5 SubjectMembership_Factory

Description

Factory class to create a subject membership relationship for a given subject.

Generalizations

- Factory (from Foundations)
- SubjectMembership_Init (from SystemInitializers)

Association Ends

- subject : Type [1]

Operations

- create (in subject : Type) : SubjectMembership [1]
- ownedMemberParameter () : Feature [1] {redefines ownedMemberParameter}

subject

7.5.2.6 AssignmentActionUsage_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory to create an assignment action usage.

Generalizations

- AssignmentActionUsage_Init (from SystemInitializers)
- Factory (from Foundations)

Operations

- create () : AssignmentActionUsage [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

Set{AssignmentActionUsageParameterMembership_Factory.create(),
DirectedReferenceUsageParameterMembership_Factory.create(KerML::FeatureDirectionKind::_'in')}

7.5.2.7 AssignmentActionUsageFeatureMembership2_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class to create a feature membership relationship for a feature element created by the factory class AssignmentActionUsageTargetReferenceUsageIn2_Factory.

Generalizations

- Factory (from Foundations)
- FeatureMembership_Init (from KerMLInitializers)

Operations

- create () : FeatureMembership [1]
- ownedMemberFeature () : Feature [1] {redefines ownedMemberFeature}

```
AssignmentActionUsageTargetReferenceUsageIn2_Factory.create()
```

7.5.2.8 AssignmentActionUsageFeatureMembership3_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class to create a feature membership relationship for a feature element created by the factory class AssignmentActionUsageTargetReferenceUsageIn3_Factory.

Generalizations

- Factory (from Foundations)
- FeatureMembership_Init (from KerMLInitializers)

Operations

- create () : FeatureMembership [1]
- ownedMemberFeature () : Feature [1] {redefines ownedMemberFeature}

```
AssignmentActionUsageTargetReferenceUsageIn3_Factory.create()
```

7.5.2.9 AssignmentActionUsageOwningMembership_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class to create a owning membership relationship for an element created by the factory class AssignmentActionUsage_Factory.

Generalizations

- Factory (from Foundations)
- OwningMembership_Init (from KerMLInitializers)

Operations

- create () : OwningMembership [1]
- ownedMemberElement () : Element [1] {redefines ownedMemberElement}

```
AssignmentActionUsage_Factory.create()
```

7.5.2.10 AssignmentActionUsageParameterMembership_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class to create a parameter membership relationship for a feature element created by the factory class AssignmentActionUsageReferenceUsageIn1_Factory.

Generalizations

- Factory (from Foundations)
- ParameterMembership_Init (from KerMLInitializers)

Operations

- create () : ParameterMembership [1]
- ownedMemberParameter () : Feature [1] {redefines ownedMemberParameter}

```
AssignmentActionUsageReferenceUsageIn1_Factory.create()
```

7.5.2.11 AssignmentActionUsageReferenceUsageIn1_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class creating a reference usage element with direction "in" as parameter of an assignment action usage.

Generalizations

- Factory (from Foundations)
- ReferenceUsage_Init (from SystemInitializers)

Operations

- create () : ReferenceUsage [1]
- direction () : FeatureDirectionKind [0..1] {redefines direction}

```
KerML::FeatureDirectionKind::_'in'
```

- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set{AssignmentActionUsageFeatureMembership2_Factory.create() }
```

7.5.2.12 AssignmentActionUsageTargetReferenceUsageIn2_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class creating a reference usage element as an owned feature of the reference usage of an assignment action usage.

Generalizations

- Factory (from Foundations)
- ReferenceUsage_Init (from SystemInitializers)

Operations

- create () : ReferenceUsage [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set{AssignmentActionUsageFeatureMembership3_Factory.create() }
```

7.5.2.13 AssignmentActionUsageTargetReferenceUsageIn3_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class creating a reference usage element as an owned feature of the reference usage of an assignment action usage.

Generalizations

- Factory (from Foundations)
- ReferenceUsage_Init (from SystemInitializers)

Operations

- create () : ReferenceUsage [1]

7.5.2.14 DirectedReferenceUsage_Factory

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Factory class creating a reference usage element with a given direction and without owned relationships.

Generalizations

- Factory (from Foundations)
- ReferenceUsage_Init (from SystemInitializers)

Association Ends

- featureDirectionKind : FeatureDirectionKind [1]

Operations

- create (in featureDirectionKind : FeatureDirectionKind) : ReferenceUsage [1]
- direction () : FeatureDirectionKind [0..1] {redefines direction}

```
featureDirectionKind
```

7.5.2.15 DirectedReferenceUsageParameterMembership_Factory

[SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct](#)

Description

Factory class to create a parameter membership relationship for a feature element created by the factory class DirectedReferenceUsage_Factory.

Generalizations

- Factory (from Foundations)
- ParameterMembership_Init (from KerMLInitializers)

Association Ends

- featureDirectionKind : FeatureDirectionKind [1]

Operations

- create (in featureDirectionKind : FeatureDirectionKind) : ParameterMembership [1]
- ownedMemberParameter () : Feature [1] {redefines ownedMemberParameter}

DirectedReferenceUsage_Factory.create(featureDirectionKind)

7.5.2.16 EmptyObjectiveMembership_Factory

[SYSML2-240: TestCaseActivity_Mapping uses non-existing mapping classes](#)

Description

Factory class to create an objective membership without a source in the SysML v1 model.

Generalizations

- Factory (from Foundations)
- ObjectiveMembership_Init (from SystemInitializers)

Operations

- create () : ObjectiveMembership [1]
- ownedMemberFeature () : Feature [1] {redefines ownedMemberFeature}

EmptyRequirementUsage_Factory.create()

7.5.2.17 EmptyRequirementUsage_Factory

[SYSML2-240: TestCaseActivity_Mapping uses non-existing mapping classes](#)

Description

Factory class to create a requirement usage without a source in the SysML v1 model.

Generalizations

- Factory (from Foundations)
- RequirementUsage_Init (from SystemInitializers)

Operations

- create () : RequirementUsage [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set<EmptySubjectMembership> EmptySubjectMembership_Factory.create(),  
ReturnParameterFeatureMembership_Factory.create()
```

7.5.2.18 EmptySubject_Factory

Description

Factory class to create a reference usage representing a subject without a source in the SysML v1 model.

Generalizations

- Factory (from Foundations)
- ReferenceUsage_Init (from SystemInitializers)

Operations

- create () : ReferenceUsage [1]
- direction () : FeatureDirectionKind [0..1] {redefines direction}

```
KerML::FeatureDirectionKind::_in'
```

7.5.2.19 EmptySubjectMembership_Factory

Description

Factory class to create a membership relationship for a reference usage representing a subject without a source in the SysML v1 model.

Generalizations

- Factory (from Foundations)
- SubjectMembership_Init (from SystemInitializers)

Operations

- create () : SubjectMembership [1]
- ownedMemberParameter () : Feature [1] {redefines ownedMemberParameter}

```
EmptySubject_Factory.create()
```

7.5.2.20 FeatureTyping_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a FeatureTyping relationship. The create parameter is set as the type.

Generalizations

- Factory (from Foundations)
- FeatureTyping_Init (from KerMLInitializers)

Association Ends

- type : NamedElement [1]

Operations

- create (in type : NamedElement) : FeatureTyping [1]
- type () : Type [1] {redefines type}

type

7.5.2.21 FlowConnectionUsage_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a FlowConnectionUsage as a target element for a UML4SysML::InformationFlow that is realized by a UML4SysML::Connector. The factory class only supports UML4SysML::InformationFlows which have exactly one source and one target element, which is implicitly assured since connectors in SysML may only ever have two ends.

Generalizations

- Factory (from Foundations)
- FlowConnectionUsage_Init (from SystemInitializers)

Association Ends

- informationFlow : InformationFlow [1]

Operations

- create (in informationFlow : InformationFlow) : FlowConnectionUsage [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
let relationships : Set(KerML::Relationship) =
```

```

informationFlow.realizingConnector->collect(c|Subsetting_Factory.create(c))
->including(FeatureTyping_Factory.create(informationFlow))
->including(FlowEndParameterMembership_Factory.create(
    informationFlow,informationFlow.source.get(0)))
->including(FlowEndParameterMembership_Factory.create(
    informationFlow,informationFlow.target.get(0))) in
let itemProperty : UML::Property =
    if Helper.hasStereotypeApplied(informationFlow, 'SysML::Ports&Flows::ItemFlow') then
        Helper.getTagValueAsElement(informationFlow, 'SysML::Ports&Flows::ItemFlow', 'itemProp')
    else
        invalid
    endif in

if itemProperty.oclIsUndefined() then
    relationships->union(informationFlow.conveyed->flatten()
        ->collect(i | FlowItemFeatureMembership_Factory.create(i)))
else
    relationships->including(
        FlowItemFeatureMembership_Factory.create(itemProperty))
endif

```

7.5.2.22 FlowConnectionUsageFeatureMembership_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a FeatureMembership relationship for a FlowConnectionUsage as a target element for a UML4SysML::InformationFlow that is realized by a UML4SysML::Connector.

Generalizations

- Factory (from Foundations)
- FeatureMembership_Init (from KerMLInitializers)

Association Ends

- informationFlow : InformationFlow [1]

Operations

- create (in informationFlow : InformationFlow) : FeatureMembership [1]
- ownedMemberFeature () : Feature [1] {redefines ownedMemberFeature}

FlowConnectionUsage_Factory.create(informationFlow)

7.5.2.23 FlowEndParameterMembership_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a ParameterMembership relationship for an end of a FlowConnectionUsage as a target element for a UML4SysML::InformationFlow that is realized by a UML4SysML::Connector.

Generalizations

- Factory (from Foundations)
- ParameterMembership_Init (from KerMLInitializers)

Association Ends

- end : NamedElement [1]
- informationFlow : InformationFlow [1]

Operations

- create (in informationFlow : InformationFlow, in end : NamedElement) : ParameterMembership [1]
- ownedMemberParameter () : Feature [1] {redefines ownedMemberParameter}

```
InformationFlowEventOccurrenceUsage_Factory.create(informationFlow, end)
```

7.5.2.24 FlowItem_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a ItemFeature element as a target element for the flowing entity specified by an UML4SysML::InformationFlow.

Generalizations

- Factory (from Foundations)
- ItemFeature_Init (from SystemInitializers)

Association Ends

- item : NamedElement [1]

Operations

- create (in item : NamedElement) : ItemFeature [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
if item.oclIsKindOf(UML::Classifier) then
    Set{FeatureTyping_Factory.create(item)}
else if item.oclIsKindOf(UML::Property) then
    Set{ReferenceSubsetting_Factory.create(item)}
else
    Set{}
endif
endif
```

7.5.2.25 FlowItemFeatureMembership_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a FeatureMembership relationship for an ItemFeature as a target element for the flowing entity specified by an UML4SysML::InformationFlow.

Generalizations

- Factory (from Foundations)
- FeatureMembership_Init (from KerMLInitializers)

Association Ends

- item : NamedElement [1]

Operations

- create (in item : NamedElement) : FeatureMembership [1]
- ownedMemberFeature () : Feature [1] {redefines ownedMemberFeature}

```
FlowItem_Factory.create(item)
```

7.5.2.26 InformationFlowEventOccurrenceUsage_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Generalizations

- EventOccurrenceUsage_Init (from SystemInitializers)
- Factory (from Foundations)

Association Ends

- end : NamedElement [1]
- informationFlow : InformationFlow [1]

Operations

- create (in informationFlow : InformationFlow, in end : NamedElement) : EventOccurrenceUsage [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set{InformationFlowReferenceSubsetting_Factory.create(informationFlow, end)}
```

7.5.2.27 InformationFlowReferenceSubsetting_Factory

Description

Factory class to create a ReferenceSubsetting relationship for an end of a FlowConnectionUsage subsetting the target element of an end element of an UML4SysML::InformationFlow.

Generalizations

- Factory (from Foundations)
- ReferenceSubsetting_Init (from KerMLInitializers)

Association Ends

- end : NamedElement [1]
- informationFlow : InformationFlow [1]

Operations

- create (in informationFlow : InformationFlow, in end : NamedElement) : ReferenceSubsetting [1]
- referencedFeature () : Feature [1] {redefines referencedFeature}

```
InformationFlowEnd_Mapping.getMapped(informationFlow, end)
```

7.5.2.28 LiteralBoolean_Factory

Description

Factory class to create a LiteralBoolean element.

Generalizations

- Expression_Init (from KerMLInitializers)
- Factory (from Foundations)

Association Ends

- boolean : Boolean [1]
- to : LiteralBoolean [1]
(redefines: Expression_Init::to)

Operations

- create (in boolean : Boolean) : LiteralBoolean [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set{ReturnParameterFeatureMembership_Factory.create()}
```

7.5.2.29 LiteralNull_Factory

SYSML2-14: UML4SysML::ClearVariableAction transformation does not include a ReturnParameter

Description

Factory class to create a LiteralNull element.

Generalizations

- Expression_Init (from KerMLInitializers)
- Factory (from Foundations)

Association Ends

- to : NullExpression [1]
(redefines: Expression_Init::to)

Operations

- create () : NullExpression [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set{ReturnParameterFeatureMembership_Factory.create() }
```

7.5.2.30 LiteralRational_Factory

Description

Factory class to create a LiteralRational element.

Generalizations

- Expression_Init (from KerMLInitializers)
- Factory (from Foundations)

Association Ends

- real : Real [1]
- to : LiteralRational [1]
(redefines: Expression_Init::to)

Operations

- create (in real : Real) : LiteralReal [1]
- ownedRelationship () : Relationship [0..*] {redefines ownedRelationship}

```
Set{ReturnParameterFeatureMembership_Factory.create() }
```

7.5.2.31 ObjectFlowItemFlowEndRedefinition_Factory

Description

Generalizations

- Factory (from Foundations)
- Redefinition_Init (from KerMLInitializers)

Association Ends

- feature : Feature [1]

Operations

- create (in feature : Feature) : Redefinition [1]
- redefinedFeature () : Feature [1] {redefines redefinedFeature}

feature

7.5.2.32 ReferenceSubsetting_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a ReferenceSubseeting relationship. The create parameter is set as the referenced feature.

Generalizations

- Factory (from Foundations)
- ReferenceSubsetting_Init (from KerMLInitializers)

Association Ends

- property : Property [1]

Operations

- create (in property : Property) : ReferenceSubsetting [1]
- referencedFeature () : Feature [1] {redefines referencedFeature}

property

7.5.2.33 ReturnParameterFeature_Factory

Description

Factory class to create a feature element with direction 'out' representing a return parameter.

Generalizations

- Factory (from Foundations)
- Feature_Init (from KerMLInitializers)

Operations

- create () : Feature [1]
- direction () : FeatureDirectionKind [0..1] {redefines direction}

`KerML::FeatureDirectionKind::_'out'`

7.5.2.34 ReturnParameterFeatureMembership_Factory

Description

Factory class to create a feature membership relationship for a feature element with direction 'out' representing a return parameter.

Generalizations

- Factory (from Foundations)
- ReturnParameterMembership_Init (from KerMLInitializers)

Operations

- create () : ReturnParameterMembership [1]
- ownedMemberParameter () : Feature [1] {redefines ownedMemberParameter}

```
ReturnParameterFeature_Factory.create()
```

7.5.2.35 Subsetting_Factory

SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported

Description

Factory class to create a Subsetting relationship. The create parameter is set as the subsetted feature.

Generalizations

- Factory (from Foundations)
- Subsetting_Init (from KerMLInitializers)

Association Ends

- subsetted : NamedElement [1]

Operations

- create (in subsetted : NamedElement) : Subsetting [1]
- subsettedFeature () : Feature [1] {redefines subsettedFeature}

```
subsetted
```

7.6 Generic Mappings

7.6.1 Overview

Generic mappings are partial definitions of transformation rules that are intended to factorize reusable algorithms for making the global specification more compact and easier to read and maintain. Basically, they provide a default value for all the non-derived attributes of their target metaclass wherever possible, or declare an abstract operation for them otherwise. They are similar to initializers, except that they have a source element defined. The operations provided by the generic mappings can be redefined by their specialization, as appropriate according to the source type specified by the redefinition of their `from` attribute.

All of these generic mappings are abstract.

7.6.2 Common Mappings

7.6.2.1 CommonFeatureReferenceExpression_Mapping

Description

Common mapping class for a feature reference expression.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

TypedElement

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{CommonMembership_Mapping.getMapped(from),  
CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
```

7.6.2.2 CommonMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

TypedElement

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

from

7.6.2.3 CommonParameterReferenceUsageInMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

Element

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

```
if not from.oclIsKindOf(UML::TypedElement) then
    CommonParameterReferenceUsageIn_Mapping.getMapped(from)
else if from.oclAsType(UML::TypedElement).type.oclIsUndefined() then
    CommonParameterReferenceUsageIn_Mapping.getMapped(from)
else
    CommonParameterReferenceUsageInUntyped_Mapping.getMapped(from)
```

```
        endif  
    endif
```

7.6.2.4 CommonParameterReferenceUsageIn_Mapping

Description

Common mapping class that creates a parameter reference usage element with direction 'in' and with a type.

General Mappings

CommonParameterReferenceUsageInUntyped_Mapping

Mapping Source

Element

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
    if from.oclIsKindOf(UML::TypedElement) then  
        Set{CommonParameterReferenceUsageInFeatureTyping_Mapping.getMapped(from)}  
    else Set{} endif
```

7.6.2.5 CommonParameterReferenceUsageInFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

Element

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if from.oclIsKindOf(UML::TypedElement)
then
  if from.oclaType(UML::TypedElement).type.oclIsKindOf(UML::PrimitiveType) then
    Helper.getScalarValueType(from.oclaType(UML::TypedElement).type)
  else
    from.oclaType(UML::TypedElement).type
  endif
else invalid endif
```

7.6.2.6 CommonParameterReferenceUsageInUntyped_Mapping

Description

Common mapping class that creates a parameter reference usage element with direction 'in' and without a type.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

Element

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_ 'in'
```

7.6.2.7 CommonReturnParameterFeature_Mapping

Description

Common mapping class that creates a parameter feature element with a type.

General Mappings

CommonReturnParameterFeatureUntyped_Mapping

Mapping Source

Element

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]

```
if from.oclIsKindOf(UML::Property) then
    Set{CommonReturnParameterFeatureTyping_Mapping.getMapped(from)}
else
    Set{}
endif
```

7.6.2.8 CommonReturnParameterFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

Element

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if from.oclIsKindOf(UML::Property)
then
  if from.oclAsType(UML::TypedElement).type.oclIsKindOf(UML::PrimitiveType) then
    Helper.getScalarValueType(from.oclAsType(UML::TypedElement).type)
  else
    from.oclAsType(UML::TypedElement).type
  endif
else invalid endif
```

7.6.2.9 CommonReturnParameterFeatureUntyped_Mapping

Description

Common mapping class that creates a parameter feature element without a type.

General Mappings

GenericToFeature_Mapping

Mapping Source

Element

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_'out'
```

7.6.2.10 CommonReturnParameterFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToReturnParameterMembership_Mapping

Mapping Source

Element

Mapping Target

ReturnParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReturnParameterMembership::ownedMemberParameter () : Feature [1]

```
if not from.oclIsKindOf(UML::TypedElement) then
    CommonReturnParameterFeatureUntyped_Mapping.getMapped(from)
else if from.oclAsType(UML::TypedElement).type.oclIsUndefined() then
    CommonReturnParameterFeatureUntyped_Mapping.getMapped(from)
else
    CommonReturnParameterFeatureUntyped_Mapping.getMapped(from)
endif
endif
```

7.6.2.11 CommonReturnParameterReferenceUsageMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToReturnParameterMembership_Mapping

Mapping Source

Element

Mapping Target

ReturnParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReturnParameterMembership::ownedMemberParameter () : Feature [0..1]

```
if not from.oclIsKindOf(UML::TypedElement) then
    CommonReturnParameterReferenceUsageUntyped_Mapping.getMapped(from)
else if from.oclAsType(UML::TypedElement).type.oclIsUndefined() then
    CommonReturnParameterReferenceUsageUntyped_Mapping.getMapped(from)
else
    CommonReturnParameterReferenceUsageUntyped_Mapping.getMapped(from)
endif
endif
```

7.6.2.12 CommonReturnParameterReferenceUsage_Untyped_Mapping

Description

Creates a reference usage.

General Mappings

CommonReturnParameterReferenceUsageUntyped_Mapping

Mapping Source

Element

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
if from.oclIsKindOf(UMLO::TypedElement) then
Set{CommonReturnParameterReferenceUsageFeatureTyping_Mapping.getMapped(from)}
else Set{} endif
```

7.6.2.13 CommonReturnParameterReferenceUsageFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

Element

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if from.oclIsKindOf(UMLO::TypedElement)
then
if from.oclAsType(UMLO::TypedElement).type.oclIsKindOf(UMLO::PrimitiveType) then
    Helper.getScalarValueType(from.oclAsType(UMLO::TypedElement).type)
else
    from.oclAsType(UMLO::TypedElement).type
```

```
endif
else invalid endif
```

7.6.2.14 CommonReturnParameterReferenceUsageUntyped_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

Element

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_ 'out'
```

7.6.2.15 CommonReferenceUsageIn_Mapping

Description

Common mapping class that creates a reference usage element with direction 'in'.

General Mappings

CommonReferenceUsageInUntyped_Mapping

Mapping Source

TypedElement

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

Common mapping class that creates a reference usage element with direction 'in'.

```
Set{CommonReferenceUsageInFeatureTyping_Mapping.getMapped(from)}
```

7.6.2.16 CommonReferenceUsageInFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

TypedElement

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
if from.type.oclisUndefined() then
    CommonReferenceUsageInUntyped_Mapping.getMapped(from)
else
```

```
    CommonReferenceUsageIn_Mapping.getMapped(from)
endif
```

7.6.2.17 CommonReferenceUsageInFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

TypedElement

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if from.type.oclIsKindOf(UML::PrimitiveType) then
    Helper.getScalarValueType(from.type)
else
    from.type
endif
```

7.6.2.18 CommonReferenceUsageInUntyped_Mapping

Description

Common mapping class that creates an untyped reference usage element with direction 'in'.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

TypedElement

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]
 KerML::FeatureDirectionKind::_ 'in'
- ReferenceUsage::declaredName () : String [0..1]
 from.name

7.6.3 Generic Mappings To KerML

7.6.3.1 GenericToAnnotatingElement_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *AnnotatingElement*.

General Mappings

GenericToElement_Mapping

Mapping Source

Element

Mapping Target

AnnotatingElement

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- AnnotatingElement::annotation () : Annotation [0..*]

Set {}

7.6.3.2 GenericToAnnotation_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Annotation*.

General Mappings

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

Annotation

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Annotation::annotatedElement () : Element [1]
abstract rule
- Annotation::owningAnnotatedElement () : Element [0..1]
null
- Annotation::annotatingElement () : AnnotatingElement [1]
abstract rule

7.6.3.3 GenericToAssociation_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Association*.

General Mappings

GenericToRelationship_Mapping
GenericToClassifier_Mapping

Mapping Source

Element

Mapping Target

Association

Owned Mappings

(none)

7.6.3.4 GenericToBehavior_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Behavior*.

General Mappings

GenericToClassifier_Mapping

Mapping Source

Element

Mapping Target

Behavior

Owned Mappings

(none)

7.6.3.5 GenericToClassifier_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Classifier*.

General Mappings

GenericToType_Mapping

Mapping Source

Element

Mapping Target

Classifier

Owned Mappings

(none)

7.6.3.6 GenericToComment_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Comment*.

General Mappings

GenericToAnnotatingElement_Mapping

Mapping Source

Element

Mapping Target

Comment

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Comment::locale () : String [1]

 null

- Comment::body () : String [1]
 abstract rule

7.6.3.7 GenericToConjugation_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Conjugation*.

General Mappings

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

Conjugation

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Conjugation::conjugatedType () : Type [1]
abstract rule
- Conjugation::originalType () : Type [1]
abstract rule

7.6.3.8 GenericToConnector_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Connector*.

General Mappings

GenericToFeature_Mapping

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

Connector

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Connector::isDirected () : Boolean [1]

false

7.6.3.9 GenericToDocumentation_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Documentation*.

General Mappings

GenericToComment_Mapping

Mapping Source

Element

Mapping Target

Documentation

Owned Mappings

(none)

7.6.3.10 GenericToElement_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

This is the general abstract class to be used as an ancestor for any class mapping specification.

General Mappings

Mapping

Mapping Source

Element

Mapping Target

Element

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Element::ownedRelationship () : Relationship [0..*]

Set{ }

- Element::aliasId () : String [0..*]

Set{ }

- Element::shortName () : String [0..1]

null

- Element::declaredName () : String [0..1]

null

- Element::elementId () : String [1]

Helper.createUUID()

7.6.3.11 GenericToEndFeatureMembership_Mapping

[SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto](#)

Description

Generic mapping class for mappings to the SysML v2 element *EndFeatureMembership*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

Element

Mapping Target

EndFeatureMembership

Owned Mappings

(none)

7.6.3.12 GenericToExpression_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Expression*.

General Mappings

GenericToStep_Mapping

Mapping Source

Element

Mapping Target

Expression

Owned Mappings

(none)

7.6.3.13 GenericToFeature_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Feature*.

General Mappings

GenericToType_Mapping

Mapping Source

Element

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isComposite () : Boolean [1]
 false
- Feature::isOrdered () : Boolean [1]
 false
- Feature::isEnd () : Boolean [1]
 false
- Feature::isReadOnly () : Boolean [1]
 false
- Feature::direction () : FeatureDirectionKind [0..1]
 null
- Feature::isDerived () : Boolean [1]
 false
- Feature::isPortion () : Boolean [1]
 false
- Feature::isUnique () : Boolean [1]
 true

7.6.3.14 GenericToFeatureChainExpression_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *FeatureChainExpression*.

General Mappings

GenericToOperatorExpression_Mapping

Mapping Source

Element

Mapping Target

FeatureChainExpression

Owned Mappings

(none)

7.6.3.15 GenericToFeatureChaining_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *FeatureChaining*.

General Mappings

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

FeatureChaining

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]
abstract rule

7.6.3.16 GenericToFeatureMembership_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *FeatureMembership*.

General Mappings

GenericToOwningMembership_Mapping
GenericToTypeFeaturing_Mapping

Mapping Source

Element

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]
abstract rule
- FeatureMembership::ownedRelatedElement () : Element [0..*]

```
Set{self.ownedMemberFeature() }
```

7.6.3.17 GenericToFeatureReferenceExpression_Mapping

[SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto](#)

Description

Generic mapping class for mappings to the SysML v2 element *FeatureReferenceExpression*.

General Mappings

GenericToExpression_Mapping

Mapping Source

Element

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

7.6.3.18 GenericToFeatureTyping_Mapping

[SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto](#)

Description

Generic mapping class for mappings to the SysML v2 element *FeatureTyping*.

General Mappings

GenericToSpecialization_Mapping

Mapping Source

Element

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::typedFeature () : Feature [1]
abstract rule
- FeatureTyping::type () : Type [1]
abstract rule

7.6.3.19 GenericToFeatureValue_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *FeatureValue*.

General Mappings

GenericToOwningMembership_Mapping

Mapping Source

Element

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::featureWithValue () : Feature [1]
abstract rule
- FeatureValue::value () : Expression [1]
abstract rule
- FeatureValue::isDefault () : Boolean [1]

false
- FeatureValue::ownedRelatedElement () : Element [0..*]

Set{self.value ()}
- FeatureValue::isInitial () : Boolean [1]

false

7.6.3.20 GenericToFunction_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Function*.

General Mappings

GenericToBehavior_Mapping

Mapping Source

Element

Mapping Target

Function

Owned Mappings

(none)

7.6.3.21 GenericToImport_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Import*.

General Mappings

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

Import

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Import::isImportAll () : Boolean [1]
 false
- Import::isRecursive () : Boolean [1]
 false
- Import::importedMemberName () : String [0..1]
 null
- Import::visibility () : VisibilityKind [1]
 KerML::VisibilityKind::public

7.6.3.22 GenericToInvocationExpression_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *InvocationExpression*.

General Mappings

GenericToExpression_Mapping

Mapping Source

Element

Mapping Target

InvocationExpression

Owned Mappings

(none)

7.6.3.23 GenericToInteraction_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Interaction*.

General Mappings

GenericToBehavior_Mapping

GenericToAssociation_Mapping

Mapping Source

Element

Mapping Target

Interaction

Owned Mappings

(none)

7.6.3.24 GenericToItemFlow_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ItemFlow*.

General Mappings

GenericToConnector_Mapping

Mapping Source

Element

Mapping Target

ItemFlow

Owned Mappings

(none)

7.6.3.25 GenericToMembership_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Membership*.

General Mappings

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberShortName () : String [0..1]
 null
- Membership::membershipOwningNamespace () : Element [0..*]
 abstract rule
- Membership::visibility () : VisibilityKind [1]
 KerML::VisibilityKind::public
 - Membership::memberElement () : Element [1]
 abstract rule
 - Membership::memberName () : String [0..1]
 null

7.6.3.26 GenericToMembershipImport_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *MembershipImport*.

General Mappings

GenericToImport_Mapping

Mapping Source

Element

Mapping Target

MembershipImport

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MembershipImport::importedMembership () : Namespace [1]
abstract rule

7.6.3.27 GenericToNamespace_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Namespace*.

General Mappings

GenericToElement_Mapping

Mapping Source

Element

Mapping Target

Namespace

Owned Mappings

(none)

7.6.3.28 GenericToNamespacelImport_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *NamespaceImport*.

General Mappings

GenericToImport_Mapping

Mapping Source

Element

Mapping Target

NamespaceImport

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- NamespaceImport::importedNamespace () : Namespace [1]
abstract rule

7.6.3.29 GenericToOperatorExpression_Mapping

[SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto](#)

Description

Generic mapping class for mappings to the SysML v2 element *OperatorExpression*.

General Mappings

GenericToExpression_Mapping

Mapping Source

Element

Mapping Target

OperatorExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::operator () : String [1]
abstract rule

7.6.3.30 GenericToOwningMembership_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *OwningMembership*.

General Mappings

GenericToMembership_Mapping

Mapping Source

Element

Mapping Target

OwningMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]
abstract rule
- OwningMembership::ownedRelatedElement () : Element [0..*]

`Set{self.ownedMemberElement () }`

7.6.3.31 GenericToPackage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Package*.

General Mappings

GenericToNamespace_Mapping

Mapping Source

Element

Mapping Target

Package

Owned Mappings

(none)

7.6.3.32 GenericToParameterMembership_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ParameterMembership*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

Element

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedRelatedElement () : Element [0..*]

```
Set{self.ownedMemberParameter() }  
• ParameterMembership::ownedMemberParameter () : Feature [1]  
    null
```

7.6.3.33 GenericToPredicate_Mapping

[SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto](#)

Description

Generic mapping class for mappings to the SysML v2 element *Predicate*.

General Mappings

GenericToFunction_Mapping

Mapping Source

Element

Mapping Target

Predicate

Owned Mappings

(none)

7.6.3.34 GenericToRedefinition_Mapping

[SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto](#)

Description

Generic mapping class for mappings to the SysML v2 element *Redefinition*.

General Mappings

GenericToSubsetting_Mapping

Mapping Source

Element

Mapping Target

Redefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefiningFeature () : Feature [1]
abstract rule
- Redefinition::redefinedFeature () : Feature [1]
abstract rule

7.6.3.35 GenericToReferenceSubsetting_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ReferenceSubsetting*.

General Mappings

GenericToSubsetting_Mapping

Mapping Source

Element

Mapping Target

ReferenceSubsetting

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]
abstract rule

7.6.3.36 GenericToRelationship_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Relationship*.

General Mappings

GenericToElement_Mapping

Mapping Source

Element

Mapping Target

Relationship

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Relationship::target () : Element [0..*]
 Set {}
- Relationship::ownedRelatedElement () : Element [0..*]
 Set {}
- Relationship::source () : Element [0..*]
 Set {}

7.6.3.37 GenericToReturnParameterMembership_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ReturnParameterMembership*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

Element

Mapping Target

ReturnParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReturnParameterMembership::isComposite (in src : Element) : Boolean [1]

returns "true" if the element provided as the actual parameter value can have a mapping to an instance of the type specified by the "to" attribute (i.e. can be used as a value for the "from" attribute)

false

7.6.3.38 GenericToSpecialization_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Specialization*.

General Mappings

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

Specialization

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Specialization::general () : Type [1]
abstract rule

- Specialization::specific () : Type [1]
abstract rule

7.6.3.39 GenericToStep_Mapping

SYSML2-213: **Typo in section 7.6.3 and section 7.6.4: mappingsto**

Description

Generic mapping class for mappings to the SysML v2 element *Step*.

General Mappings

GenericToFeature_Mapping

Mapping Source

Element

Mapping Target

Step

Owned Mappings

(none)

7.6.3.40 GenericToSubclassification_Mapping

SYSML2-213: **Typo in section 7.6.3 and section 7.6.4: mappingsto**

Description

Generic mapping class for mappings to the SysML v2 element *Subclassification*.

General Mappings

GenericToSpecialization_Mapping

Mapping Source

Element

Mapping Target

Subclassification

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subclassification::subclassifer () : Classifier [1]
 null
- Subclassification::superclassifier () : Classifier [1]
 null

7.6.3.41 GenericToSubsetting_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Subsetting*.

General Mappings

GenericToSpecialization_Mapping

Mapping Source

Element

Mapping Target

Subsetting

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subsetting::ownedRelatedElement () : Element [0..*]
 Set { }
- Subsetting::subsettiedFeature () : Feature [1]
 abstract rule
- Subsetting::subsettingFeature () : Feature [1]
 from

7.6.3.42 GenericToSuccession_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Succession*.

General Mappings

GenericToConnector_Mapping

Mapping Source

Element

Mapping Target

Succession

Owned Mappings

(none)

7.6.3.43 GenericToSuccessionItemFlow_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *SuccessionItemFlow*.

General Mappings

GenericToSuccession_Mapping

GenericToItemFlow_Mapping

Mapping Source

Element

Mapping Target

SuccessionItemFlow

Owned Mappings

(none)

7.6.3.44 GenericToTextualRepresentation_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *TextualRepresentation*.

General Mappings

GenericToAnnotatingElement_Mapping

Mapping Source

Element

Mapping Target

TextualRepresentation

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TextualRepresentation::language () : String [1]
abstract rule
- TextualRepresentation::body () : String [1]
abstract rule

7.6.3.45 GenericToType_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Type*.

General Mappings

GenericToNamespace_Mapping

Mapping Source

Element

Mapping Target

Type

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Type::isAbstract () : Boolean [1]

false

- Type::isSufficient () : Boolean [1]

false

7.6.3.46 GenericToTypeFeaturing_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *TypeFeaturing*.

General Mappings

GenericToRelationship_Mapping

Mapping Source

Element

Mapping Target

TypeFeaturing

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TypeFeaturing::featuringType () : Type [1]
abstract rule
- TypeFeaturing::featureOfType () : Feature [1]
abstract rule

7.6.4 Generic Mappings to Systems

7.6.4.1 GenericToActionUsage_Mapping

SYSML2-213: **Typo in section 7.6.3 and section 7.6.4: mappingsto**

Description

Generic mapping class for mappings to the SysML v2 element *ActionUsage*.

General Mappings

GenericToUsage_Mapping

GenericToStep_Mapping

Mapping Source

Element

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::isComposite () : Boolean [1]

true

7.6.4.2 GenericToActorMembership_Mapping

SYSML2-213: **Typo in section 7.6.3 and section 7.6.4: mappingsto**

Description

Generic mapping class for mappings to the SysML v2 element *ActorMembership*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

Element

Mapping Target

ActorMembership

Owned Mappings

(none)

7.6.4.3 GenericToAssignmentActionUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *AssignmentActionUsage*.

General Mappings

GenericToActionUsage_Mapping

Mapping Source

Element

Mapping Target

AssignmentActionUsage

Owned Mappings

(none)

7.6.4.4 GenericToConnectionUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ConnectionUsage*.

General Mappings

GenericToPartUsage_Mapping

Mapping Source

Element

Mapping Target

ConnectionUsage

Owned Mappings

(none)

7.6.4.5 GenericToConjugatedPortDefinition_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ConjugatedPortDefinition*.

General Mappings

GenericToPortDefinition_Mapping

Mapping Source

Element

Mapping Target

ConjugatedPortDefinition

Owned Mappings

(none)

7.6.4.6 GenericToConjugatedPortTyping_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ConjugatedPortTyping*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

Element

Mapping Target

ConjugatedPortTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConjugatedPortTyping::conjugatedPortDefinition () : ConjugatedPortDefinition [1]
abstract rule
- ConjugatedPortTyping::portDefinition () : PortDefinition [1]
abstract rule

7.6.4.7 GenericToConstraintDefinition_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ConstraintDefinition*.

General Mappings

GenericToDefinition_Mapping

Mapping Source

Element

Mapping Target

ConstraintDefinition

Owned Mappings

(none)

7.6.4.8 GenericToConstraintUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ConstraintUsage*.

General Mappings

GenericToUsage_Mapping

Mapping Source

Element

Mapping Target

ConstraintUsage

Owned Mappings

(none)

7.6.4.9 GenericToDefinition_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *Definition*.

General Mappings

GenericToClassifier_Mapping

Mapping Source

Element

Mapping Target

Definition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Definition::isVariation () : Boolean [1]

false

7.6.4.10 GenericToEventOccurrenceUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *EventOccurrenceUsage*.

General Mappings

GenericToOccurrenceUsage_Mapping

Mapping Source

Element

Mapping Target

EventOccurrenceUsage

Owned Mappings

(none)

7.6.4.11 GenericToltemDefinition_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ItemDefinition*.

General Mappings

GenericToDefinition_Mapping

Mapping Source

Element

Mapping Target

ItemDefinition

Owned Mappings

(none)

7.6.4.12 GenericToltemUsage

**SYSML2-412: SYSML2-180 uses non-existing general mapping class
GenericToltemUsage_Mapping**

Description

Generic mapping class for mappings to the SysML v2 element ItemUsage.

General Mappings

GenericToOccurrenceUsage_Mapping

Mapping Source

Element

Mapping Target

ItemUsage

Owned Mappings

(none)

7.6.4.13 GenericToMetadataUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *MetadataUsage*.

General Mappings

GenericToUsage_Mapping

Mapping Source

Element

Mapping Target

MetadataUsage

Owned Mappings

(none)

7.6.4.14 GenericToObjectivemembership_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *ObjectiveMembership*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

Element

Mapping Target

ObjectiveMembership

Owned Mappings

(none)

7.6.4.15 GenericToOccurrenceDefinition_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *OccurrenceDefinition*.

General Mappings

GenericToDefinition_Mapping

Mapping Source

Element

Mapping Target

OccurrenceDefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OccurrenceDefinition::isIndividual () : Boolean [1]
 false

7.6.4.16 GenericToOccurrenceUsage_Mapping

[SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto](#)

Description

Generic mapping class for mappings to the SysML v2 element *OccurrenceUsage*.

General Mappings

GenericToUsage_Mapping

Mapping Source

Element

Mapping Target

OccurrenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OccurrenceUsage::isIndividual () : Boolean [1]
 false
- OccurrenceUsage::portionKind () : PortionKind [1]
 invalid

7.6.4.17 GenericToPartUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *PartUsage*.

General Mappings

GenericToUsage_Mapping

Mapping Source

Element

Mapping Target

PartUsage

Owned Mappings

(none)

7.6.4.18 GenericToPortConjugation_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *PortConjugation*.

General Mappings

GenericToConjugation_Mapping

Mapping Source

Element

Mapping Target

PortConjugation

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PortConjugation::originalPortDefinition () : PortDefinition [1]
abstract rule

7.6.4.19 GenericToPortDefinition_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *PortDefinition*.

General Mappings

GenericToDefinition_Mapping

Mapping Source

Element

Mapping Target

PortDefinition

Owned Mappings

(none)

7.6.4.20 GenericToReferenceUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Provides the basic features to map to a ReferenceUsage element.

General Mappings

GenericToUsage_Mapping

Mapping Source

Element

Mapping Target

ReferenceUsage

Owned Mappings

(none)

7.6.4.21 GenericToRequirementUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *RequirementUsage*.

General Mappings

GenericToUsage_Mapping

Mapping Source

Element

Mapping Target

RequirementUsage

Owned Mappings

(none)

7.6.4.22 GenericToStateUsage_Mapping

SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto

Description

Generic mapping class for mappings to the SysML v2 element *StateUsage*.

General Mappings

GenericToActionUsage_Mapping

Mapping Source

Element

Mapping Target

StateUsage

Owned Mappings

(none)

7.6.4.23 GenericToSubjectMembership_Mapping

[**SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto**](#)

Description

Generic mapping class for mappings to the SysML v2 element *SubjectMembership*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

Element

Mapping Target

SubjectMembership

Owned Mappings

(none)

7.6.4.24 GenericToTransitionUsage_Mapping

[**SYSML2-211: Introduce GenericToTransitionUsage_Mapping class**](#)

[**SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto**](#)

Description

Generic mapping class for mappings to the SysML v2 element *TransitionUsage*.

General Mappings

GenericToActionUsage_Mapping

Mapping Source

Element

Mapping Target

TransitionUsage

Owned Mappings

(none)

7.6.4.25 GenericToUsage_Mapping

[**SYSML2-213: Typo in section 7.6.3 and section 7.6.4: mappingsto**](#)

Description

Generic mapping class for mappings to the SysML v2 element *Usage*.

General Mappings

GenericToFeature_Mapping

Mapping Source

Element

Mapping Target

Usage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Usage::isVariation () : Boolean [1]

false

7.7 Mappings from UML4SysML metaclasses

7.7.1 Overview

UML4SysML is the subset of UML containing all model elements that are reused by SysML. The complete list of model elements is defined in [SysMLv1], subclause 4.1.

7.7.2 Actions

This chapter lists all mapping specifications of UML4SysML::Actions model elements.

7.7.2.1 Overview

SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters

SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column

Table 1. List of all mappings

SysML v1 Abstract Syntax/Stereotype	SysML v2 Abstract Syntax
AcceptCallAction	AcceptActionUsage

SysML v1 Abstract Syntax/Stereotype	SysML v2 Abstract Syntax
AcceptEventAction	AcceptActionUsage
ActionInputPin	ReferenceUsage
AddStructuralFeatureValueAction	ActionUsage
AddVariableValueAction	ActionUsage
BroadcastSignalAction	ActionUsage
CallBehaviorAction	ActionUsage
CallOperationAction	ActionUsage
Clause	not mapped; see next section
ClearAssociationAction	ActionUsage
ClearStructuralFeatureAction	ActionUsage
ClearVariableAction	ActionUsage
ConditionalNode	not mapped; see next section
CreateLinkAction	ActionUsage
CreateLinkObjectAction	ActionUsage
CreateObjectAction	ActionUsage
DestroyLinkAction	ActionUsage
DestroyObjectAction	ActionUsage
InputPin	not mapped; see next section
LinkEndCreationData	not mapped; see next section
LinkEndData	not mapped; see next section
LinkEndDestructionData	not mapped; see next section
LoopNode	ActionUsage
OpaqueAction	ActionUsage
OutputPin	ReferenceUsage
RaiseExceptionAction	ActionUsage
ReadExtentAction	ActionUsage
ReadIsClassifiedObjectAction	ActionUsage
ReadLinkAction	ActionUsage
ReadLinkObjectEndAction	ActionUsage
ReadSelfAction	ActionUsage
ReadStructuralFeatureAction	ActionUsage
ReadVariableAction	ActionUsage
ReclassifyObjectAction	ActionUsage

SysML v1 Abstract Syntax/Stereotype	SysML v2 Abstract Syntax
ReduceAction	ActionUsage
RemoveStructuralFeatureValueAction	ActionUsage
RemoveVariableValueAction	ActionUsage
ReplyAction	ActionUsage
SendObjectAction	ActionUsage
SendSignalAction	ActionUsage
SequenceNode	ActionUsage
StartClassifierBehaviorAction	ActionUsage
StartObjectBehaviorAction	ActionUsage
StructuredActivityNode	ActionUsage
TestIdentityAction	CalculationUsage
UnmarshallAction	ActionUsage
ValuePin	ReferenceUsage
ValueSpecificationAction	ActionUsage

The following table gives an overview of which SysML v2 elements the UML4SysML::Actions elements are transformed with which mapping class. The mapping details are in [7.7.2.3](#).

The justifications for the elements without mapping are given in [7.7.2.2](#).

7.7.2.2 UML4SysML::Actions elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

SYSML2-566: Section containing tables about elements not mapped should get an introductory text

Table 2. List of SysML v1 elements not mapped of this section

SysML v1 Concept	Rationale
AcceptCallAction	Since the CallEvent is not supported by SysML v2, the AcceptCallAction is also not covered. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.
ActionInputPin	The UML4SysML::ActionInputPin concept is not covered by SysML v2. The model element is mapped as a input or output pin, but without the special action input pin semantics.
Clause	Mapping is not specified yet.
ConditionalNode	Mapping is not specified yet.
LinkEndCreationData	Mapping is not specified yet.

SysML v1 Concept	Rationale
LinkEndData	Mapping is not specified yet.
LinkEndDestructionData	Mapping is not specified yet.
ReclassifyObjectAction	The UML4SysML::ReclassifyObjectAction is not supported by SysML v2. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.
ReplyAction	The UML4SysML::ReplyAction is only used with UML4SysML::AcceptCallAction. Since we have no mapping of AcceptCallAction to SysML v2, there is also no mapping for ReplyAction. However, it is mapped to an empty action usage to keep the connections within the activity respectively action definition.
StartClassifierBehaviorAction	The UML4SysML::StartClassifierBehaviorAction is not supported by SysML v2. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.
StartObjectBehaviorAction	The UML4SysML::StartObjectBehaviorAction is not supported by SysML v2. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.
UnmarshallAction	Mapping is not specified yet.

7.7.2.3 Mapping Specifications

7.7.2.3.1 Accept Event Actions

7.7.2.3.1.1 AcceptCallAction_Mapping

Description

Since the CallEvent is not supported by SysML v2, the AcceptCallAction is also not covered. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.

General Mappings

AcceptEventAction_Mapping

Mapping Source

AcceptCallAction

Mapping Target

AcceptActionUsage

Owned Mappings

(none)

7.7.2.3.1.2 AcceptEventAction_Mapping

SYSML2-246: AEAParameterMembership_Mapping::ownedMemberParameter cannot return OclUndefined

Description

The UML4SysML::AcceptEventAction is mapped to a AcceptActionUsage element.

If the trigger is a signal, it is mapped to an accept parameter typed by the signal.

SysMLv2 does not support more than one trigger. Therefore only the first specified trigger of the action is transformed. All further triggers are ignored.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action acceptEventActionSignalEvent1 accept : SysMLv1Signal via sysMLv1Port;
action acceptEventActionChangeEvent1 accept when when changeExpression.result {
    calc changeExpression {
        return : ScalarValues::Boolean;
        language "OCL"
        /*
         * x > 0
         */
    }
}
```

General Mappings

CommonAction_Mapping

Mapping Source

AcceptEventAction

Mapping Target

AcceptActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- AcceptActionUsage::ownedRelationship () : Relationship [0..*]

```

let relationships : Set(KerML::Relationship) = Helper.actionOwnedRelationship(from)
->including(AEAREceiverParameterMembership_Mapping.getMapped(from)) in
let relationshipsWithParameter : Set(KerML::Relationship) =
if (from.trigger.get(0).event.oclIsTypeOf(UML::SignalEvent) or
    from.trigger.get(0).event.oclIsTypeOf(UML::ChangeEvent)) then
    relationships->including(AEAParameterMembership_Mapping.getMapped(from))
else
    relationships
endif in
if from.trigger.get(0).event.oclIsTypeOf(UML::ChangeEvent) then
    relationshipsWithParameter
    ->including(ElementFeatureMembership_Mapping.getMapped(
        from.trigger.get(0).event.oclAsType(UML::ChangeEvent).changeExpression))
else relationshipsWithParameter
endif

```

7.7.2.3.1.3 AEAChangeExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]


```
from.trigger.get(0).event.oclAsType(UML::ChangeEvent).changeExpression
```

7.7.2.3.1.4 AEAChangeParameter_Mapping

Description

The mapping class transforms the change event specified at the AcceptEventAction.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

AcceptEventAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]
 KerML::FeatureDirectionKind::_ 'in'
- ReferenceUsage::ownedRelationship () : Relationship [0..*]
 Set{AEAChangeParameterFeatureValue_Mapping.getMapped(from) }

7.7.2.3.1.5 AEAChangeParameterFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
 AEAChangeParameterTrigger_Mapping.getMapped (from)

7.7.2.3.1.6 AEAChangeParameterTrigger_Mapping

Description

The mapping class creates a TriggerInvocationExpression from the change event specified at the AcceptEventAction.

General Mappings

GenericToInvocationExpression_Mapping

Mapping Source

AcceptEventAction

Mapping Target

TriggerInvocationExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TriggerInvocationExpression::ownedRelationship () : Relationship [0..*]
 Set {AEAChangeParameterFeatureMembership_Mapping.getMapped (from) }

7.7.2.3.1.7 AEAChangeParameterTriggerExpression_Mapping

Description

The mapping class creates the trigger expression element for the change parameter of the SysML v2 AcceptActionUsage element.

General Mappings

GenericToExpression_Mapping

Mapping Source

AcceptEventAction

Mapping Target

Expression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Expression::ownedRelationship () : Relationship [0..*]

```
Set{AEAChangeParameterResultExpressionMembership_Mapping.getMapped(from)}
```

7.7.2.3.1.8 AEAChangeParameterResultExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

AcceptEventAction

Mapping Target

ResultExpressionMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ResultExpressionMembership::ownedMemberFeature () : Feature [1]
AEAChangeParameterFeatureChainExpression_Mapping.getMapped(from)

7.7.2.3.1.9 AEAChangeParameterFeatureChainExpression_Mapping

Description

The mapping class creates the feature chain expression element for the change parameter of the SysML v2 AcceptActionUsage element.

General Mappings

GenericToInvocationExpression_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureChainExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChainExpression::ownedRelationship () : Relationship [0..*]
Set{AEAChangeParameterParameterMembership_Mapping.getMapped(from)}

7.7.2.3.1.10 AEAChangeParameterFeature_Mapping

Description

The mapping class creates the feature for the feature chain expression element for the change parameter of the SysML v2 AcceptActionUsage element.

General Mappings

GenericToFeature_Mapping

Mapping Source

AcceptEventAction

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]

```
Set{AEAChangeParameterExpressionFeatureValue_Mapping.getMapped(from)}
```

7.7.2.3.1.11 AEAChangeParameterExpressionFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
AEAChangeParameterFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.1.12 AEAChangeParameterFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression for the feature chain expression element for the change parameter of the SysML v2 AcceptActionUsage element.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{AEAChangeParameterMembership_Mapping.getMapped(from)}
```

7.7.2.3.1.13 AEAChangeParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

AcceptEventAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from.trigger.get(0).event.oclaType(UML::ChangeEvent).changeExpression
```

7.7.2.3.1.14 AEAChangeParameterParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

AcceptEventAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

```
AEAChangeParameterFeature_Mapping.getMapped(from)
```

7.7.2.3.1.15 AEAReceiverParameter_Mapping

Description

The mapping class creates the reference usage element for the receiver parameter of the SysML v2 AcceptActionUsage element.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

AcceptEventAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

KerML::FeatureDirectionKind::'_in'

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
if from.trigger.get(0).port->size() > 0
then Set{AEAReceiverFeatureValue_Mapping.getMapped(from)}
else Set{}
endif
```

7.7.2.3.1.16 AEAReceiverParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

AcceptEventAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

AEAResceiverParameter_Mapping.getMapped(from)

7.7.2.3.1.17 AEAResceiverFeatureValue_Mapping

[SYSML2-250: Typo in AEAResceiverFeatureValue_Mapping::value\(\)](#)

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

AEAResceiverFeatureReferenceExpression_Mapping.getMapped(from)

7.7.2.3.1.18 AEASignalParameter_Mapping

Description

The mapping class creates the reference usage element for the signal parameter of the SysML v2 AcceptActionUsage element.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

AcceptEventAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::'_in'
```
- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set{AEASignalParameterFeatureTyping_Mapping.getMapped(from)}
```

7.7.2.3.1.19 AEASignalParameterFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
let event : UML::Event = from.trigger.get(0).event in
if event.oclIsTypeOf(UML::SignalEvent) then
    event.oclAsType(UML::SignalEvent).signal
else invalid endif
```

7.7.2.3.1.20 AEAParameterMembership_Mapping

Description

The mapping class creates the parameter membership relationship for the element that can be received by the accept action. The source of the element is the trigger of the UML4SysML::AcceptEventAction.

Currently, more than one trigger is not supported by the transformation.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

AcceptEventAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

```

if from.trigger.get(0).event.oclIsTypeOf(UML::SignalEvent) then
    AEASignalParameter_Mapping.getMapped(from)
else if from.trigger.get(0).event.oclIsTypeOf(UML::ChangeEvent) then
    AEACHangeParameter_Mapping.getMapped(from)
else
    invalid
endif endif

```

7.7.2.3.1.21 AEAReserverFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression for the reference usage element for the receiver parameter of the SysML v2 AcceptActionUsage element.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

AcceptEventAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```

Set{AEAReserverFeatureReferenceExpressionMembership_Mapping.getMapped(from),
ReturnParameterFeatureMembership_Factory.create()}

```

7.7.2.3.1.22 AEAReserverFeatureReferenceExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

AcceptEventAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
if from.trigger.get(0).port->size() > 0 then
    from.trigger.get(0).port.get(0)
else
    invalid
endif
```

7.7.2.3.1.23 ReplyAction_Mapping

Description

The UML4SysML::ReplyAction is only used with UML4SysML::AcceptCallAction. Since we have no mapping of AcceptCallAction to SysML v2, there is also no mapping for ReplyAction. However, it is mapped to an empty action usage to keep the connections within the activity respectively action definition.

General Mappings

CommonAction_Mapping

Mapping Source

ReplyAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.1.24 UnmarshallAction_Mapping

Description

The mapping of UML4SysML::UnmarshallAction is not specified yet. It is currently mapped to an empty action usage to keep the connections within the activity respectively action definition.

General Mappings

CommonAction_Mapping

Mapping Source

UnmarshallAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.2 Actions

7.7.2.3.2.1 CommonAction_Mapping

Description

Base mapping class for model elements of kind UML4SysML::Action. The target element is a SysML v2 ActionUsage.

General Mappings

GenericToActionUsage_Mapping

NamedElementMain_Mapping

Mapping Source

Action

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
let actionInputPin: Set(UML::Element) =
    from.ownedElement->select(e | e.oclIsTypeOf(UML::ActionInputPin)) in
```

```

let triggers: Set(UML::Element) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::Trigger)) in
let toElementFMS: Set(UML::Element) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::Pin)) in
let toElementOMS: Set(UML::Element) =
    (((from.ownedElement - toElementFMS) - actionInputPin) - triggers) - from.ownedElement in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e))->asSet())

```

- ActionUsage::isComposite () : Boolean [1]

true

7.7.2.3.2.2 OpaqueAction_Mapping

Description

The UML4SysML::OpaqueAction is mapped to a SysML v2 ActionUsage with a textual representation.

The following shows an example of the expected SysMLv2 textual syntax of a UML4SysML::OpaqueAction.

```

action thisIsAOpaqueAction {
    in x : ScalarValues::Integer;
    in y : ScalarValues::Integer;
    out result : ScalarValues::Boolean;

    language "OCL"
    /*
     * x = y + 1;
     */
}

```

General Mappings

CommonAction_Mapping

Mapping Source

OpaqueAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
if from.body->size() > 0 then
  Helper.actionOwnedRelationship(from) ->append(OABodyMembership_Mapping.getMapped(from))
else
  Helper.actionOwnedRelationship(from)
endif
```

7.7.2.3.2.3 OABody_Mapping

Description

The languages and bodies of a UML4SysML::OpaqueAction are mapped to SysMLv2 TextualRepresentations.

General Mappings

GenericToAnnotatingElement_Mapping

Mapping Source

OpaqueAction

Mapping Target

TextualRepresentation

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TextualRepresentation::body () : String [1]

```
if from.body.notEmpty() then from.body.first() else invalid endif
```

- TextualRepresentation::language () : String [1]

```
if from.language.notEmpty() then from.language.first() else invalid endif
```

7.7.2.3.2.4 OABodyMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToOwningMembership_Mapping

Mapping Source

OpaqueAction

Mapping Target

OwningMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

OABody_Mapping.getMapped (from)

7.7.2.3.2.5 Pin_Mapping

[SYSML2-7: Pin_Mapping::filter: property src should be from](#)

[SYSML2-280: ElementMain_Mapping::ownedRelationship is wrong](#)

[SYSML2-278: UntypedPin_Mapping redefines operation without any changes](#)

[SYSML2-171: Optimize Pin mapping class generalization hierarchy](#)

[SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct](#)

Description

Mapping class for model elements of kind UML4SysML::Pin. The operation ownedRelationship() makes a distinction between typed and untyped pins. The target element is a SysMLv2 ReferenceUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    action sysMLv1Action {
        in sysMLv1InputPin : ScalarValues::Integer;
        out sysMLv1UntypedOutputPin;
    }
}
```

General Mappings

GenericToReferenceUsage_Mapping
NamedElementMain_Mapping

Mapping Source

Pin

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
not Helper.excludedPin(src)
```

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()  
->including(MultiplicityMembership_Mapping.getMapped(from))
```

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
if from.oclIsTypeOf(UML::InputPin) then  
    KerML::FeatureDirectionKind::_in'  
else if from.oclIsTypeOf(UML::OutputPin) then  
    KerML::FeatureDirectionKind::_out'  
else  
    invalid  
endif endif
```

7.7.2.3.2.6 ValuePin_Mapping

Description

A UML4SysML::ValuePin is mapped to a SysML v2 ReferenceUsage with assigned value.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action sysMLv1Action {  
    in sysMLv1ValuePin1 : ScalarValues::Integer = 42;  
    in sysMLv1ValuePin2 = {  
        return result;  
        language "English"  
    /*
```

```

        * this is a opaque expression
        */
    }.result;
}

```

General Mappings

No general mappings.

Mapping Source

ValuePin

Mapping Target

No target element.

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ownedRelationship () : Relationship [0..*]

```

Set<PinFeatureTyping_Mapping.getMapped(from),
ValuePinFeatureValue_Mapping.getMapped(from),
MultiplicityMembership_Mapping.getMapped(from) }

```

7.7.2.3.2.7 ValuePinFeatureValue_Mapping

Description

The mapping class creates the value expression for the reference usage element.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

ValuePin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
if from.value.oclIsUndefined() then invalid else from.value endif

7.7.2.3.2.8 ValuePinUntyped_Mapping

SYSML2-280: ElementMain_Mapping::ownedRelationship is wrong

Description

Same as ValuePin_Mapping, but for UML4SysML::ValuePins without a specified type.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action sysMLv1Action {
    in sysMLv1ValuePin1 = 42;
}
```

General Mappings

Pin_Mapping

Mapping Source

ValuePin

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
self.oclAsType(Pin_Mapping).ownedRelationship() -> including(ValuePinFeatureValue_Mapping.getValuePinFeatureValueMapping())
```

7.7.2.3.3 Invocation Actions

7.7.2.3.3.1 BroadcastSignalAction_Mapping

Description

The UML4SysML::BroadcastSignalAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

BroadcastSignalAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.3.2 CallBehaviorAction_Mapping

Description

A UML4SysML::CallBehaviorAction is mapped to a SysML v2 ActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity1 {
    action sysMLv1CallBehaviorAction : SysMLv1Activity2;
}
action def SysMLv1Activity2;
```

General Mappings

CommonAction_Mapping

Mapping Source

CallBehaviorAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
->append(CBAFeatureTyping_Mapping.getMapped(from))
```

7.7.2.3.3.3 CBAFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

CallBehaviorAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
from.behavior
```

7.7.2.3.3.4 CallOperationAction_Mapping

Description

A UML4SysML::CallOperationAction is mapped to a SysML v2 ActionUsage which calls the operation.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action sysMLv1CallOperationAction {
    in paramIn;
    in target : ThisIsABlock;
    out paramReturn = target.sysMLv1Operation;
}
```

General Mappings

CommonAction_Mapping

Mapping Source

CallOperationAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
->including(COAPerformActionFeatureMembership_Mapping.getMapped(from))
```

7.7.2.3.3.5 COAOutputPinFeature_Mapping

Description

The mapping class creates the feature element for the output parameter.

General Mappings

GenericToFeature_Mapping

Mapping Source

OutputPin

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]

```
Set{COAOutputPinFeatureFeatureValue_Mapping.getMapped(from),  
COAOutputPinFeatureFeatureMembership_Mapping.getMapped(from)}
```

- Feature::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_'in'
```

7.7.2.3.3.6 COAOutputPinFeatureChainExpression_Mapping

Description

The mapping class creates the feature chain expression for the output parameter feature value.

General Mappings

GenericToInvocationExpression_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureChainExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChainExpression::ownedRelationship () : Relationship [0..*]

```
Set{COAOutputPinParameterMembership_Mapping.getMapped(from),  
COAOutputPinFeatureChainExpressionMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()}
```

7.7.2.3.3.7 COAOutputPinFeatureChainExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

OutputPin

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from.owner.oclAsType(UML::CallOperationAction).operation
```

7.7.2.3.3.8 COAOutputPinFeatureFeature_Mapping

Description

Creates a feature element for the UML4SysML::CallOperationAction mapping.

General Mappings

GenericToFeature_Mapping

Mapping Source

OutputPin

Mapping Target

Feature

Owned Mappings

(none)

7.7.2.3.3.9 COAOutputPinFeatureFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]
COAOutputPinFeatureFeature_Mapping.getMapped (from)

7.7.2.3.3.10 COAOutputPinFeatureFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
COAOutputPinFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.3.11 COAOutputPinFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
COAOutputPinReferenceUsage_Mapping.getMapped(from)
```

7.7.2.3.3.12 COAOutputPinFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression for the output parameter.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{COAOutputPinFeatureReferenceExpressionMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()}
```

7.7.2.3.3.13 COAOutputPinFeatureReferenceExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

OutputPin

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from.owner.oclAsType(UML::CallOperationAction).target
```

7.7.2.3.3.14 COAOutputPinParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

OutputPin

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::visibility () : VisibilityKind [1]

```
KerML::VisibilityKind::private
```

- ParameterMembership::ownedMemberParameter () : Feature [1]

```
COAOutputPinFeature_Mapping.getMapped(from)
```

7.7.2.3.3.15 COAOutputPinReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

OutputPin

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set{COAOutputPinReferenceUsageFeatureValue_Mapping.getMapped(from)}
```

7.7.2.3.3.16 COAOutputPinReferenceUsageFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
 COAOutputPinFeatureChainExpression_Mapping.getMapped(from)

7.7.2.3.3.17 COAPerformAction_Mapping

Description

The mapping class creates the PerformActionUsage element.

General Mappings

GenericToActionUsage_Mapping

Mapping Source

CallOperationAction

Mapping Target

PerformActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PerformActionUsage::ownedRelationship () : Relationship [0..*]
 Set{COAPerformActionReferenceSubsetting_Mapping.getMapped(from)}

7.7.2.3.3.18 COAPerformActionFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToEndFeatureMembership_Mapping

Mapping Source

CallOperationAction

Mapping Target

EndFeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

COAPerformAction_Mapping.getMapped(from)

7.7.2.3.3.19 COAPerformActionReferenceSubsetting_Mapping**[SYSML2-200](#): Description of Subsetting mapping classes is not correct****Description**

Creates a subsetting relationship.

General Mappings

GenericToReferenceSubsetting_Mapping

Mapping Source

CallOperationAction

Mapping Target

ReferenceSubsetting

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::ownedRelatedElement () : Element [0..*]

```
Set{COAPerformActionFeature_Mapping.getMapped(from)}
```

7.7.2.3.3.20 COAPerformActionFeature_Mapping

Description

The mapping class creates the feature element for the perform action usage.

General Mappings

GenericToFeature_Mapping

Mapping Source

CallOperationAction

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]

```
Set{COAPerformActionFeatureChainingTarget_Mapping.getMapped(from),  
COAPerformActionFeatureChainingOperation_Mapping.getMapped(from)}
```

7.7.2.3.3.21 COAPerformActionFeatureChainingOperation_Mapping

Description

The mapping class creates the feature chaining element for the operation of the perform action usage.

General Mappings

GenericToFeatureChaining_Mapping

Mapping Source

CallOperationAction

Mapping Target

FeatureChaining

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]

from.operation

7.7.2.3.3.22 COAPerformActionFeatureChainingTarget_Mapping

Description

The mapping class creates the feature chaining element for the target element of the perform action usage.

General Mappings

GenericToFeatureChaining_Mapping

Mapping Source

CallOperationAction

Mapping Target

FeatureChaining

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]

from.target

7.7.2.3.3.23 SendObjectAction_Mapping

Description

A UML4SysML::SendObjectAction is mapped to a SysMLv2 ActionUsage that includes a SendActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action sysMLv1SendObjectAction {
    in target : SysMLv1Block;
    send SysMLv1Object1() to target;
}
part def SysMLv1Block;
item def SysMLv1Object;
```

General Mappings

SendSignalAction_Mapping

Mapping Source

SendObjectAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.3.24 SendSignalAction_Mapping

Description

A UML4SysML::SendSignalAction is mapped to a SysMLv2 ActionUsage that includes a SendActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action sysMLv1SendSignalAction {
    in target : SysMLv1Block;
    send SysMLv1Signal() to target;
}
part def SysMLv1Block;
item def SysMLv1Signal;
```

General Mappings

CommonAction_Mapping

Mapping Source

SendSignalAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
->including(SSAFeatureMembership_Mapping.getMapped(from))
```

7.7.2.3.3.25 SSAFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

InvocationAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
SSASendActionUsage_Mapping.getMapped(from)
```

7.7.2.3.3.26 SSAParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

InvocationAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

```
SSAReferenceUsage_Mapping.getMapped(from)
```

7.7.2.3.3.27 SSAReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

InvocationAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_ 'in'
```

7.7.2.3.3.28 SSAItemParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

InvocationAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

```
SSAItemReferenceUsage_Mapping.getMapped(from)
```

7.7.2.3.3.29 SSAItemReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

InvocationAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]
 KerML::FeatureDirectionKind::_ 'in'
- ReferenceUsage::ownedRelationship () : Relationship [0..*]
 Set{SSAItemReferenceUsageFeatureValue_Mapping.getMapped(from)}

7.7.2.3.3.30 SSAItemReferenceUsageFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

InvocationAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
SSAItemReferenceUsageInvocationExpression_Mapping.getMapped(from)
```

7.7.2.3.3.31 SSAItemReferenceUsageFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

InvocationAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if from.oclIsTypeOf(UML::SendSignalAction) then
    from.signal
else if from.oclIsTypeOf(UML::SendObjectAction) then
    from.request
else
    invalid
endif endif
```

7.7.2.3.3.32 SSAItemReferenceUsageInvocationExpression_Mapping

Description

The mapping class creates the invocation expression for the SysML v2 SendActionUsage.

General Mappings

GenericToInvocationExpression_Mapping

Mapping Source

InvocationAction

Mapping Target

InvocationExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- InvocationExpression::ownedRelationship () : Relationship [0..*]

```
Set<SSAIItemReferenceUsageFeatureTyping_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()>
```

7.7.2.3.3.33 SSATargetParameterMembership_Mapping**Description**

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

InvocationAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

```
SSATargetReferenceUsage_Mapping.getMapped(from)
```

7.7.2.3.3.34 SSATargetReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

InvocationAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]
 KerML::FeatureDirectionKind::'_in'
- ReferenceUsage::ownedRelationship () : Relationship [0..*]
 Set{SSATargetReferenceUsageFeatureValue_Mapping.getMapped(from)}

7.7.2.3.3.35 SSATargetReferenceUsageFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

InvocationAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
SSATargetReferenceUsageFeatureValueExpression_Mapping.getMapped(from)
```

7.7.2.3.3.36 SSATargetReferenceUsageFeatureValueMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

InvocationAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from.target
```

7.7.2.3.3.37 SSATargetReferenceUsageFeatureValueExpression_Mapping

Description

The mapping class creates the feature reference expression for the target reference usage element of the SysML v2 SendActionUsage.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

InvocationAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set<SSATargetReferenceUsageFeatureValueMembership_Mapping>.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()
```

7.7.2.3.3.38 SSASendActionUsage_Mapping

Description

The mapping class creates the SysML v2 element SendActionUsage for the UML4SysML::SendSignalAction mapping.

General Mappings

GenericToActionUsage_Mapping

Mapping Source

InvocationAction

Mapping Target

SendActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SendActionUsage::ownedRelationship () : Relationship [0..*]

```
Set{SSAItemParameterMembership_Mapping.getMapped(from),  
SSAParameterMembership_Mapping.getMapped(from),  
SSATargetParameterMembership_Mapping.getMapped(from) }
```

7.7.2.3.3.39 StartClassifierBehaviorAction_Mapping

Description

The UML4SysML::StartClassifierBehaviorAction is not supported by SysML v2. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.

General Mappings

CommonAction_Mapping

Mapping Source

StartClassifierBehaviorAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.3.40 StartObjectBehaviorAction_Mapping

Description

The UML4SysML::StartObjectBehaviorAction is not supported by SysML v2. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.

General Mappings

CommonAction_Mapping

Mapping Source

StartObjectBehaviorAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.4 Link Actions

7.7.2.3.4.1 ClearAssociationAction_Mapping

Description

The UML4SysML::ClearAssociationAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

ClearAssociationAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.4.2 CreateLinkAction_Mapping

Description

The UML4SysML::CreateLinkAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not completely defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

CreateLinkAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
let linkEndCreationData : Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsTypeOf(UML::LinkEndCreationData)) in
let actionInputPin: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsTypeOf(UML::ActionInputPin)) in
let triggers: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsKindOf(UML::Trigger)) in
let toElementFMS: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsKindOf(UML::Pin)) in
let toElementOMS: Set(UML::Element) =
    (((from.ownedElement - toElementFMS) - actionInputPin)
     - triggers) - linkEndCreationData) in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
```

7.7.2.3.4.3 CreateLinkObjectAction_Mapping

SYSML2-248: CreateLinkObjectAction_Mapping should specialize CreateLinkAction_Mapping

Description

A UML4SysML::CreateLinkObjectAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CreateLinkAction_Mapping

Mapping Source

CreateLinkObjectAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.4.4 DestroyLinkAction_Mapping

Description

The UML4SysML::DestroyLinkAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not completely defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

DestroyLinkAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
let actionInputPin: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsTypeOf(UML::ActionInputPin)) in
let triggers: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsKindOf(UML::Trigger)) in
let linkData: Set(UML::Element) =
    from.ownedElement->select( e | e.ocliIsKindOf(UML::LinkEndData) or
        e.ocliIsKindOf(UML::LinkEndDestructionData)) in
let toElementFMS: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsKindOf(UML::Pin)) in
let toElementOMS: Set(UML::Element) =
    (((from.ownedElement - toElementFMS) - actionInputPin)
        - triggers) - linkData) in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
```

7.7.2.3.4.5 ReadLinkAction_Mapping

Description

The UML4SysML::ReadLinkAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not completely defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

ReadLinkAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
let actionInputPin: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsTypeOf(UML::ActionInputPin)) in
let triggers: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsKindOf(UML::Trigger)) in
let linkData: Set(UML::Element) =
    from.ownedElement->select( e | e.ocliIsKindOf(UML::LinkEndData)) in
let toElementFMS: Set(UML::Element) =
    from.ownedElement->select(e | e.ocliIsKindOf(UML::Pin)) in
let toElementOMS: Set(UML::Element) =
    (((from.ownedElement - toElementFMS) - actionInputPin)
     - triggers) - linkData) in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
```

7.7.2.3.4.6 ReadLinkObjectEndAction_Mapping

Description

The UML4SysML::ReadLinkObjectEndAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

ReadLinkObjectEndAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.4.7 ReadLinkObjectEndQualifierAction_Mapping

Description

The UML4SysML::ReadLinkObjectEndQualifierAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

ReadLinkObjectEndQualifierAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.5 Object Actions

7.7.2.3.5.1 CreateObjectAction_Mapping

Description

A UML4SysML::CreateObjectAction is mapped to a SysML v2 ActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    action sysMLv1CreateObjectAction {
        out result : SysMLv1Block = SysMLv1Block();
    }
}
part def SysMLv1Block;
```

General Mappings

CommonAction_Mapping

Mapping Source

CreateObjectAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.5.2 COAInvocationExpressionFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

CreateObjectAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
from classifier
```

7.7.2.3.5.3 COAInvocationExpression_Mapping

Description

The mapping class creates the invocation expression to create the object.

General Mappings

GenericToInvocationExpression_Mapping

Mapping Source

CreateObjectAction

Mapping Target

InvocationExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- InvocationExpression::ownedRelationship () : Relationship [0..*]
Set{COAInvocationExressionFeatureTyping_Mapping.getMapped(from),
CommonReturnParameterFeatureMembership_Mapping.getMapped(from.result)}

7.7.2.3.5.4 COAPin_Mapping

[SYSML2-7: Pin_Mapping::filter: property src should be from](#)

Description

The mapping class creates the output parameter of the ActionUsage for the mapping of UML4SysML::CreateObjectAction.

General Mappings

No general mappings.

Mapping Source

OutputPin

Mapping Target

No target element.

Owned Mappings

(none)

Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.owner.oclIsTypeOf(UML::CreateObjectAction)
```

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ownedRelationship () : Relationship [0..*]
Set{PinFeatureTyping_Mapping.getMapped(from),
COAPinFeatureValue_Mapping.getMapped(from)}

7.7.2.3.5.5 COAPinFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
 COAInvocationExpression_Mapping.getMapped(from.owner)

7.7.2.3.5.6 DestroyObjectAction_Mapping

Description

The UML4SysML::DestroyObjectAction is conceptually mapped to the SysML v2 library function OccurrenceFunctions::destroy.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    action sysMLv1DestroyObjectAction {
        in target : SysMLv1Block;
        action : OccurrenceFunctions::destroy {
            in occ = target;
        }
    }
}
part def SysMLv1Block;
```

General Mappings

CommonAction_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
->including(DOADestroyFeatureMembership_Mapping.getMapped(from))
```

7.7.2.3.5.7 DOADestroyActionUsage_Mapping

Description

The mapping class creates the action usage for the destroy function.

General Mappings

GenericToActionUsage_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Set{DOADestroyActionUsageFeatureTyping_Mapping.getMapped(from),  
DOADestroyActionUsageFeatureMembership_Mapping.getMapped(from)}
```

7.7.2.3.5.8 DOADestroyActionUsageFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
DOADestroyActionUsageReferenceUsage_Mapping.getMapped(from)
```

7.7.2.3.5.9 DOADestroyActionUsageFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression for the UML4SysML::DestroyObjectAction mapping.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{DOADestroyActionUsageMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()}
```

7.7.2.3.5.10 DOADestroyActionUsageMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from.target
```

7.7.2.3.5.11 DOADestroyActionUsageFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SysMLv2::Function.allInstances()  
  ) ->any(e | e.qualifiedName = 'OccurrenceFunctions::destroy')
```

7.7.2.3.5.12 DOADestroyActionUsageFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
DOADestroyActionUsageFeatureReferenceExpression_Mapping.getMapped(from)

7.7.2.3.5.13 DOADestroyActionUsageReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]
Set{DOADestroyActionUsageFeatureValue_Mapping.getMapped(from)}

7.7.2.3.5.14 DOADestroyFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

DestroyObjectAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]
DOADestroyActionUsage_Mapping.getMapped (from)

7.7.2.3.5.15 ReadIsClassifiedObjectAction_Mapping

Description

The UML4SysML::ReadIsClassifiedObjectAction is conceptually mapped to a SysML v2 ActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    action sysMLv1ReadIsClassifiedObjectActionDirect {
        in object;
        out result : ScalarValues::Boolean =
            object istype ThisIsABlock;
    }

    action sysMLv1ReadIsClassifiedObjectActionNonDirect {
        in object;
        out result : ScalarValues::Boolean =
            object hastype ThisIsABlock;
    }
}
```

General Mappings

CommonAction_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.5.16 RICOAFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
RICOAFeatureValueOperatorExpression_Mapping.getMapped(from)

7.7.2.3.5.17 RICOAFeatureValueOperatorExpression_Mapping

Description

The mapping class creates the operator expression for the UML4SysML::ReadIsClassifiedObjectAction mapping.

General Mappings

GenericToOperatorExpression_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

OperatorExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::ownedRelationship () : Relationship [0..*]
Set{RICOAFeatureValueOperatorParameterMembership_Mapping.getMapped(from)}
- OperatorExpression::operator () : String [1]
if from.isDirect then 'istype' else 'hasType' endif

7.7.2.3.5.18 RICOAFeatureValueOperatorExpressionFeature_Mapping

Description

The mapping class creates the feature for the operator expression of the UML4SysML::ReadIsClassifiedObjectAction mapping.

General Mappings

GenericToFeature_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]
Set{RICOAFeatureValueOperatorExpressionFeatureValue_Mapping.getMapped(from)}

- Feature::direction () : FeatureDirectionKind [0..1]

KerML::FeatureDirectionKind::_'in'

7.7.2.3.5.19 RICOAFeatureValueOperatorExpressionFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

RICOAFeatureValueOperatorFeatureReferenceExpression_Mapping.getMapped(from)

7.7.2.3.5.20 RICOAFeatureValueOperatorFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression for the UML4SysML::ReadIsClassifiedObjectAction mapping.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{RICOAFeatureValueOperatorMembership_Mapping.getMapped(from),  
CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
```

7.7.2.3.5.21 RICOAFeatureValueOperatorMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

Membership

Owned Mappings

(none)

7.7.2.3.5.22 RICOAFeatureValueOperatorParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

ReadIsClassifiedObjectAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]
RICOAFeatureValueOperatorExpressionFeature_Mapping.getMapped(from)
- ParameterMembership::visibility () : VisibilityKind [1]
KerML::VisibilityKind::private

7.7.2.3.5.23 RICOAOutputPin_Mapping

SYSML2-7: Pin_Mapping::filter: property src should be from

Description

The mapping class creates the output parameter of the ActionUsage element for the UML4SysML::ReadIsClassifiedObjectAction mapping.

General Mappings

No general mappings.

Mapping Source

OutputPin

Mapping Target

No target element.

Owned Mappings

(none)

Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.owner.oclIsTypeOf(UML::ReadIsClassifiedObjectAction)
```

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ownedRelationship () : Relationship [0..*]

```
Set{PinFeatureTyping_Mapping.getMapped(from),  
RICOAFeatureValue_Mapping.getMapped(from.owner),  
MultiplicityMembership_Mapping.getMapped(from)}
```

7.7.2.3.5.24 ReadExtentAction_Mapping

Description

A UML4SysML::ReadExtentAction is mapped to a SysML v2 ActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {  
    action sysMLv1ReadExtentAction {  
        out thisIsTheOutputPin : SysMLv1Block =  
            all SysMLv1Block;  
    }  
}  
part def SysMLv1Block;
```

General Mappings

CommonAction_Mapping

Mapping Source

ReadExtentAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
```

7.7.2.3.5.25 REAFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
REAFeatureValueOperatorExpression_Mapping.getMapped(from)
```

7.7.2.3.5.26 REAFeatureValueOperatorExpression_Mapping

Description

The mapping class creates the operator expression for the UML4SysML::ReadExtentAction mapping.

General Mappings

GenericToOperatorExpression_Mapping

Mapping Source

OutputPin

Mapping Target

OperatorExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::operator () : String [1]

'all'
- OperatorExpression::ownedRelationship () : Relationship [0..*]

Set{REAFEATUREVALUEOPERATOREXPRESSIONMEMBERSHIP_MAPPING.getMAPPED(from),
COMMONRETURNPARAMETERFEATUREMEMBERSHIP_MAPPING.getMAPPED(from)}

7.7.2.3.5.27 REAFeatureValueOperatorExpressionFeature_Mapping

Description

The mapping class creates the feature for the operator expression for the UML4SysML::ReadExtentAction mapping.

General Mappings

GenericToFeature_Mapping

Mapping Source

OutputPin

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]

Set{REAFEATUREVALUEOPERATOREXPRESSIONFEATURETYPING_MAPPING.getMAPPED(from)}

7.7.2.3.5.28 REAFeatureValueOperatorExpressionFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

from.owner.classifier

7.7.2.3.5.29 REAFeatureValueOperatorExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]
REAFeatureValueOperatorExpressionFeature_Mapping.getMapped(from)

7.7.2.3.5.30 REAOutputPin_Mapping

SYSML2-19: REAOutputPin_Mapping should specialize OutputPin_Mapping

SYSML2-7: Pin_Mapping::filter: property src should be from

SYSML2-280: ElementMain_Mapping::ownedRelationship is wrong

SYSML2-171: Optimize Pin mapping class generalization hierarchy

Description

The mapping class creates the output parameter of the ActionUsage for the mapping of UML4SysML::ReadExtentAction.

General Mappings

Pin_Mapping

Mapping Source

OutputPin

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.owner.oclIsTypeOf(UML::ReadExtentAction)
```

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set {TypedElementFeatureTyping_Mapping.getMapped(from),  
REAFeatureValue_Mapping.getMapped(from)}  
->union(self.oclAsType(Pin_Mapping).ownedRelationship())
```

7.7.2.3.5.31 ReadSelfAction_Mapping

Description

A UML4SysML::ReadSelfAction is mapped to a SysML v2 ActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    action sysMLv1ReadSelfAction {
        out : Base::Anything = this;
    }
}
```

General Mappings

CommonAction_Mapping

Mapping Source

ReadSelfAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.5.32 RSAFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
RSAFeatureValueFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.5.33 RSAFeatureValueFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression for the mapping of UML4SysML::ReadSelfAction.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{RSAFeatureValueMembership_Mapping.getMapped(from),  
CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
```

7.7.2.3.5.34 RSAFeatureValueMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

OutputPin

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
SYSML2::Feature.allInstances()  
->any(e | e.qualifiedName = 'Occurrences::Occurrence::this')
```

7.7.2.3.5.35 RSAOutputPin_Mapping

SYSML2-7: Pin_Mapping::filter: property src should be from
SYSML2-280: ElementMain_Mapping::ownedRelationship is wrong
SYSML2-171: Optimize Pin mapping class generalization hierarchy

Description

The mapping class creates the output parameter of the ActionUsage for the mapping of UML4SysML::ReadSelfAction.

General Mappings

Pin_Mapping

Mapping Source

OutputPin

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.owner.oclIsKindOf(UML::ReadSelfAction)
```

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::isUnique () : Boolean [1]
 false
- ReferenceUsage::isAbstract () : Boolean [1]
 true
- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set<TypedElementFeatureTyping_Mapping.getMapped(from),  
RSAFeatureValue_Mapping.getMapped(from)}  
->union(self.oclAsType(Pin_Mapping).ownedRelationship())
```

7.7.2.3.5.36 ReclassifyObjectAction_Mapping

Description

The UML4SysML::ReclassifyObjectAction is not supported by SysML v2. It is mapped to an empty action usage to keep the connections within the activity respectively action definition.

General Mappings

CommonAction_Mapping

Mapping Source

ReclassifyObjectAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.5.37 TestIdentityAction_Mapping

Description

A UML4SysML::TestIdentityAction is mapped to a SysML v2 ActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {  
    action sysMLv1TestIdentityAction {  
        in firstParameter;  
        in secondParameter;  
        out result : ScalarValues::Boolean =  
            firstParameter == secondParameter;
```

```
    }  
}
```

General Mappings

CommonAction_Mapping

Mapping Source

TestIdentityAction

Mapping Target

CalculationUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- CalculationUsage::ownedRelationship () : Relationship [0..*]

```
    Helper.actionOwnedRelationship(from)  
    ->including(TIAResultExpressionMembership_Mapping.getMapped(from))
```

7.7.2.3.5.38 TIAOperatorExpression_Mapping

SYSML2-232: TIAOperatorExpression_Mapping uses non-existing mapping class EqualOperatorExpressionOperand_Mapping

Description

The mapping class creates the operator expression for the UML4SysML::TestIdentityAction mapping.

General Mappings

GenericToOperatorExpression_Mapping

Mapping Source

TestIdentityAction

Mapping Target

OperatorExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::operator () : String [1]

' == '

- OperatorExpression::ownedRelationship () : Relationship [0..*]

```
Set{EqualOperatorExpressionOperandParameterMembership_Mapping.getMapped(from.first),  
EqualOperatorExpressionOperandParameterMembership_Mapping.getMapped(from.second),  
CommonReturnParameterFeatureMembership_Mapping.getMapped(from.result)}
```

7.7.2.3.5.39 TIAResultExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

TestIdentityAction

Mapping Target

ResultExpressionMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ResultExpressionMembership::ownedMemberFeature () : Feature [0..1]

TIAOperatorExpression_Mapping.getMapped(from)

7.7.2.3.5.40 ValueSpecificationAction_Mapping

Description

A UML4SysML::ValueSpecificationAction is mapped to a SysML v2 ActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Acticity {
    action sysMLv1ValueSpecificationAction1 {
        out result : ScalarValues::Integer = 42;
    }

    action sysMLv1ValueSpecificationAction2 {
        out result = sysMLv1OpaqueExpression.result;
        calc sysMLv1OpaqueExpression {
            language "Math"
            /*
             * 42 + 23
            */
        }
    }
}
```

General Mappings

CommonAction_Mapping

Mapping Source

ValueSpecificationAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
let toElementFMS: Set(UML::Element) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::Pin)) in
let toElementOMS: Set(UML::Element) =
    (from.ownedElement - toElementFMS) - Set{from.value} in
toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e))
->union(toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e)))
```

7.7.2.3.5.41 VSAOutputPin_Mapping

SYSML2-7: Pin_Mapping::filter: property src should be from

SYSML2-280: ElementMain_Mapping::ownedRelationship is wrong

SYSML2-171: Optimize Pin mapping class generalization hierarchy

Description

The mapping class creates the output parameter of the ActionUsage for the mapping of UML4SysML::ValueSpecificationAction.

General Mappings

Pin_Mapping

Mapping Source

OutputPin

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.owner.oclIsKindOf(UML::ValueSpecificationAction)
```

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
let relationships : Set(KerML::Relationship) = self.oclAsType(Pin_Mapping).ownedRelationship  
->including(VSAOutputPinFeatureValue_Mapping.getMapped(from)) in  
if from.type.oclIsUndefined() then  
relationships  
else  
relationships->including(TypedElementFeatureTyping_Mapping.getMapped(from))  
endif
```

7.7.2.3.5.42 VSAOutputPinFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

OutputPin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
if from.owner.valueoclIsTypeOf(UML::OpaqueExpression) then
    OpaqueExpressionAsValue_Mapping.getMapped(from.owner.value)
else
    from.owner.value
endif
```

7.7.2.3.6 Other Actions

7.7.2.3.6.1 RaiseExceptionAction_Mapping

Description

The UML4SysML::RaiseExceptionAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

RaiseExceptionAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.6.2 ReduceAction_Mapping

Description

The UML4SysML::ReduceAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

ReduceAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.7 Structural Feature Actions

7.7.2.3.7.1 AddStructuralFeatureValueAction_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

A UML4SysML::AddStructuralFeatureValueAction is mapped to a SysML v2 ActionUsage defined by the SysML v1 library action definition SysMLv1Library::AddStructuralFeatureValueAction.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action thisIsAAddStructuralFeatureValueAction : SysMLv1Library::AddStructuralFeatureValueAction {
    :>> target := object.thisIsAnAttribute;
    :>> object : ThisIsABlock;
}
part def SysMLv1Block {
    attribute sysMLv1Property;
}
```

General Mappings

CommonAction_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Set{ASFVAFeatureTyping_Mapping.getMapped(from),  
ASFVATargetFeatureMembership_Mapping.getMapped(from),  
ASFVAObjectFeatureMembership_Mapping.getMapped(from)}
```

7.7.2.3.7.2 ASFVAFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYSML2::ActionDefinition.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::AddStructuralFeatureValueAction')
```

7.7.2.3.7.3 ASFVAObjectFeatureMembership_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

ASFVAObjectReferenceUsage_Mapping.getMapped (from)

7.7.2.3.7.4 ASFVAObjectReferenceUsage_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

Creates a reference usage.

General Mappings

UniqueMapping

GenericToReferenceUsage_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set{ASFVAObjectReferenceUsageRedefinition_Mapping.getMapped(from),  
ASFVAObjectReferenceUsageFeatureTyping_Mapping.getMapped(from)}
```

7.7.2.3.7.5 ASFVAObjectReferenceUsageFeatureTyping_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
from.structuralFeature.owner
```

7.7.2.3.7.6 ASFVAObjectReferenceUsageRedefinition_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

General Mappings

GenericToRedefinition_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

Redefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::ReferenceUsage.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::AddStructuralFeatureValueAction::object')
```

7.7.2.3.7.7 ASFVATargetFeatureChainExpression_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

The mapping class creates the feature chain expression element for the target element of the UML4SysML::AddStructuralFeatureValueAction mapping.

General Mappings

GenericToFeatureChainExpression_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureChainExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChainExpression::ownedRelationship () : Relationship [0..*]

```
Set<ASFVATargetParameterMembership_Mapping.getMapped(from),  
ASFVATargetParameterFeatureExpressionMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create() }
```

7.7.2.3.7.8 ASFVATargetFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ASFVATargetReferenceUsage_Mapping.getMapped(from)
```

7.7.2.3.7.9 ASFVATargetFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
ASFVATargetFeatureChainExpression_Mapping.getMapped(from)
- FeatureValue::isInitial () : Boolean [1]
true

7.7.2.3.7.10 ASFVATargetParameterExpressionFeature_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

The mapping class creates the feature element of the feature reference expression for the target element of the UML4SysML::AddStructuralFeatureValueAction mapping.

General Mappings

GenericToFeature_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

Feature

Owned Mappings

(none)

7.7.2.3.7.11 ASFVATargetParameterExpressionFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ASFVATargetParameterExpressionFeature_Mapping.getMapped(from)
```

7.7.2.3.7.12 ASFVATargetParameterExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]
ASFVAObjectReferenceUsage_Mapping.getMapped (from)

7.7.2.3.7.13 ASFVATargetParameterFeature_Mapping

Description

The mapping class creates the feature element for the target element of the UML4SysML::AddStructuralFeatureValueAction mapping.

General Mappings

GenericToFeature_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]

```
Set{ASFVATargetParameterFeatureValue_Mapping.getMapped(from),  
ASFVATargetParameterExpressionFeatureMembership_Mapping.getMapped(from)}
```

- Feature::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_'in'
```

7.7.2.3.7.14 ASFVATargetParameterFeatureExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from.structuralFeature
```

7.7.2.3.7.15 ASFVATargetParameterFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression element for the target element of the UML4SysML::AddStructuralFeatureValueAction mapping.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{ASFVATargetParameterExpressionMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()}
```

7.7.2.3.7.16 ASFVATargetParameterFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
ASFVATargetParameterFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.7.17 ASFVATargetParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::visibility () : VisibilityKind [1]
 KerML::VisibilityKind::private
- ParameterMembership::ownedMemberParameter () : Feature [1]
 ASFVATargetParameterFeature_Mapping.getMapped(from)

7.7.2.3.7.18 ASFVATargetReferenceUsage_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set<ASFVATargetReferenceUsageRedefinition_Mapping> getMapped(from),  
ASFVATargetFeatureValue_Mapping getMapped(from),  
AssignmentActionUsageOwningMembership_Factory.create()
```

7.7.2.3.7.19 ASFVATargetReferenceUsageRedefinition_Mapping

SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct

Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

General Mappings

GenericToRedefinition_Mapping

Mapping Source

AddStructuralFeatureValueAction

Mapping Target

Redefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::ReferenceUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::AddValueAction::target')
```

7.7.2.3.7.20 ClearStructuralFeatureAction_Mapping

Description

The UML4SysML::ClearStructuralFeatureAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

ClearStructuralFeatureAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.7.21 ReadStructuralFeatureAction_Mapping

Description

A UML4SysML::ReadStructuralFeatureAction is mapped to a SysML v2 ActionUsage that returns the value of the specified structural feature of the given object.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    action sysMLv1ReadStructuralFeatureAction {
        in object : SysMLv1Block;
        out result = object.sysMLv1Property;
    }
}
part def SysMLv1Block {
    attribute sysMLv1Property;
}
```

General Mappings

CommonAction_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
->including(RSFAReferenceUsageFeatureMembership_Mapping.getMapped(from))
```

7.7.2.3.7.22 RSFAReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_'out'
```

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set{RSFAResourceUsageFeatureValue_Mapping.getMapped(from)}
```

7.7.2.3.7.23 RSFAResourceUsageExpressionFeature_Mapping

Description

The mapping class creates the feature of the feature chain expression for the reference usage of the UML4SysML::ReadStructuralFeatureValueAction mapping.

General Mappings

GenericToFeature_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..*]

```
Set{RSFAResourceUsageExpressionFeatureValue_Mapping.getMapped(from),  
RSFAResourceUsageFeatureMembership_Mapping.getMapped(from)}
```

7.7.2.3.7.24 RSFAResourceUsageExpressionFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
RSFAResourceUsageFeatureChainExpressionFeature_Mapping.getMapped(from)
```

7.7.2.3.7.25 RSFAResourceUsageExpressionFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression element for the UML4SysML::RemoveStructuralFeatureValueAction mapping.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{RSFAResourceUsageExpressionFeatureMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()}
```

7.7.2.3.7.26 RSFAResourceUsageExpressionFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
RSFAResourceUsageExpressionFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.7.27 RSFAResourceUsageFeatureChainExpression_Mapping

Description

The mapping class creates the feature chain expression element for the reference usage of the UML4SysML::ReadStructuralFeatureValueAction mapping.

General Mappings

GenericToFeatureChainExpression_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

FeatureChainExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChainExpression::ownedRelationship () : Relationship [0..*]
Set{RSFAResourceUsageParameterMembership_Mapping.getMapped(from),
RSFAResourceUsageMembership_Mapping.getMapped(from),
ReturnParameterFeatureMembership_Factory.create()}

7.7.2.3.7.28 RSFAResourceUsageFeatureChainExpressionFeature_Mapping

Description

The mapping class creates the feature element for the feature chain expression for the UML4SysML::RemoveStructuralFeatureValueAction mapping.

General Mappings

GenericToFeature_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

Feature

Owned Mappings

(none)

7.7.2.3.7.29 RSFAResourceUsageFeatureChainExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

from.structuralFeature

7.7.2.3.7.30 RSFAResourceUsageFeatureMembership_Mapping

SYSML2-234: RSFAResourceUsageFeatureMembership_Mapping uses non-existing mapping class

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

RSFAResourceUsageFeatureValue_Mapping.getMapped(from)

7.7.2.3.7.31 RSFAResourceUsageFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
 RSFAResourceUsageFeatureChainExpression_Mapping.getMapped(from)

7.7.2.3.7.32 RSFAResourceUsageMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]
 from.object

7.7.2.3.7.33 RSFAResourceUsageParameterMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToParameterMembership_Mapping

Mapping Source

ReadStructuralFeatureAction

Mapping Target

ParameterMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]
 RSFAResourceUsageExpressionFeature_Mapping.getMapped(from)

7.7.2.3.7.34 RemoveStructuralFeatureValueAction_Mapping

Description

The UML4SysML::RemoveStructuralFeatureValueAction is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

RemoveStructuralFeatureValueAction

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.8 Structured Actions

7.7.2.3.8.1 LoopNode_Mapping

Description

The UML4SysML::LoopNode is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

StructuredActivityNode_Mapping

Mapping Source

LoopNode

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.8.2 SequenceNode_Mapping

Description

The UML4SysML::SequenceNode is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

StructuredActivityNode_Mapping

Mapping Source

SequenceNode

Mapping Target

ActionUsage

Owned Mappings

(none)

7.7.2.3.8.3 StructuredActivityNode_Mapping

Description

The UML4SysML::StructuredActivityNode is mapped to a SysML v2 ActionUsage. The details of the mapping are not defined yet.

General Mappings

CommonAction_Mapping

Mapping Source

StructuredActivityNode

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
let initialNodes : Set(UML::Element) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::InitialNode)) in
let finalNodes : Set(UML::Element) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::FinalNode)) in
let objectFlowsWithGuard : Set(UML::ObjectFlow) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::ObjectFlow)
        and not e.oclastype(UML::ObjectFlow).guard.oclisUndefined()) in
let objectFlows : Set(UML::ObjectFlow) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::ObjectFlow)) in
let ignoreInterruptibleActivityRegion: Set(UML::InterruptibleActivityRegion) =
    from.ownedElement->select(e | e.oclIsKindOf(UML::InterruptibleActivityRegion)) in
let elementsFMS : Set(UML::Element) =
    ((from.ownedElement->select(e | e.oclIsKindOf(UML::ControlNode) or
        e.oclIsKindOf(UML::Action) or (e.oclIsKindOf(UML::ControlFlow) or
        e.oclIsKindOf(UML::Pin))) - initialNodes) - finalNodes) in
let elementsOMS: Set(UML::Element) =
    (((((from.ownedElement-initialNodes)-finalNodes)-objectFlowsWithGuard)
        -objectFlows)-elementsFMS)-ignoreInterruptibleActivityRegion) in
elementsOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(elementsFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
->union(initialNodes->collect(e | InitialNodeMembership_Mapping.getMapped(e)))
```

```

->union(finalNodes->collect(e | FlowFinalNodeMembership_Mapping.getMapped(e)))
->union(objectFlowsWithGuard
    ->collect(e | ObjectFlowGuardFeatureMembership_Mapping.getMapped(e)))
->union(objectFlows->collect(e | ObjectFlowFeatureMembership_Mapping.getMapped(e)))

```

7.7.2.3.9 Variable Actions

[SYML2-16: Subsections for mapping classes in section 7.7.2.3.9 should be ordered alphabetically](#)

7.7.2.3.9.1 AddVariableValueAction_Mapping

Description

A UML4SysML::AddVariableValueAction is mapped to a SysML v2 ActionUsage defined by the SysML v1 library action definition SysMLv1Library::AddValueAction. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```

action def SysMLv1Activity {
    private attribute sysMLv1Variable1 : ScalarValues::Integer;
    private attribute sysMLv1Variable2 [0..*] : ScalarValues::Integer;

    action sysMLv1AddVariableValueAction1 : SysMLv1Library::AddValueAction {
        :>> target := sysMLv1Variable1;
    }

    action sysMLv1AddVariableValueAction1 : SysMLv1Library::AddValueAction {
        :>> target := thisIsAVariable;
        :>> isReplaceAll := true;
    }
}

```

General Mappings

CommonAction_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
let relationships : Set(KerML::Relationship) =
Set{AVVAFeatureTyping_Mapping.getMapped(from)}
->including(AVVAVariableFeatureMembership_Mapping.getMapped(from)) in
if from.isReplaceAll then
    relationships->including(AVVAIsReplaceAllFeatureMembership_Mapping.getMapped(from))
else
    relationships
endif
```

7.7.2.3.9.2 AVVAFeatureTyping_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYSML2::ActionDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::AddValueAction')
```

7.7.2.3.9.3 AVVAFeatureValue_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
AVVAValueFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.9.4 AVVAIsReplaceAll_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

The mapping class creates a reference usage element as mapping target for the AddVariableValueAction::isReplaceAll property.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set{AVVAIsReplaceAllRedefinition_Mapping.getMapped(from),  
AVVAIsReplaceAllValue_Mapping.getMapped(from),  
AssignmentActionUsageOwningMembership_Factory.create()}
```

7.7.2.3.9.5 AVVAIsReplaceAllFeatureMembership_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
AVVAIsReplaceAll_Mapping.getMapped(from)
```

7.7.2.3.9.6 AVVAIsReplaceAllRedefinition_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

General Mappings

GenericToRedefinition_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

Redefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::ReferenceUsage.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::AddValueAction::isReplaceAll')
```

7.7.2.3.9.7 AVVAIsReplaceAllValue_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

The mapping class maps the value of the AddVariableValueAction::isReplaceAll property.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]
 LiteralBoolean_Factory.create(from.isReplaceAll)

7.7.2.3.9.8 AVVAValueExpressionMembership_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

 from.variable

7.7.2.3.9.9 AVVAValueFeatureReferenceExpression_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

The mapping class creates the feature reference expression element for the UML4SysML::AddStructuralFeatureValueAction mapping.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{AVVAValueExpressionMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()}
```

7.7.2.3.9.10 AVVAVariable_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

The mapping class creates a reference usage element for the UML4SysML::AddVariableValueAction mapping.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]
Set{AVVAVariableRedefinition_Mapping.getMapped(from),
AVVAFeatureValue_Mapping.getMapped(from),
AssignmentActionUsageOwningMembership_Factory.create()}

7.7.2.3.9.11 AVVAVariableFeatureMembership_Mapping

[**SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct**](#)

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
AVVAVariable_Mapping.getMapped(from)
```

7.7.2.3.9.12 AVVAVariableRedefinition_Mapping

SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct

Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

General Mappings

GenericToRedefinition_Mapping

Mapping Source

AddVariableValueAction

Mapping Target

Redefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::ReferenceUsage.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::AddValueAction::target')
```

7.7.2.3.9.13 ClearVariableAction_Mapping

Description

The UML4SysML::ClearVariableAction is mapped to a SysML v2 ActionUsage that sets the attribute usage representing the variable to null.

The expected SysML v2 textual notation of a SysMLv1::ClearVariableAction is as follows

```
action def SysMLv1Activity {  
    private attribute sysMLv1Variable : ScalarValues::Integer;  
  
    action sysMLv1ClearVariableAction {  
        sysMLv1Variable := null;  
    }  
}
```

General Mappings

CommonAction_Mapping

Mapping Source

ClearVariableAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
->including(CVAFeatureMembership_Mapping.getMapped(from))
```

7.7.2.3.9.14 CVAFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

ClearVariableAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
CVAReferenceUsage_Mapping.getMapped(from)
```

7.7.2.3.9.15 CVAReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

ClearVariableAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::declaredName () : String [0..1]

```
from.variable.name
```

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set{CVAReferenceUsageFeatureValue_Mapping.getMapped(from),  
AssignmentActionUsageOwningMembership_Factory.create()}
```

7.7.2.3.9.16 CVAReferenceUsageFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

ClearVariableAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

 LiteralNull_Factory.create()

7.7.2.3.9.17 ReadVariableAction_Mapping

Description

A UML4SysML::ReadVariableValueAction is mapped to a SysML v2 ActionUsage with an out parameter that returns the value of the attribute usage that is the transformation target of the UML4SysML::Variable.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    private attribute sysMLv1Variable : ScalarValues::Integer;

    action sysMLv1ReadVariableAction {
        out result : ScalarValues::Integer = sysMLv1Variable;
    }
}
```

General Mappings

CommonAction_Mapping

Mapping Source

ReadVariableAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]
Set{RVAFeatureMembership_Mapping.getMapped(from)}

7.7.2.3.9.18 RVAFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

ReadVariableAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]
RVAReferenceUsage_Mapping.getMapped(from.result)

7.7.2.3.9.19 RVAReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

Pin

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
let featureTyping : Set(KerML::FeatureTyping) =
    if from.type.oclIsUndefined() then
        Set{}
    else
        Set{RVAReferenceUsageFeatureTyping_Mapping.getMapped(from)}
    endif in
featureTyping
->including(RVAReferenceUsageFeatureValue_Mapping.getMapped(from))
```

7.7.2.3.9.20 RVAReferenceUsageFeatureReferenceExpression_Mapping

Description

The mapping class creates the feature reference expression element for the UML4SysML::ReadVariableAction mapping.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

Pin

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set{RVAReferenceUsageExpressionMembership_Mapping.getMapped(from),  
ReturnParameterFeatureMembership_Factory.create()}
```

7.7.2.3.9.21 RVAReferenceUsageFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

TypedElementFeatureTyping_Mapping

Mapping Source

Pin

Mapping Target

FeatureTyping

Owned Mappings

(none)

7.7.2.3.9.22 RVAReferenceUsageFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

Pin

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
RVAReferenceUsageFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.9.23 RVAReferenceUsageExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

Pin

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from.owner.oclAsType(UML::ReadVariableAction).variable
```

7.7.2.3.9.24 RemoveVariableValueAction_Mapping

Description

A UML4SysML::RemoveVariableValueAction is mapped to a SysML v2 ActionUsage defined by the SysML v1 library action definition SysMLv1Library::RemoveVariableValueAction.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    private sysMLv1Variable : ScalarValues::Integer;

    action sysMLv1RemoveVariableValueAction
        : SysMLv1Library::RemoveVariableValueAction {
            :>> variable := sysMLv1Variable;
    }
}
```

General Mappings

CommonAction_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

ActionUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..*]

```
Helper.actionOwnedRelationship(from)
->including (RVVAFeatureTyping_Mapping.getMapped(from))
->including (RVVAVariableFeatureMembership_Mapping.getMapped(from))
```

7.7.2.3.9.25 RVVAFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYSML2::ActionDefinition.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::RemoveVariableValueAction')
```

7.7.2.3.9.26 RVVAVariable_Mapping

SYSML2-244: RVVAVariable_Mapping uses

CommonAssignmentActionOwningMembership_Mapping, but should be a factory class

Description

The mapping class creates a reference usage element for the UML4SysML::RemoveVariableValueAction mapping.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]
Set{RVVAVariableRedefinition_Mapping.getMapped(from),
RVVAVariableFeatureValue_Mapping.getMapped(from),
AssignmentActionUsageOwningMembership_Factory.create()}

7.7.2.3.9.27 RVVAVariableExpressionMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]
from.variable

7.7.2.3.9.28 RVVAVariableFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

RVVAVariable_Mapping.getMapped(from)

7.7.2.3.9.29 RVVAVariableFeatureReferenceExpression_Mapping

SYSML2-174: EmptyReturnParameterFeatureMembership_Mapping does not exist

Description

The mapping class creates the feature reference expression element for the UML4SysML::RemoveVariableValueAction mapping.

General Mappings

GenericToFeatureReferenceExpression_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

FeatureReferenceExpression

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..*]

```
Set<RVVAVariableExpressionMembership_Mapping> getMapped(from),  
ReturnParameterFeatureMembership_Factory.create() }
```

7.7.2.3.9.30 RVVAVariableFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
RVVAVariableFeatureReferenceExpression_Mapping.getMapped(from)
```

7.7.2.3.9.31 RVVAVariableRedefinition_Mapping

Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

General Mappings

GenericToRedefinition_Mapping

Mapping Source

RemoveVariableValueAction

Mapping Target

Redefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::ReferenceUsage.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::RemoveVariableValueAction::variable')
```

7.7.3 Activities

This chapter lists all mapping specifications of UML4SysML::Activities model elements.

7.7.3.1 Overview

SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters

SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column

Table 3. List of all mappings

SysML v1 Abstract Syntax/Stereotype	SysML v2 Abstract Syntax
Activity	ViewDefinition ActionDefinition RequirementUsage
ActivityFinalNode	not mapped; see next section
ActivityParameterNode	not mapped; see next section
ActivityPartition	not mapped; see next section
CentralBufferNode	ActionUsage
ControlFlow	TransitionUsage SuccessionAsUsage
DataStoreNode	ActionUsage
DecisionNode	DecisionNode
ExceptionHandler	not mapped; see next section
FlowFinalNode	not mapped; see next section
ForkNode	ForkNode
InitialNode	not mapped; see next section
InterruptibleActivityRegion	not mapped; see next section
JoinNode	JoinNode

SysML v1 Abstract Syntax/Stereotype	SysML v2 Abstract Syntax
MergeNode	MergeNode
ObjectFlow	TransitionUsage SuccessionFlowConnectionUsage
Variable	not mapped; see next section

The following table gives an overview of which SysML v2 elements the UML4SysML::Activities elements are transformed with which mapping class. The mapping details are in [7.7.3.3](#).

The justifications for the elements without mapping are given in [7.7.3.2](#).

7.7.3.2 UML4SysML::Activities elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

[SYSML2-566](#): Section containing tables about elements not mapped should get an introductory text

Table 4. List of SysML v1 elements not mapped of this section

SysML v1 Concept	Rationale
ActivityFinalNode	Mapping is not specified yet.
ActivityParameterNode	The parameter of the activity is mapped from SysML v1 to SysML v2. The additional concept of the activity parameter node is necessary for the token semantic of SysML v1 activities, which is not part of SysML v2. Therefore, the additional concept of the activity parameter node is not mapped to SysML v2.
ActivityPartition	Mapping is not specified yet.
ExceptionHandler	Mapping is not specified yet.
InterruptibleActivityRegion	Mapping is not specified yet.

7.7.3.3 Mapping Specifications

[SYSML2-221](#): UML4SysML::Activities and StateMachines owned by blocks should be mapped to definition elements

7.7.3.3.1 ActivityAsDefinition_Mapping

[SYSML2-202](#): Filter for mapping class Behavior_Mapping is useless

[SYSML2-7](#): Pin_Mapping::filter: property src should be from

[SYSML2-221](#): UML4SysML::Activities and StateMachines owned by blocks should be mapped to definition elements

Description

A UML4SysML::Activity is mapped to a SysMLv2 ActionDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
    in parIn : SysMLv1Block;
    out parOut;
    out parReturn;
}
part def SysMLv1Block;
```

General Mappings

Behavior_Mapping

Mapping Source

Activity

Mapping Target

ActionDefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionDefinition::ownedRelationship () : Relationship [0..*]

```
let relationships : Set(KerML::Relationship) =
    Helper.activityOwnedRelationship(from) in
let parameters : Set(UML::Paramter) =
    from.ownedElement->select(e | e.ocliIsKindOf(UML::Parameter)) in
relationships->union(parameters
    ->collect(p | ParameterMembership_Mapping.getMapped(p))
)
```

7.7.3.3.2 ActivityEdgeInitialNodeFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToEndFeatureMembership_Mapping

Mapping Source

InitialNode

Mapping Target

EndFeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

```
ActivityEdgeSourceInitialNode_Mapping.getMapped(from)
```

7.7.3.3 ActivityEdgeMetadata_Mapping

Description

Adds metadata to the transformation target elements of UML4SysML::ControlFlow and UML::ObjectFlow to map the UML4SysML::ActivityEdge::weight property which has no direct target in SysML v2.

General Mappings

GenericToMetadataUsage_Mapping

Mapping Source

ActivityEdge

Mapping Target

MetadataUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::declaredName () : String [0..1]

- 'weight'
- MetadataUsage::ownedRelationship () : Relationship [0..*]


```
Set<ActivityEdgeMetadataFeatureTyping_Mapping.getMapped(from),  
ActivityEdgeMetadataFeatureMembership_Mapping.getMapped(from) }
```

7.7.3.3.4 ActivityEdgeMetadataFeatureMembership_Mapping

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToFeatureMembership_Mapping

Mapping Source

ActivityEdge

Mapping Target

FeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]


```
ActivityEdgeMetadataReferenceUsage_Mapping.getMapped(from)
```

7.7.3.3.5 ActivityEdgeMetadataFeatureTyping_Mapping

Description

Creates a feature typing relationship owned by the element *typedFeature()*.

General Mappings

GenericToFeatureTyping_Mapping

Mapping Source

ActivityEdge

Mapping Target

FeatureTyping

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYSPML2::MetadataDefinition.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::ActivityEdgeData')
```

7.7.3.3.6 ActivityEdgeMetadataFeatureValue_Mapping

Description

Creates a feature value relationship.

General Mappings

GenericToFeatureValue_Mapping

Mapping Source

ActivityEdge

Mapping Target

FeatureValue

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
from.weight
```

7.7.3.3.7 ActivityEdgeMetadataOwningMembership_Mapping

Description

Creates a owning membership relationship for *ownedMemberElement()*.

General Mappings

GenericToOwningMembership_Mapping

Mapping Source

ActivityEdge

Mapping Target

OwningMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
ActivityEdgeMetadata_Mapping.getMapped(from)
```

7.7.3.3.8 ActivityEdgeMetadataRedefinition_Mapping

Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

General Mappings

GenericToRedefinition_Mapping

Mapping Source

ActivityEdge

Mapping Target

Redefinition

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::AttributeUsage.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::ActivityEdgeData::weight')
```

7.7.3.3.9 ActivityEdgeMetadataReferenceUsage_Mapping

Description

Creates a reference usage.

General Mappings

GenericToReferenceUsage_Mapping

Mapping Source

ActivityEdge

Mapping Target

ReferenceUsage

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..*]

```
Set{ActivityEdgeMetadataRedefinition_Mapping.getMapped(from),  
ActivityEdgeMetadataFeatureValue_Mapping.getMapped(from)}
```

7.7.3.3.10 ActivityEdgeSourceEndFeature_Mapping

Description

Creates a SysML v2 feature for the source activity node of the SysML v1 activity edge which subsets the SysML v2 target element of the source activity node.

General Mappings

GenericToFeature_Mapping

Mapping Source

Element

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]
 true
- Feature::ownedRelationship () : Relationship [0..*]
 Set{ActivityEdgeSourceEndSubsetting_Mapping.getMapped(from)}

7.7.3.3.11 ActivityEdgeSourceInitialNode_Mapping

Description

The UML4SysML::InitialNode is mapped to a subsetted feature of the SysML v2 library element Actions::start.

General Mappings

GenericToFeature_Mapping

Mapping Source

InitialNode

Mapping Target

Feature

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]
true
- Feature::ownedRelationship () : Relationship [0..*]

```
Set<ActivityEdgeSourceInitialNodeSubsetting_Mapping>.getMapped(from)
```

7.7.3.3.12 ActivityEdgeSourceEndFeatureMembership_Mapping

SYSML2-304: Mapping of ActivityEdge does not consider ActivityParameterNodes

Description

Creates a feature membership relationship for *ownedMemberFeature()*.

General Mappings

GenericToEndFeatureMembership_Mapping

Mapping Source

Element

Mapping Target

EndFeatureMembership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

```
ActivityEdgeSourceEndFeature_Mapping.getMapped(from)
```

7.7.3.3.13 ActivityEdgeSourceInitialNodeSubsetting_Mapping

SYSML2-200: Description of Subsetting mapping classes is not correct

SYSML2-197: ControlFlow target SuccessionAsUsage should have end feature with reference subsetting

Description

Creates a subsetting relationship.

General Mappings

GenericToReferenceSubsetting_Mapping

Mapping Source

InitialNode

Mapping Target

ReferenceSubsetting

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

```
SYSML2::ActionUsage.allInstances()  
->any(m | m.qualifiedName = 'Actions::Action::start')
```

7.7.3.3.14 ActivityEdgeSourceEndSubsetting_Mapping

SYSML2-200: Description of Subsetting mapping classes is not correct

SYSML2-197: ControlFlow target SuccessionAsUsage should have end feature with reference subsetting

Description

Creates a subsetting relationship.

General Mappings

GenericToReferenceSubsetting_Mapping

Mapping Source

Element

Mapping Target

ReferenceSubsetting

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

from

7.7.3.3.15 ActivityEdgeTransitionUsageSourceMembership_Mapping

Description

Creates a membership relationship for *memberElement()*.

General Mappings

GenericToMembership_Mapping

Mapping Source

ActivityNode

Mapping Target

Membership

Owned Mappings

(none)

Applicable filters

(none)

Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]
- ```
if from.oclIsTypeOf(UML::ActivityParameterNode) then
 from.parameter
else
 from
endif
```

### **7.7.3.3.16 CentralBufferNode\_Mapping**

#### **Description**

The mapping of the UML4SysML::CentralBufferNode is not defined in detail yet. It will be an action usage which contains the behavior of a central buffer node.

#### **General Mappings**

GenericToActionUsage\_Mapping  
NamedElementMain\_Mapping

#### **Mapping Source**

CentralBufferNode

#### **Mapping Target**

ActionUsage

#### **Owned Mappings**

(none)

### **7.7.3.3.17 CommonActivityEdgeSuccessionAsUsage\_Mapping**

#### ***SYSML2-304: Mapping of ActivityEdge does not consider ActivityParameterNodes***

#### **Description**

The mapping class provides a common mapping of a UML4SysML::ActivityEdge to a SysMLv2 SucessionAsUsage. The mapping is used for UML4SysML::ControlFlows and UML4SysML::ObjectFlows.

#### **General Mappings**

GenericToConnector\_Mapping

#### **Mapping Source**

ActivityEdge

#### **Mapping Target**

SuccessionAsUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SuccessionAsUsage::ownedRelationship () : Relationship [0..\*]

```
let relationships : Set(KerML::Relationship) = Set{
 if from.source.oclIsKindOf(UML::InitialNode) then
 ActivityEdgeInitialNodeFeatureMembership_Mapping.getMapped(from.source)
 else if from.source.oclIsKindOf(UML::ActivityParameterNode) then
 ActivityEdgeSourceEndFeatureMembership_Mapping.getMapped(from.source.parameter)
 else
 ActivityEdgeSourceEndFeatureMembership_Mapping.getMapped(from.source)
 endif
endif,
if from.oclIsKindOf(UML::ObjectFlow) then
 ObjectFlowGuardSuccessionTargetEndFeatureMembership_Mapping.getMapped(from)
else if from.target.oclIsKindOf(UML::FinalNode) then
 ControlFlowFinalNodeFeatureMembership_Mapping.getMapped(from.target)
else
 ControlFlowTargetFeatureMembership_Mapping.getMapped(from.target)
endif
endif} in
if from.guard.oclIsUndefined() then
 relationships
else
 relationships
 ->including(ElementFeatureMembership_Mapping.getMapped(from.guard))
endif
```

### 7.7.3.3.18 CommonVariable\_Mapping

#### Description

Abstract mapping class for UML4SysML::Variable which is defined in the context of UML4SysML::Activity. A UML4SysML::Variable is mapped to a SysMLv2 AttributeUsage or SysMLv2 ItemUsage. See specialized mapping classes for the specific mapping rules.

#### General Mappings

PropertyCommon\_Mapping

#### Mapping Source

Variable

#### Mapping Target

Feature

#### Owned Mappings

(none)

#### Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]

```
 false
```
- Feature::isComposite () : Boolean [1]

```
 false
```
- Feature::ownedRelationship () : Relationship [0..\*]

```
let typing: KerML::FeatureTyping =
 VariableFeatureTyping_Mapping.getMapped(from) in
if typing.oclIsUndefined() then
 Set{MultiplicityMembership_Mapping.getMapped(from)}
else
 Set{MultiplicityMembership_Mapping.getMapped(from), typing}
endif
```
- Feature::isDerived () : Boolean [1]

```
 false
```

### 7.7.3.3.19 ControlFlowTransitionUsage\_Mapping

[SYSML2-211: Introduce GenericToTransitionUsage\\_Mapping class](#)  
[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)  
[SYSML2-280: ElementMain\\_Mapping::ownedRelationship is wrong](#)

#### Description

A UML4SysML::ControlFlow with a guard condition is mapped to a SysMLv2 TransitionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 action sysMLv1Action1;
 succession sysMLv1ControlFlow first sysMLv1Action1
 if guardCondition.result then sysMLv1Action2 {
 calc guardCondition {
 return : ScalarValues::Boolean;
 language "English"
 /*
 * thisIsAGuard
 */
 }
 }
 action sysMLv1Action2;
}
```

#### General Mappings

GenericToTransitionUsage\_Mapping  
NamedElementMain\_Mapping

### Mapping Source

ControlFlow

### Mapping Target

TransitionUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
not src.guard.oclIsUndefined()
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TransitionUsage::ownedRelationship () : Relationship [0..\*]

```
let relationships : Set(KerML::Relationship) = self.oclAsType(ElementMain_Mapping).ownedRelationships
->union(Set{ActivityEdgeTransitionUsageSourceMembership_Mapping.getMapped(from.source),
,CommonParameterReferenceUsageInMembership_Mapping.getMapped(from.source)
,ControlFlowTransitionUsageFeatureMembership_Mapping.getMapped(from)
,CommonActivityEdgeSuccessionAsUsage_Mapping.getMapped(from)
,CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from) }) in
let relationshipsWithGuard : Set(KerML::Relationship) =
if from.guard.oclIsTypeOf(UML::OpaqueExpression) then
 relationships
 ->including(ElementFeatureMembership_Mapping.getMapped(from.guard))
else
 relationships
endif in
let relationshipsConsideringWeight : Set(KerML::Relationship) =
if from.weight.oclIsUndefined() then
 relationshipsWithGuard
else
 relationshipsWithGuard
 ->including(ActivityEdgeMetadataOwningMembership_Mapping.getMapped(from))
endif in
if Helper.hasStereotypeApplied(from, 'SysML::Activities::Probability') then
 relationshipsConsideringWeight
 ->including(ProbabilityOwningMembership_Mapping.getMapped(from))
else
 relationshipsConsideringWeight
endif
```

### 7.7.3.3.20 ControlFlowFinalNodeFeatureMembership\_Mapping

## Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## General Mappings

GenericToEndFeatureMembership\_Mapping

## Mapping Source

ActivityNode

## Mapping Target

EndFeatureMembership

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]  
ControlFlowTargetFinalNode\_Mapping.getMapped (from)

## 7.7.3.3.21 ControlFlowTargetFinalNodeSubsetting\_Mapping

**SYSML2-200: Description of Subsetting mapping classes is not correct**

**SYSML2-197: ControlFlow target SuccessionAsUsage should have end feature with reference subsetting**

## Description

Creates a subsetting relationship.

## General Mappings

GenericToReferenceSubsetting\_Mapping

## Mapping Source

FinalNode

## Mapping Target

ReferenceSubsetting

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

```
SYSML2::ActionUsage.allInstances()
->any(m | m.qualifiedName = 'Actions::Action::done')
```

### **7.7.3.3.22 ControlFlowSuccessionAsUsage\_Mapping**

**SYSML2-229: ControlFlowSuccessionAsUsage\_Mapping uses non-existing mapping class**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-193: ControlFlowSuccessionAsUsage\_Mapping uses non existing mapping class**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

**SYSML2-189: ControlFlowSuccessionAsUsage\_Mapping uses non existing mapping class**

**ActivityEdgeInitialNodeSourceEndFeatureMembership\_Mapping**

### **Description**

A UML4SysML::ControlFlow without a guard condition is mapped to a SysMLv2 SuccessionAsUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 action sysMLv1Action1;
 succession sysMLv1ControlFlow
 first sysMLv1Action1 then sysMLv1Action2;
 action sysMLv1Action2;
}
```

### **General Mappings**

NamedElementMain\_Mapping

CommonActivityEdgeSuccessionAsUsage\_Mapping

### **Mapping Source**

ControlFlow

### **Mapping Target**

SuccessionAsUsage

### **Owned Mappings**

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
src.guard.oclIsUndefined()
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SuccessionAsUsage::ownedRelationship () : Relationship [0..\*]

```
let relationships : Set(KerML::Relationship) = Set{
 if from.source.oclIsKindOf(UML::InitialNode) then
 ActivityEdgeInitialNodeFeatureMembership_Mapping.getMapped(from.source)
 else
 ActivityEdgeSourceEndFeatureMembership_Mapping.getMapped(from.source)
 endif,
 if from.oclIsKindOf(UML::ObjectFlow) then
 ObjectFlowGuardSuccessionTargetEndFeatureMembership_Mapping.getMapped(from)
 else if from.target.oclIsKindOf(UML::FinalNode) then
 ControlFlowFinalNodeFeatureMembership_Mapping.getMapped(from.target)
 else
 ControlFlowTargetFeatureMembership_Mapping.getMapped(from.target)
 endif
endif} in
let relationshipsWithGuard : Set(KerML::Relationship) =
if from.guard.oclIsUndefined() then
 relationships
else
 relationships
 ->including(ElementFeatureMembership_Mapping.getMapped(from.guard))
endif in
let relationshipsConsideringWeight : Set(KerML::Relationship) =
if from.weight.oclIsUndefined() then
 relationshipsWithGuard
else
 relationshipsWithGuard
 ->including(ActivityEdgeMetadataOwningMembership_Mapping.getMapped(from))
endif in

(if Helper.hasStereotypeApplied(from, 'SysML::Activities::Probability') then
 relationshipsConsideringWeight
 ->including(ProbabilityOwningMembership_Mapping.getMapped(from))
else
 relationshipsConsideringWeight
endif)->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

### 7.7.3.3.23 ControlFlowTargetFinalNode\_Mapping

#### Description

The mapping class maps a UML4SysML::FinalNode to a Feature which will be subsetted by Actions::Action::done. The subsetting is created by the mapping class ControlFlowTargetFinalNodeSubsetting\_Mapping.

### **General Mappings**

GenericToFeature\_Mapping

### **Mapping Source**

FinalNode

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]  
    true
- Feature::ownedRelationship () : Relationship [0..\*]  

```
Set<ControlFlowTargetFinalNodeSubsetting_Mapping.getMapped(from) }
```

### **7.7.3.3.24 ControlFlowTargetEndFeature\_Mapping**

**SYSML2-197: ControlFlow target SuccessionAsUsage should have end feature with reference subsetting**

### **Description**

The mapping class maps the UML4SysML::ActivityNode to a Feature which is subsetted by the mapping target of the UML4SysML::ActivityNode. The subsetting is created by the mapping class ControlFlowTargetEndSubsetting\_Mapping.

### **General Mappings**

GenericToFeature\_Mapping

### **Mapping Source**

ActivityNode

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]  
true
- Feature::ownedRelationship () : Relationship [0..\*]  

```
Set<ControlFlowTargetEndSubsetting_Mapping.getMapped(from) >
```

### **7.7.3.3.25 ControlFlowTargetFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToEndFeatureMembership\_Mapping

#### **Mapping Source**

ActivityNode

#### **Mapping Target**

EndFeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]  

```
ControlFlowTargetEndFeature_Mapping.getMapped(from)
```

### **7.7.3.3.26 ControlFlowTargetEndSubsetting\_Mapping**

**SYSML2-200:** Description of Subsetting mapping classes is not correct

**SYSML2-197:** ControlFlow target SuccessionAsUsage should have end feature with reference subsetting

#### **Description**

Creates a subsetting relationship.

#### **General Mappings**

GenericToReferenceSubsetting\_Mapping

#### **Mapping Source**

ActivityNode

#### **Mapping Target**

ReferenceSubsetting

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

from

### **7.7.3.3.27 ControlFlowTransitionUsageFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

ControlFlow

#### **Mapping Target**

TransitionFeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TransitionFeatureMembership::kind () : TransitionFeatureKind [1]  

```
KerML::TransitionFeatureKind::guard
```
- TransitionFeatureMembership::ownedMemberFeature () : Feature [1]  

```
if from.guard.oclIsKindOf(UML::OpaqueExpression) then
 OpaqueExpressionAsValue_Mapping.getMapped(from.guard)
else
 from.guard
endif
```

## **7.7.3.3.28 DataStoreNode\_Mapping**

### **Description**

The mapping of the UML4SysML::DataStoreNode is not defined in detail yet. It will an action usage which contains the behavior of a data store node.

### **General Mappings**

CentralBufferNode\_Mapping

### **Mapping Source**

DataStoreNode

### **Mapping Target**

ActionUsage

### **Owned Mappings**

(none)

## **7.7.3.3.29 DecisionNode\_Mapping**

### **Description**

The UML4SysML::DecisionNode is mapped to a SysMLv2 DecisionNode.

There is no suitable element in SysML v2 for the else condition of an outgoing UML4SysML::ActivityEdge. Therefore, it is mapped to a TextualRepresentation with language "SysML v1" and body "else" (see ExpressionElse\_Mapping class).

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 action sysMLv1Action1;
 succession sysMLv1ControlFlow1 first sysMLv1Action1 then sysMLv1DecisionNode;
 decide sysMLv1DecisionNode;
 succession sysMLv1ControlFlow2 first sysMLv1DecisionNode if {
 return : ScalarValues::Boolean;
 // guard expression, for example, opaque expression
 }.result then sysMLv1Action2;
 succession flow2 first sysMLv1DecisionNode if {
 return : ScalarValues::Boolean;
 language "SysMLv1"
 /*
 * else
 */
 }.result then sysMLv1Action2;
 action sysMLv1Action2;
}
```

## General Mappings

GenericToUsage\_Mapping  
NamedElementMain\_Mapping

### Mapping Source

DecisionNode

### Mapping Target

DecisionNode

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- DecisionNode::isComposite () : Boolean [1]

true

### 7.7.3.3.30 FlowFinalNodeMembership\_Mapping

#### Description

The mapping class creates a membership relationship to the action usage library element Actions::Action::done.

## **General Mappings**

GenericToMembership\_Mapping

### **Mapping Source**

FlowFinalNode

### **Mapping Target**

Membership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]  

```
SysMLv2::ActionUsage.allInstances()
->any(e | e.qualifiedName = 'Actions::Action::done')
```

## **7.7.3.3.31 ForkNode\_Mapping**

### **Description**

The UML4SysML::ForkNode is mapped to a SysMLv2 ForkNode.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 first start;
 action sysMLv1Action1;

 then fork sysMLv1ForkNode;

 then sysMLv1Action2;
 then sysMLv1Action3;
 action sysMLv1Action2;
 then sysMLv1JoinNode;
 action sysMLv1Action3;
 then sysMLv1JoinNode;

 join sysMLv1JoinNode;

 then done;
}
```

### **General Mappings**

GenericToUsage\_Mapping  
NamedElementMain\_Mapping

### **Mapping Source**

ForkNode

### **Mapping Target**

ForkNode

### **Owned Mappings**

(none)

## **7.7.3.3.32 InitialNodeMembership\_Mapping**

### **Description**

The mapping class creates a membership relationship to the action usage library element Actions::Action::start.

### **General Mappings**

GenericToMembership\_Mapping

### **Mapping Source**

InitialNode

### **Mapping Target**

Membership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberName () : String [0..1]  

```
if from.name = '' then null else from.name endif
```
- Membership::memberElement () : Element [1]

```
SysMLv2::ActionUsage.allInstances()
->any(e | e.qualifiedName = 'Actions::Action::start')
```

### 7.7.3.3.33 JoinNode\_Mapping

#### Description

The UML4SysML::JoinNode is mapped to a SysMLv2JoinNode.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 first start;
 action sysMLv1Action1;

 then fork sysMLv1ForkNode;

 then sysMLv1Action2;
 then sysMLv1Action3;
 action sysMLv1Action2;
 then sysMLv1JoinNode;
 action sysMLv1Action3;
 then sysMLv1JoinNode;

 join sysMLv1JoinNode;

 then done;
}
```

#### General Mappings

GenericToUsage\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

JoinNode

#### Mapping Target

JoinNode

#### Owned Mappings

(none)

### 7.7.3.3.34 MergeNode\_Mapping

#### Description

The UML4SysML::MergeNode is mapped to a SysMLv2 MergeNode.

#### General Mappings

GenericToUsage\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

MergeNode

#### Mapping Target

MergeNode

#### Owned Mappings

(none)

### 7.7.3.3.35 ObjectFlow\_Mapping

**SYSML2-238: ObjectFlows targeting a final node or a activity parameter node cannot be mapped**  
**SYSML2-7: Pin\_Mapping::filter: property src should be from**  
**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### Description

A UML4SysML::ObjectFlowFlow without a guard condition is mapped to a SysMLv2SuccessionFlowConnectionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Acticity {
 action sysMLv1Action1 {
 out outputValue;
 }
 succession flow sysMLv1ObjectFlow of ScalarValues::String
 from sysMLv1Action1.outputValue to sysMLv1Action1.inputValue;
 action sysMLv1Action2 {
 out inputValue;
 }
}
```

#### General Mappings

GenericToConnector\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

ObjectFlow

#### Mapping Target

SuccessionFlowConnectionUsage

#### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.guard.oclIsUndefined()
and (not src.target.oclIsTypeOf(UML::ActivityFinalNode))
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SuccessionFlowConnectionUsage::ownedRelationship () : Relationship [0..\*]

```
let relationships : Set(KerML::Relationship) =
let sourceFeatureMembership : KerML::FeatureMembership = ObjectFlowEndFeatureMembership_Mapping.get
let targetFeatureMembership : KerML::FeatureMembership = ObjectFlowEndFeatureMembership_Mapping.get
if from.source.oclIsKindOf(UML::ObjectNode) then
 Set{ObjectFlowItemFeatureMembership_Mapping.getMapped(from),
 sourceFeatureMembership, targetFeatureMembership}
else
 Set{sourceFeatureMembership, targetFeatureMembership}
endif in

let relationshipsConsideringWeight : Set(KerML::Relationship) =
if from.weight.oclIsUndefined() then
 relationships
else
 relationships
 ->including(ActivityEdgeMetadataOwningMembership_Mapping.getMapped(from))
endif in

let relationshipsConsideringRate : Set(KerML::Relationship) =
if (Helper.hasStereotypeApplied(from, 'SysML::Activities::Rate') or
 Helper.hasStereotypeApplied(from, 'SysML::Activities::Discrete') or
 Helper.hasStereotypeApplied(from, 'SysML::Activities::Continuous')) then
 relationshipsConsideringWeight
 ->including(RateOwningMembership_Mapping.getMapped(from))
else
 relationshipsConsideringWeight
endif in

self.oclAsType(ElementMain_Mapping).ownedRelationship() ->union(
 if Helper.hasStereotypeApplied(from, 'SysML::Activities::Probability') then
 relationshipsConsideringRate
 ->including(ProbabilityOwningMembership_Mapping.getMapped(from))
 else
 relationshipsConsideringRate
 endif
)
```

### 7.7.3.3.36 ObjectFlowFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

### General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

ObjectFlow

### Mapping Target

FeatureMembership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ObjectFlow_Mapping.getMapped(from)
```

## 7.7.3.3.37 ObjectFlowGuardFeatureMembership\_Mapping

### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

### General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

ObjectFlow

### Mapping Target

FeatureMembership

### Owned Mappings

(none)

### Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ObjectFlowGuard_Mapping.getMapped(from)
```

### 7.7.3.3.38 ObjectFlowGuard\_Mapping

**SYSML2-211: Introduce GenericToTransitionUsage\_Mapping class**

**SYSML2-238: ObjectFlows targeting a final node or a activity parameter node cannot be mapped**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### Description

A UML4SysML::ObjectFlowFlow with a guard condition is mapped to a combined SysMLv2 TransitionUsage and SysMLv2 SuccessionFlowConnectionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 action sysMLv1Action1 {
 out outputValue;
 }

 first sysMLv1Action1 if guardCondition.result then sysMLv1ObjectFlow {
 calc guardCondition {
 return : ScalarValues::Boolean;
 language "English"
 /*
 * guard says ok
 */
 }
 }
 succession flow sysMLv1ObjectFlow of SysMLv1Block from
 sysMLv1Action1.outputValue to sysMLv1Action2.inputValue;

 action sysMLv1Action2 {
 out inputValue;
 }
}
```

#### General Mappings

GenericToTransitionUsage\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

ObjectFlow

#### Mapping Target

TransitionUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
(not src.guard.oclisUndefined())
and (not src.target.oclisTypeOf(UML::ActivityFinalNode))
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TransitionUsage::ownedRelationship () : Relationship [0..\*]

```
Set{
 ActivityEdgeTransitionUsageSourceMembership_Mapping.getMapped(from.source),
 CommonParameterReferenceUsageInMembership_Mapping.getMapped(from.source),
 ObjectFlowTransitionUsageFeatureMembership_Mapping.getMapped(from),
 ObjectFlowGuardSuccessionTargetEndFeatureMembership_Mapping.getMapped(from),
 CommonActivityEdgeSuccessionAsUsage_Mapping.getMapped(from),
 CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from)
} -> union(self.oclastype(ElementMain_Mapping).ownedRelationship())
```

## 7.7.3.3.39 ObjectFlowGuardSuccessionTargetEndFeature\_Mapping

### Description

Creates a feature element for the UML4SysML::ObjectFlow mapping.

### General Mappings

GenericToFeature\_Mapping

### Mapping Source

ObjectFlow

### Mapping Target

Feature

### Owned Mappings

- objectFlowGuardSuccessionTargetEndSubsetting :
 ObjectFlowGuardSuccessionTargetEndSubsetting\_Mapping

### Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]  
true
- Feature::ownedRelationship () : Relationship [0..\*]  
Set{objectFlowGuardSuccessionTargetEndSubsetting.to}

### 7.7.3.3.40 ObjectFlowGuardSuccessionTargetEndFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

#### General Mappings

GenericToEndFeatureMembership\_Mapping

#### Mapping Source

ObjectFlow

#### Mapping Target

EndFeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]  
ObjectFlowGuardSuccessionTargetEndFeature\_Mapping.getMapped(from)

### 7.7.3.3.41 ObjectFlowGuardSuccessionTargetEndSubsetting\_Mapping

#### **SYSML2-200: Description of Subsetting mapping classes is not correct**

#### Description

Creates a subsetting relationship.

## **General Mappings**

GenericToSubsetting\_Mapping

### **Mapping Source**

ObjectFlow

### **Mapping Target**

Subsetting

## **Owned Mappings**

- objectFlowGuardSuccessionTargetEndFeature : ObjectFlowGuardSuccessionTargetEndFeature\_Mapping

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subsetting::subsettingFeature () : Feature [1]  
objectFlowGuardSuccessionTargetEndFeature.to
- Subsetting::subsettedFeature () : Feature [1]  
ObjectFlow\_Mapping.getMapped(from)

## **7.7.3.3.42 ObjectFlowItemFeature\_Mapping**

### **Description**

The mapping class maps the source UML4SysML::ObjectNode to a ItemFeature which is typed by the UML4SysML::ObjectNode type.

## **General Mappings**

ObjectFlowItemFeatureUntyped\_Mapping

### **Mapping Source**

ObjectNode

### **Mapping Target**

ItemFeature

## **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ItemFeature::ownedRelationship () : Relationship [0..\*]  

```
Set{ObjectFlowItemFeatureTyping_Mapping.getMapped(from)}
```

## **7.7.3.3.43 ObjectFlowItemFeatureMembership\_Mapping**

### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

### **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

ObjectFlow

### **Mapping Target**

FeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  

```
if from.source.type.oclIsUndefined() then
 ObjectFlowItemFeatureUntyped_Mapping.getMapped(from.source)
else
 ObjectFlowItemFeature_Mapping.getMapped(from.source)
endif
```

## **7.7.3.3.44 ObjectFlowItemFeatureTyping\_Mapping**

### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

### **General Mappings**

TypedElementFeatureTyping\_Mapping

#### **Mapping Source**

ObjectNode

#### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

## **7.7.3.3.45 ObjectFlowItemFeatureUntyped\_Mapping**

#### **Description**

The mapping class maps the source UML4SysML::ObjectNode to a ItemFeature without a type.

### **General Mappings**

GenericToFeature\_Mapping

#### **Mapping Source**

ObjectNode

#### **Mapping Target**

ItemFeature

### **Owned Mappings**

(none)

## **7.7.3.3.46 ObjectFlowEndFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

### **General Mappings**

GenericToEndFeatureMembership\_Mapping

#### **Mapping Source**

ActivityNode

#### **Mapping Target**

EndFeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

```
ObjectFlowItemFlowEnd_Mapping.getMapped(from)
```

### **7.7.3.3.47 ObjectFlowItemFlowEnd\_Mapping**

**SYSML2-2: ItemFlowEnds of ObjectFlow transformation target are not defined correctly**

#### **Description**

The mapping class maps a UML4SysML::ActivityNode to a ItemFlowEnd which is subsetted by the transformation target of the UML4SysML::ActivityNode.

#### **General Mappings**

GenericToFeature\_Mapping

#### **Mapping Source**

ActivityNode

#### **Mapping Target**

ItemFlowEnd

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ItemFlowEnd::ownedRelationship () : Relationship [0..\*]

```
Set{ObjectFlowItemFlowEndSubsetting_Mapping.getMapped(from),
ObjectFlowItemFlowEndFeatureMembership_Mapping.getMapped(from)}
```

- ItemFlowEnd::isEnd () : Boolean [1]

true

### 7.7.3.3.48 ObjectFlowItemFlowEndReferenceUsage\_Mapping

**SYSML2-23: Transformation of UML4SysML::AddStructuralFeatureValueAction is not correct**  
**SYSML2-238: ObjectFlows targeting a final node or a activity parameter node cannot be mapped**  
**SYSML2-236: Resolution of approved issue SYSML2-23 uses outdated mapping classes**  
**SYSML2-2: ItemFlowEnds of ObjectFlow transformation target are not defined correctly**  
**SYSML2-4: Transformation of UML4SysML::AddVariableValueAction is not correct**

#### Description

Creates a feature element for the UML4SysML::ObjectFlow mapping.

#### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

ActivityNode

#### Mapping Target

ReferenceUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
let redefinition : KerML::Redefinition =
 if from.owner.oclIsTypeOf(UML::AddVariableValueAction) or
 from.owner.oclIsTypeOf(UML::AddStructuralFeatureValueAction) then
 if from.name = 'value' then
 ObjectFlowItemFlowEndRedefinition_Factory.create(SYSML2::ReferenceUsage.allInstances()
 ->any(m | m.qualifiedName = 'SysMLv1Library::AddValueAction::value'))
 else if from.name = 'insertAt' then
 ObjectFlowItemFlowEndRedefinition_Factory.create(SYSML2::ReferenceUsage.allInstances()
 ->any(m | m.qualifiedName = 'SysMLv1Library::AddValueAction::insertAt'))
 else if from.owner.oclIsTypeOf(UML::AddStructuralFeatureValueAction) and (from.name = 'obj')
 ObjectFlowItemFlowEndRedefinition_Factory.create(SYSML2::ReferenceUsage.allInstances()
 ->any(m | m.qualifiedName = 'SysMLv1Library::AddStructuralFeatureValueAction::obj'))
 else
```

```

 ObjectFlowItemFlowEndRedefinition_Factory.create(ElementMain_Mapping.getMapped(from))
 endif endif endif
else
 if from.oclIsTypeOf(UML::ActivityParameterNode) then
 ObjectFlowItemFlowEndRedefinition_Factory.create(
 ElementMain_Mapping.getMapped(from.oclAsType(UML::ActivityParameterNode)).parameter)
 else if from.oclIsTypeOf(UML::FlowFinalNode) then
 ObjectFlowItemFlowEndRedefinition_Factory.create(ElementMain_Mapping.getMapped(
 SysMLv2::ActionUsage.allInstances()-->any(e | e.qualifiedName = 'Actions::Action::done'))
 else
 ObjectFlowItemFlowEndRedefinition_Factory.create(ElementMain_Mapping.getMapped(from))
 endif endif
endif in
Set{redefinition}

```

### 7.7.3.3.49 ObjectFlowItemFlowEndFeatureMembership\_Mapping

**SYSML2-2: ItemFlowEnds of ObjectFlow transformation target are not defined correctly**

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

ActivityNode

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ObjectFlowItemFlowEndReferenceUsage_Mapping.getMapped(from)
```

### 7.7.3.3.50 ObjectFlowItemFlowEndRedefinition\_Mapping

**SYSML2-2: ItemFlowEnds of ObjectFlow transformation target are not defined correctly**

## Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

## General Mappings

GenericToRedefinition\_Mapping

### Mapping Source

ActivityNode

### Mapping Target

Redefinition

## Owned Mappings

(none)

### 7.7.3.3.51 ObjectFlowItemFlowEndSubsetting\_Mapping

**SYSML2-200: Description of Subsetting mapping classes is not correct**

**SYSML2-2: ItemFlowEnds of ObjectFlow transformation target are not defined correctly**

## Description

Creates a subsetting relationship.

## General Mappings

GenericToReferenceSubsetting\_Mapping

### Mapping Source

ActivityNode

### Mapping Target

ReferenceSubsetting

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

```

if from.oclIsKindOf(UML::ActivityParameterNode) then
 Parameter_Mapping.getMapped(from.parameter)
else if from.oclIsKindOf(UML::Pin) then
 CommonAction_Mapping.getMapped(from.owner)
else if from.oclIsKindOf(UML::InitialNode) then
 SysMLv2::ActionUsage.allInstances()
 ->any(e | e.qualifiedName = 'Actions::Action::start')
else if from.oclIsKindOf(UML::FinalNode) then
 SysMLv2::ActionUsage.allInstances()
 ->any(e | e.qualifiedName = 'Actions::Action::done')
else
 from
endif
endif
endif
endif
endif

```

### **7.7.3.3.52 ObjectFlowTransitionUsageFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

ObjectFlow

#### **Mapping Target**

TransitionFeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TransitionFeatureMembership::ownedMemberFeature () : Feature [1]
 

```

if from.guard.oclIsKindOf(UML::OpaqueExpression) then
 OpaqueExpressionAsValue_Mapping.getMapped(from.guard)
else
 from.guard
endif

```
- TransitionFeatureMembership::kind () : TransitionFeatureKind [1]

```
KerML::TransitionFeatureKind::guard
```

### 7.7.3.3.53 VariableAttribute\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

A UML4SysML::Variable is mapped to a SysML v2 AttributeUsage if the type of the variable is of kind UML4SysML::DataType.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 private attribute sysmlv1Variable : ScalarValues::Integer;
}
```

#### General Mappings

NamedElementMain\_Mapping  
CommonVariable\_Mapping

#### Mapping Source

Variable

#### Mapping Target

AttributeUsage

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.type.oclIsKindOf(UML:: DataType)
```

#### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.7.3.3.54 VariableFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

TypedElementFeatureTyping\_Mapping

### **Mapping Source**

Variable

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **7.7.3.3.55 VariableItem\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### **Description**

A UML4SysML::Variable is mapped to a SysML v2 ItemUsage if the type of the variable is not of kind UML4SysML::DataType.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 private item sysmlv1Variable : SysMLv1Block;
}
part def SysMLv1Block;
```

### **General Mappings**

NamedElementMain\_Mapping  
CommonVariable\_Mapping

### **Mapping Source**

Variable

### **Mapping Target**

ItemUsage

### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
not src.type.oclIsKindOf(UML:::DataType)
```

#### **Mapping rules**

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.7.3.3.56 VariableMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

ElementFeatureMembership\_Mapping

#### Mapping Source

Variable

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::visibility () : VisibilityKind [1]

KerML::VisibilityKind::private

## 7.7.4 Classification

**SYSML2-1: "Elements not mapped" table sections are empty**

**SYSML2-513: Missing text in some main mapping sections**

### 7.7.4.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**SYSML2-509: Remove sentence in Classification overview section**

**SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column**

Table 5. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax     |
|-------------------------------------|------------------------------|
| Generalization                      | Subclassification            |
| GeneralizationSet                   | not mapped; see next section |

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax                        |
|-------------------------------------|-------------------------------------------------|
| InstanceSpecification               | ConnectionUsage                                 |
| InstanceValue                       | FeatureReferenceExpression                      |
| Operation                           | PerformActionUsage                              |
| Parameter                           | ReferenceUsage                                  |
| ParameterSet                        | not mapped; see next section                    |
| Property                            | AttributeUsage                                  |
| Slot                                | Feature                                         |
| Substitution                        | SatisfyRequirementUsage<br>AllocationDefinition |

The following table gives an overview of which SysML v2 elements the UML4SysML::Classification elements are transformed with which mapping class. The mapping details are in [7.7.4.2](#).

## 7.7.4.2 Mapping Specifications

### 7.7.4.2.1 BehavioralFeature\_Mapping

#### Description

The mapping class is the abstract base class for UML4SysML::BehavioralFeature mappings.

#### General Mappings

GenericToUsage\_Mapping  
Namespace\_Mapping

#### Mapping Source

BehavioralFeature

#### Mapping Target

Usage

#### Owned Mappings

(none)

### 7.7.4.2.2 Classifier\_Mapping

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### Description

The mapping class is the abstract base class for all mapping classes that map specializations of UML4SysML::Classifier elements.

#### General Mappings

GenericToClassifier\_Mapping  
Namespace\_Mapping

#### Mapping Source

Classifier

#### Mapping Target

Classifier

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Classifier::isAbstract () : Boolean [1]

```
from.isAbstract
```

- Classifier::ownedRelationship () : Relationship [0..\*]

```
let generalizations : Set(UML::Generalization) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Generalization))->asSet() in
let toElementFMS: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Feature))->asSet() in
let toElementOMS: Set(UML::Element) =
 ((from.ownedElement - toElementFMS) - generalizations) - from.ownedComment in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e))->asSet())
->union(generalizations->collect(e | Generalization_Mapping.getMapped(e))->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

### 7.7.4.2.3 DefaultLowerBound\_Mapping

#### Description

The mapping class creates the default lower bound of a multiplicity element.

#### General Mappings

GenericToExpression\_Mapping

#### Mapping Source

Element

#### Mapping Target

LiteralInteger

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralInteger::ownedRelationship () : Relationship [0..\*]  
Set {CommonReturnParameterFeatureMembership\_Mapping.getMapped(from) }
- LiteralInteger::value () : Integer [1]  
1

### 7.7.4.2.4 DefaultMultiplicityBoundFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

Element

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::isComposite () : Boolean [1]

true

#### 7.7.4.2.5 DefaultMultiplicityElement\_Mapping

##### Description

The mapping class creates a feature element representing the default multiplicity.

##### General Mappings

GenericToFeature\_Mapping

##### Mapping Source

Element

##### Mapping Target

MultiplicityRange

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MultiplicityRange::declaredName () : String [0..1]  

```
'defaultMultiplicity'
```
- MultiplicityRange::isUnique () : Boolean [1]  

```
true
```
- MultiplicityRange::ownedRelationship () : Relationship [0..\*]  

```
OrderedSet{DefaultMultiplicityLowerBoundFeatureMembership_Mapping.getMapped(from),
DefaultMultiplicityUpperBoundFeatureMembership_Mapping.getMapped(from) }
```

#### 7.7.4.2.6 DefaultMultiplicityLowerBoundFeatureMembership\_Mapping

##### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

##### General Mappings

DefaultMultiplicityBoundFeatureMembership\_Mapping

##### Mapping Source

Element

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : MultiplicityRange [1]

DefaultLowerBound\_Mapping.getMapped(from)

### 7.7.4.2.7 DefaultMultiplicityMembership\_Mapping

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToOwningMembership\_Mapping

**Mapping Source**

Element

**Mapping Target**

OwningMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
DefaultMultiplicityElement_Mapping.getMapped(from)
```

#### 7.7.4.2.8 DefaultMultiplicityUpperBoundFeatureMembership\_Mapping

##### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

##### General Mappings

DefaultMultiplicityBoundFeatureMembership\_Mapping

##### Mapping Source

Element

##### Mapping Target

FeatureMembership

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : MultiplicityRange [1]

```
DefaultUpperBound_Mapping.getMapped(from)
```

#### 7.7.4.2.9 DefaultUpperBound\_Mapping

##### Description

The mapping class creates the default upper bound of a multiplicity element.

##### General Mappings

GenericToExpression\_Mapping

##### Mapping Source

Element

##### Mapping Target

LiteralInteger

##### Owned Mappings

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralInteger::value () : Integer [1]

1

- LiteralInteger::ownedRelationship () : Relationship [0..\*]

```
Set{CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
```

### **7.7.4.2.10 DefaultValue\_Mapping**

#### **Description**

The expected SysML v2 textual syntax of a mapped SysML v2 default value is as follows:

```
attribute sysMLv1Property : ScalarValues::String default := "default value";
```

#### **General Mappings**

GenericToFeatureValue\_Mapping

#### **Mapping Source**

Property

#### **Mapping Target**

FeatureValue

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::isDefault () : Boolean [1]

true

- FeatureValue::value () : Expression [1]

```
from.defaultValue
```

### **7.7.4.2.11 ElementFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

Element

#### **Mapping Target**

FeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
NamedElementMain_Mapping.getMapped(from)
```

- FeatureMembership::visibility () : VisibilityKind [1]

```
if from.oclIsKindOf(UML::NamedElement) then
 Helper.getKerMLVisibilityKind(from.oclAsType(UML::NamedElement).visibility)
else KerML::VisibilityKind::public endif
```

### **7.7.4.2.12 Generalization\_Mapping**

#### **Description**

A UML4SysML::Generalization relationship is mapped to a SysML v2 Subclassification.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1BlockGeneral;
part def SysMLv1BlockSpecial :> SysMLv1BlockGeneral;
```

## **General Mappings**

GenericToSpecialization\_Mapping  
ElementMain\_Mapping

### **Mapping Source**

Generalization

### **Mapping Target**

Subclassification

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subclassification::superclassifier () : Classifier [1]

```
if from.general.oclIsTypeOf(UML::PrimitiveType)
 and not (Helper.getScalarValueType(from.general)
 = invalid) then
 Helper.getScalarValueType(from.general)
else
 Classifier_Mapping.getMapped(from.general)
endif
```

- Subclassification::subclassifer () : Classifier [1]

```
Classifier_Mapping.getMapped(from.specific)
```

### **7.7.4.2.13 InstanceSpecificationLink\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

### **Description**

The UML4SysML::InstanceSpecification that is a link is mapped to a SysMLv2 ConnectionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block1;
part def SysMLv1Block2;
connection def SysMLv1Association {
```

```

 end : SysMLv1Block1[1];
 end : SysMLv1Block2[1];
 }
part sysMLv1InstanceSpecification1 : SysMLv1Block1;
part sysMLv1InstanceSpecification2 : SysMLv1Block2;
connection sysMLv1Link : SysMLv1Association
 connect sysMLv1InstanceSpecification1 to sysMLv1InstanceSpecification2;

```

## General Mappings

NamedElementMain\_Mapping  
GenericToConnectionUsage\_Mapping

### Mapping Source

InstanceSpecification

### Mapping Target

ConnectionUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.classifier->select(c | c.oclIsTypeOf(UML::Association))->size() > 0
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConnectionUsage::ownedRelationship () : Relationship [0..\*]

```

self.oclAsType(ElementMain_Mapping).ownedRelationship()
->union(SlotMembership_Mapping.getMappedColl(from.slot)->asSet())
->union(from.classifier
->collect(g | InstanceSpecificationFeatureTyping_Mapping.getMapped(from, g))->asSet())
->asSet()

```

### 7.7.4.2.14 InstanceSpecification\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**  
**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

### Description

The UML4SysML::InstanceSpecification that is not a link is mapped to a SysMLv2 PartDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block {
 attribute sysMLv1ValueProperty : ScalarValues::String;
}

part sysMLv1InstanceSpecification : SysMLv1Block {
 redefines sysMLv1ValueProperty = "Hello InstanceSpecification";
}
```

## General Mappings

NamedElementMain\_Mapping  
GenericToPartUsage\_Mapping

### Mapping Source

InstanceSpecification

### Mapping Target

PartUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.classifier->select(c | c.oclIsTypeOf(UML::Association))->size() = 0
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PartUsage::ownedRelationship () : Relationship [0..\*]

```
SlotMembership_Mapping.getMappedColl(from.slot)->asSet()
->union(from.classifier
->collect(g | InstanceSpecificationFeatureTyping_Mapping.getMapped(from, g))->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
->asSet()
```

- PartUsage::ownedFeatureMembership () : FeatureMembership [0..\*]

```
from.classifier
->collect(c | InstanceSpecificationToGeneralization_Mapping.getMapped(from, c))
```

## 7.7.4.2.15 InstanceSpecificationFeatureTyping\_Mapping

### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

InstanceSpecification

#### Mapping Target

FeatureTyping with qualifier: classifier:Classifier

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type (in classifier : Classifier) : Type [1]

    Classifier\_Mapping.getMapped(classifier)

### 7.7.4.2.16 InstanceValue\_Mapping

#### Description

The UML4SysML::InstanceValue is mapped to a SysMLv2 FeatureReferenceExpression.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block1;
part sysMLv1InstanceSpecification : SysMLv1Block1;
part def SysMLv1Block2 {
 part sysMLv1PartProperty : SysMLv1Block1
 = sysMLv1InstanceSpecification;
}
```

### General Mappings

ValueSpecification\_Mapping

#### Mapping Source

InstanceValue

### **Mapping Target**

FeatureReferenceExpression

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->including(InstanceValueMembership_Mapping.getMapped(from.instance))
->including(ReturnParameterFeatureMembership_Factory.create())
```

## **7.7.4.2.17 InstanceValueMembership\_Mapping**

### **Description**

Creates a membership relationship for *memberElement()*.

### **General Mappings**

GenericToMembership\_Mapping

### **Mapping Source**

InstanceSpecification

### **Mapping Target**

Membership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

from

#### 7.7.4.2.18 LowerBoundValueFeatureMembership\_Mapping

##### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

##### General Mappings

GenericToFeatureMembership\_Mapping

##### Mapping Source

MultiplicityElement

##### Mapping Target

FeatureMembership

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]

```
LiteralInteger_Mapping.getMapped(from.lowerValue)
```

#### 7.7.4.2.19 MultiplicityElement\_Mapping

##### Description

A UML4SysML::MultiplicityElement is mapped to a SysML v2 MultiplicityRange.

##### General Mappings

GenericToFeature\_Mapping

##### Mapping Source

MultiplicityElement

##### Mapping Target

MultiplicityRange

##### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MultiplicityRange::declaredName () : String [0..1]  
`'multiplicity'`
- MultiplicityRange::ownedRelationship () : Relationship [0..\*]  
`OrderedSet{MultiplicityLowerBoundOwningMembership_Mapping.getMapped(from), MultiplicityUpperBoundOwningMembership_Mapping.getMapped(from)}`
- MultiplicityRange::isUnique () : Boolean [1]  
`from.isUnique`

### 7.7.4.2.20 MultiplicityLowerBoundOwningMembership\_Mapping

#### Description

Creates a owning membership relationship for *ownedMemberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

MultiplicityElement

#### Mapping Target

OwningMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```

if from.lowerValue.oclIsUndefined() then
 DefaultLowerBound_Mapping.getMapped(from)
else
 from.lowerValue
endif

• OwningMembership::memberName () : String [0..1]

'lowerBound'

```

### **7.7.4.2.21 MultiplicityMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

GenericToOwningMembership\_Mapping

#### **Mapping Source**

MultiplicityElement

#### **Mapping Target**

OwningMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

MultiplicityElement\_Mapping.getMapped(from)

### **7.7.4.2.22 MultiplicityUpperBoundOwningMembership\_Mapping**

#### **Description**

Creates a owning membership relationship for *ownedMemberElement()*.

#### **General Mappings**

GenericToOwningMembership\_Mapping

#### **Mapping Source**

MultiplicityElement

#### Mapping Target

OwningMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
if from.upperValue.oclIsUndefined() then
 DefaultUpperBound_Mapping.getMapped(from)
else
 from.upperValue
endif
```

- OwningMembership::memberName () : String [0..1]

```
'upperBound'
```

### 7.7.4.2.23 Operation\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A UML4SysML::Operation is mapped to a SysML v2 PerformActionUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block {
 perform action sysMLv1Operation {
 in parIn : ScalarValues::Boolean;
 out result : ScalarValues::Integer;
 }
}
```

#### General Mappings

BehavioralFeature\_Mapping  
GenericToActionUsage\_Mapping

#### Mapping Source

Operation

**Mapping Target**

PerformActionUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PerformActionUsage::ownedRelationship () : Relationship [0..\*]

```
let parameters: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Parameter)) in
let parameterSets: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::ParameterSet)) in
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->union(parameters->collect(e | ParameterMembership_Mapping.getMapped(e))->asSet())
->union(parameterSets->collect(e | ParameterSetMembership_Mapping.getMapped(e))->asSet())
```

#### 7.7.4.2.24 Parameter\_Mapping

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

**Description**

A UML4SysML::Parameter is mapped to a SysML v2 ReferenceUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 in parIn : ScalarValues::Boolean;
}
```

**General Mappings**

GenericToReferenceUsage\_Mapping  
NamedElementMain\_Mapping

**Mapping Source**

Parameter

**Mapping Target**

## ReferenceUsage

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
 Helper.getKerMLParameterDirectionKind(from.direction)
```

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
let typings: Set(KerML::FeatureTyping) =
 if from.type.oclIsUndefined() then
 Set{}
 else
 Set{ParameterToFeatureTyping_Mapping.getMapped(from)}
 endif in
let multiplicities: Set(KerML::Relationship) =
 Set{MultiplicityMembership_Mapping.getMapped(from)} in
let defaultValues: Set(KerML::Relationship) =
 if from.defaultValue.oclIsUndefined() then
 Set{}
 else
 Set{ParameterDefaultValue_Mapping.getMapped(from)}
 endif in
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->union(typings)
->union(multiplicities)
->union(defaultValues)
```

- ReferenceUsage::declaredName () : String [0..1]

```
 if from.direction = UML::ParameterDirectionKind::return then 'result' else from.name endif
```

### 7.7.4.2.25 ParameterDefaultValue\_Mapping

#### Description

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
attribute value : ScalarValues::String default := "default value";
```

### General Mappings

#### GenericToFeatureValue\_Mapping

**Mapping Source**

Parameter

**Mapping Target**

FeatureValue

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
    from.defaultValue
- FeatureValue::isDefault () : Boolean [1]  
    true

### 7.7.4.2.26 ParameterMembership\_Mapping

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToParameterMembership\_Mapping

**Mapping Source**

Parameter

**Mapping Target**

ParameterMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]

```
Parameter_Mapping.getMapped(from)
```

#### 7.7.4.2.27 ParameterSet\_Mapping

##### Description

A UML4SysML::ParameterSet is mapped to a SysML v2 ReferenceUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 in parIn [0..1];
 inout parInOut [0..1];
 out parOut [0..1];
 out parReturn [0..1];

 sysMLv1ParameterSet1 [1] {
 ref parIn = SysMLv1Activity::parIn;
 assert constraint sysMLv1ParameterSet1Condition {
 language "English"
 /*
 * opaque expression parameter set 1
 */
 }
 }
 sysMLv1ParameterSet2 [1] {
 ref parInOut = SysMLv1Activity::parInOut;
 ref parOut = SysMLv1Activity::parOut;
 ref parReturn = SysMLv1Activity::parReturn;
 }
}
```

##### General Mappings

GenericToReferenceUsage\_Mapping

##### Mapping Source

ParameterSet

##### Mapping Target

ReferenceUsage

##### Owned Mappings

(none)

##### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
from.parameter
->collect(p | ParameterSetParameterFeatureMembership_Mapping.getMapped(from, p))
->asSet()
```
- ReferenceUsage::declaredName () : String [0..1]

```
from.name
```

### 7.7.4.2.28 ParameterSetMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

ParameterSet

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ParameterSet_Mapping.getMapped(from)
```

### 7.7.4.2.29 ParameterSetParameterFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

ParameterSet

### **Mapping Target**

FeatureMembership with qualifier: parameter:Parameter

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature (in parameter : Parameter) : Feature [1]

ParameterSetParameterReferenceUsage\_Mapping.getMapped(parameter)

## **7.7.4.2.30 ParameterSetParameterReferenceUsage\_Mapping**

### **Description**

The mapping class creates the reference usage element for the UML4SysML::ParameterSet mapping.

## **General Mappings**

GenericToReferenceUsage\_Mapping

### **Mapping Source**

Parameter

### **Mapping Target**

ReferenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ParameterSetParameterReferenceUsageFeatureValue_Mapping.getMapped(from),
MultiplicityMembership_Mapping.getMapped(from)}
```

#### 7.7.4.2.31 ParameterSetParameterReferenceUsageFeatureValue\_Mapping

##### Description

The mapping class creates the feature reference expression for the reference usage element of the UML4SysML::ParameterSet mapping.

##### General Mappings

GenericToFeatureValue\_Mapping

##### Mapping Source

Parameter

##### Mapping Target

FeatureValue

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
ParameterSetParameterReferenceUsageFeatureValueExpression_Mapping.getMapped(from)
```

#### 7.7.4.2.32 ParameterSetParameterReferenceUsageFeatureValueExpression\_Mapping

##### Description

The mapping class creates the feature reference expression for the UML4SysML::ParameterSet mapping.

##### General Mappings

GenericToFeatureReferenceExpression\_Mapping

##### Mapping Source

Parameter

**Mapping Target**

FeatureReferenceExpression

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..\*]

```
Set{ParameterSetParameterReferenceUsageMembership_Mapping.getMapped(from),
CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
```

**7.7.4.2.33 ParameterSetParameterReferenceUsageMembership\_Mapping**

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToMembership\_Mapping

**Mapping Source**

Parameter

**Mapping Target**

Membership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

```
from
```

#### 7.7.4.2.34 ParameterToFeatureTyping\_Mapping

##### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

##### General Mappings

TypedElementFeatureTyping\_Mapping

##### Mapping Source

Parameter

##### Mapping Target

FeatureTyping

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::typedFeature () : Feature [1]

```
parameter.to
```

#### 7.7.4.2.35 PropertyCommon\_Mapping

##### Description

The mapping class is the abstract base class for UML4SysML::Property mappings.

##### General Mappings

StructuralFeature\_Mapping

##### Mapping Source

Property

##### Mapping Target

Feature

##### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]

```
if from.association.oclIsUndefined() then
 false
else
 from.association.ownedEnd->includes(from)
endif
```

- Feature::isComposite () : Boolean [1]

```
from.isComposite
```

- Feature::ownedRelationship () : Relationship [0..\*]

```
let typings: Set(KerML::FeatureTyping) = if from.type.oclIsUndefined() then
 Set{}
else
 Set{StructuralFeatureToFeatureTyping_Mapping.getMapped(from)}
endif in
let subsettings: Set(KerML::Subsetting) = from.subsettedProperty
->collect(p | PropertySubsetting_Mapping.getMapped(from, p))->asSet() in
let defaultValue: Set(KerML::OwningMembership) =
 if from.defaultValue.oclIsUndefined() then
 Set{}
 else
 Set{DefaultValue_Mapping.getMapped(from)}
 endif in
 typings->union(subsettings)->union(defaultValue)
->including(MultiplicityMembership_Mapping.getMapped(from))->asSet()
```

- Feature::isDerived () : Boolean [1]

```
from.isDerived
```

#### 7.7.4.2.36 PropertySubsetting\_Mapping

**SYSML2-200: Description of Subsetting mapping classes is not correct**

##### Description

Creates a subsetting relationship.

##### General Mappings

GenericToSubsetting\_Mapping

### **Mapping Source**

Property

### **Mapping Target**

Subsetting with qualifier: subsettetedProperty:Property

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subsetting::subsettetedFeature (in subsettetedProperty : Property) : Feature [1]  
    Property\_Mapping.getMapped(subsettetedProperty)
- Subsetting::subsettingFeature () : Feature [1]  
    Property\_Mapping.getMapped(from)

### **7.7.4.2.37 PropertyTypedByClassInterface\_Mapping**

**SYSML2-443: Property\_Mapping should map to ItemUsage and the class name is misleading**  
**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### **Description**

A UML4SysML::Property typed by a UML4SysML::Class or UML4SysML::Interface is mapped to a SysML v2 OccurrenceUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block {
 occurrence sysMLv1Property1 [0..1] : SysMLv1Class;
 ref occurrence sysMLv1ReferencedProperty [0..1] : SysMLv1Class;
 occurrence sysMLv1Property2 [0..1] : SysMLv1Interface;
}
```

### **General Mappings**

PropertyCommon\_Mapping  
NamedElementMain\_Mapping

### **Mapping Source**

Property

### Mapping Target

OccurrenceUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
if src.oclIsTypeOf(UML::Property) then
 let p: UML::Property = src.oclAsType(UML::Property) in
 if p.type.oclIsUndefined() then
 false
 else
 (p.type.oclIsTypeOf(UML::Class) or
 p.type.oclIsTypeOf(UML::Interface)) and
 not (p.name.indexOf('base_') > 0) and
 (p.association.oclIsUndefined() or p.association.ownedEnd->excludes(p))
 endif
 else
 false
 endif
```

### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

#### 7.7.4.2.38 PropertyUntyped\_Mapping

##### [SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)

#### Description

A UML4SysML::Property is mapped to a SysML v2 Feature. The mapping class maps properties without a type.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block {
 attribute sysMLv1Property;
}
```

### General Mappings

PropertyCommon\_Mapping  
GenericToReferenceUsage\_Mapping  
NamedElementMain\_Mapping

### Mapping Source

Property

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

**Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.type.oclIsUndefined() and not
Helper.hasStereotypeApplied(src.owner, 'SysML::ConstraintBlocks::ConstraintBlock')
```

**Mapping rules**

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### **7.7.4.2.39 Realization\_Mapping**

**Description**

A UML4SysML::Realization relationship is mapped to a SysML v2 Dependency.

**General Mappings**

Abstraction\_Mapping

**Mapping Source**

Realization

**Mapping Target**

Dependency

**Owned Mappings**

(none)

### **7.7.4.2.40 Slot\_Mapping**

**Description**

A UML4SysML::Slot is mapped to a SysML v2 Feature.

**General Mappings**

GenericToFeature\_Mapping

ElementMain\_Mapping

**Mapping Source**

Slot

**Mapping Target**

Feature

**Owned Mappings**

(none)

**7.7.4.2.41 SlotMembership\_Mapping****Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToFeatureMembership\_Mapping

**Mapping Source**

Slot

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::memberName () : String [0..1]  

```
from.definingFeature.name
```
- FeatureMembership::ownedMemberFeature () : Feature [1]  

```
from
```
- FeatureMembership::isReadOnly () : Boolean [1]  

```
from.isReadOnly
```

**7.7.4.2.42 SlotFeatureTyping\_Mapping**

## Description

Creates a feature typing relationship owned by the element *typedFeature()*.

## General Mappings

GenericToFeatureTyping\_Mapping

### Mapping Source

Slot

### Mapping Target

FeatureTyping

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
ElementMain_Mapping.getMapped(from)
```

## 7.7.4.2.43 SlotValue\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

## Description

Issue here since a KerML feature cannot have more than one FeatureValue while a UML4SysML::Slot can. How to manage collection of values?

## General Mappings

GenericToFeatureValue\_Mapping

### Mapping Source

ValueSpecification

### Mapping Target

FeatureValue

## Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
src.owner.oclIsKindOf(UML::Slot)
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::featureWithValue () : Feature [1]  

```
Slot_Mapping.getMapped(from.owner)
```
- FeatureValue::value () : Expression [1]  

```
from
```

## 7.7.4.2.44 StructuralFeature\_Mapping

### Description

The mapping class is the abstract base class for all UML4SysML::StructuralFeature mappings.

### General Mappings

GenericToFeature\_Mapping

### Mapping Source

StructuralFeature

### Mapping Target

Feature

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isUnique () : Boolean [1]

```
from.isUnique
```

- Feature::isAbstract () : Boolean [1]
 

```
false
```
- Feature::ownedRelationship () : Relationship [0..\*]
 

```
let typing: KerML::FeatureTyping =
 StructuralFeatureToFeatureTyping_Mapping.getMapped(from) in
if typing.oclIsUndefined() then
 Set{MultiplicityMembership_Mapping.getMapped(from)}
else
 Set{MultiplicityMembership_Mapping.getMapped(from), typing}
endif
```
- Feature::isOrdered () : Boolean [1]
 

```
from.isOrdered
```
- Feature::isReadOnly () : Boolean [1]
 *abstract rule*

#### **7.7.4.2.45 StructuralFeatureMembership\_Mapping**

##### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

##### **General Mappings**

GenericToFeatureMembership\_Mapping

##### **Mapping Source**

StructuralFeature

##### **Mapping Target**

FeatureMembership

##### **Owned Mappings**

(none)

##### **Applicable filters**

(none)

##### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::visibility () : VisibilityKind [1]
 

```
if (from.oclIsKindOf(UML::NamedElement)) then
 Helper.getKerMLVisibilityKind(from.oclAsType(UML::NamedElement).visibility)
else
```

```

 KerML::VisibilityKind::public
endif

• FeatureMembership::ownedMemberFeature () : Feature [0..1]

 NamedElementMain_Mapping.getMapped(from)

```

#### **7.7.4.2.46 StructuralFeatureToFeatureTyping\_Mapping**

##### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

##### **General Mappings**

TypedElementFeatureTyping\_Mapping

##### **Mapping Source**

StructuralFeature

##### **Mapping Target**

FeatureTyping

##### **Owned Mappings**

(none)

#### **7.7.4.2.47 TypedElementFeatureTyping\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

##### **General Mappings**

GenericToFeatureTyping\_Mapping

##### **Mapping Source**

TypedElement

##### **Mapping Target**

FeatureTyping

##### **Owned Mappings**

(none)

##### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
not src.type.oclIsUndefined()
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if from.type.oclIsKindOf(UML::PrimitiveType) then
 Helper.getScalarValueType(from.type)
else if from.type.oclIsKindOf(UML::Enumeration) then
 Helper.getEnumerationType(from.type)
else
 Classifier_Mapping.getMapped(from.type)
endif endif
```

## 7.7.4.2.48 UpperBoundValueFeatureMembership\_Mapping

### Description

Creates a feature membership relationship for `ownedMemberFeature()`.

### General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

MultiplicityElement

### Mapping Target

FeatureMembership

### Owned Mappings

(none)

### Applicable filters

(none)

This chapter lists all mapping specifications of UML4SysML::Classification model elements.

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]

```
if from.upper <> -1 then
 LiteralUnlimitedToInteger_Mapping.getMapped(from.upperValue)
```

```

else
 LiteralUnlimitedToUnbounded_Mapping.getMapped(from.upperValue)
endif

```

## 7.7.5 CommonBehavior

This chapter lists all mapping specifications of UML4SysML::CommonBehavior model elements.

**SYSML2-513:** Missing text in some main mapping sections

### 7.7.5.1 Overview

**SYSML2-441:** Change the table header of the overview tables in the mapping class specification chapters

**Table 6. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax                               |
|-------------------------------------|--------------------------------------------------------|
| AnyReceiveEvent                     | not mapped; see next section                           |
| CallEvent                           | not mapped; see next section                           |
| ChangeEvent                         | TextualRepresentation                                  |
| FunctionBehavior                    | ViewDefinition<br>RequirementUsage                     |
| OpaqueBehavior                      | ViewDefinition<br>ActionDefinition<br>RequirementUsage |
| SignalEvent                         | not mapped; see next section                           |
| TimeEvent                           | TextualRepresentation                                  |
| Trigger                             | AcceptActionUsage                                      |

The following table gives an overview of which SysML v2 elements the UML4SysML::CommonBehavior elements are transformed with which mapping class. The mapping details are in [7.7.5.3](#).

The justifications for the elements without mapping are given in [7.7.5.2](#).

### 7.7.5.2 UML4SysML::CommonBehavior elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566:** Section containing tables about elements not mapped should get an introductory text

**Table 7. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept | Rationale                                                |
|------------------|----------------------------------------------------------|
| CallEvent        | The concept of a CallEvent is not supported by SysML v2. |

### 7.7.5.3 Mapping Specifications

### 7.7.5.3.1 Behavior\_Mapping

**SYSML2-202: Filter for mapping class Behavior\_Mapping is useless**  
**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

The mapping class is the abstract base class for all UML4SysML::Behavior mappings.

#### General Mappings

GenericToBehavior\_Mapping  
Class\_Mapping

#### Mapping Source

Behavior

#### Mapping Target

Behavior

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Behavior::ownedRelationship () : Relationship [0..\*]

```
let parameters: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::Parameter)) in
let parameterSets: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::ParameterSet)) in
let features: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::Property)) in
let elementsOMS: Set(UML::Element) =
 (((from.ownedElement - parameters) parameterSets) - features) in
elementsOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(features->collect(e | PropertyMembership_Mapping.getMapped(e)))
->union(parameters->collect(e | ParameterMembership_Mapping.getMapped(e)))
->union(parameterSets->collect(e | ParameterSetMembership_Mapping.getMapped(e)))
```

### 7.7.5.3.2 ChangeEvent\_Mapping

#### Description

T#3 meeting, 2022-12-14: Do not use automatic rules! Events are not single elements in SysML v2. Consider it in the transformation for AcceptEventAction, Transition

## **General Mappings**

GenericToTextualRepresentation\_Mapping  
NamedElementMain\_Mapping

### **Mapping Source**

ChangeEvent

### **Mapping Target**

TextualRepresentation

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TextualRepresentation::body () : String [1]

```
if from.changeExpression.oclIsKindOf(UML::OpaqueExpression) then
 if from.changeExpression.
 oclAsType(UML::OpaqueExpression).body.oclIsUndefined() then
 invalid
 else
 from.changeExpression.oclAsType(UML::OpaqueExpression).body.get(0)
 endif
 else
 invalid
 endif
```

- TextualRepresentation::language () : String [1]

```
if from.changeExpression.oclIsKindOf(UML::OpaqueExpression) then
 if from.changeExpression.
 oclAsType(UML::OpaqueExpression).language->size() = 0 then
 invalid
 else
 from.changeExpression.oclAsType(UML::OpaqueExpression).language.get(0)
 endif
 else
 invalid
 endif
```

### **7.7.5.3.3 OpaqueBehavior\_Mapping**

**SYSML2-202: Filter for mapping class Behavior\_Mapping is useless**  
**SYSML2-7: Pin\_Mapping::filter: property src should be from**

## **SYSML2-221: UML4SysML::Activities and StateMachines owned by blocks should be mapped to definition elements**

### **Description**

A UML4SysML::OpaqueBehavior is mapped to a SysML v2 ActionDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1OpaqueBehavior {
 language "Built-in Math"
 /*
 * result = 42 + 23;
 */
}
```

### **General Mappings**

Behavior\_Mapping

#### **Mapping Source**

OpaqueBehavior

#### **Mapping Target**

ActionDefinition

#### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.owner.oclIsKindOf(UML::Package)
```

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionDefinition::ownedRelationship () : Relationship [0..\*]

```
let parameters : Set(UML::Parameter) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Parameter)) in
let parameterSets : Set(UML::ParameterSet) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::ParameterSet)) in
let features : Set(UML::Property) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Property)) in
let elementsOMS: Set(UML::Element) =
 ((from.ownedElement - parameters) - parameterSets) - features) in
```

```

elementsOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(features->collect(e | PropertyMembership_Mapping.getMapped(e)))
->union(parameters->collect(e | ParameterMembership_Mapping.getMapped(e)))
->union(parameterSets->collect(e | ParameterSetMembership_Mapping.getMapped(e)))
->union(from.language
 ->collect(l | OpaqueBehaviorMembership_Mapping.getMapped(from, l)))

```

#### **7.7.5.3.4 OpaqueBehaviorMembership\_Mapping**

##### **Description**

Creates a membership relationship for *memberElement()*.

##### **General Mappings**

GenericToOwningMembership\_Mapping

##### **Mapping Source**

OpaqueBehavior

##### **Mapping Target**

OwningMembership with qualifier: language:String

##### **Owned Mappings**

(none)

##### **Applicable filters**

(none)

##### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement (in language : String) : Element [1]
   
OpaqueBehaviorSpecification\_Mapping.getMapped(from, language)

#### **7.7.5.3.5 OpaqueBehaviorSpecification\_Mapping**

##### **Description**

The mapping class creates the SysML v2 TextualRepresentation elements from the languages and bodies properties of the given UML4SysML::OpaqueBehavior.

##### **General Mappings**

GenericToTextualRepresentation\_Mapping

##### **Mapping Source**

OpaqueBehavior

### **Mapping Target**

TextualRepresentation with qualifier: language:String

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TextualRepresentation::body () : String [1]  

```
let index:Integer = from.language->indexOf(language) in
from._'body'->at(index)
```
- TextualRepresentation::language () : String [1]  

```
language
```

## **7.7.5.3.6 TimeEvent\_Mapping**

### **Description**

T#3 meeting, 2022-12-14: Do not use automatic rules! Events are not single elements in SysML v2. Consider it in the transformation for AcceptEventAction, Transition

### **General Mappings**

NamedElementMain\_Mapping  
GenericToTextualRepresentation\_Mapping

### **Mapping Source**

TimeEvent

### **Mapping Target**

TextualRepresentation

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TextualRepresentation::body () : String [1]  
`'tbd timeevent'`

### 7.7.5.3.7 Trigger\_Mapping

## 7.7.6 CommonStructure

This chapter lists all mapping specifications of UML4SysML::CommonStructure model elements.

**SYSML2-513: Missing text in some main mapping sections**

### 7.7.6.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column**

**Table 9. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax                        |
|-------------------------------------|-------------------------------------------------|
| Abstraction                         | SatisfyRequirementUsage<br>AllocationDefinition |
| Comment                             | Package                                         |
| Constraint                          | ConstraintDefinition                            |
| Dependency                          | Dependency                                      |
| ElementImport                       | MembershipImport                                |
| PackageImport                       | NamespaceImport                                 |
| Realization                         | Dependency                                      |
| Usage                               | Dependency                                      |

The following table gives an overview of which SysML v2 elements the UML4SysML::CommonStructure elements are transformed with which mapping class. The mapping details are in [7.7.6.2](#).

### 7.7.6.2 Mapping Specifications

#### 7.7.6.2.1 Abstraction\_Mapping

##### Description

A UML4SysML::Abstraction relationship is mapped to a SysML v2 Dependency relationship.

##### General Mappings

Dependency\_Mapping

##### Mapping Source

Abstraction

#### **Mapping Target**

Dependency

#### **Owned Mappings**

(none)

#### **7.7.6.2 Comment\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**  
**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### **Description**

A UML4SysML::Comment is mapped to a SysML v2 Comment.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block1;
part def SysMLv1Block2;
action def SysMLv1Activitiy {
 comment about SysMLv1Activity, SysMLv1Block1
 /* comment body */
}
comment about SysMLv1Block1, SysMLv1Block /* comment body */
```

#### **General Mappings**

ElementMain\_Mapping  
GenericToAnnotatingElement\_Mapping

#### **Mapping Source**

Comment

#### **Mapping Target**

Comment

#### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
not Helper.hasStereotypeApplied(src, 'SysML::ModelElements::ElementGroup')
```

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Comment::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->union(self.annotation()->asSet())
```
- Comment::body () : String [1]

```
if from.body->isEmpty() then '' else from.body endif
```
- Comment::annotation () : Annotation [0..\*]

```
from.annotatedElement
->collect(e | CommentAnnotation_Mapping.getMapped(from, e))
```

### 7.7.6.2.3 CommentAnnotation\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### **Description**

The mapping class creates the annotation relationship for the UML4SysML::Comment mapping.

##### **General Mappings**

GenericToAnnotation\_Mapping

##### **Mapping Source**

Comment

##### **Mapping Target**

Annotation with qualifier: annotatedElement:Element

##### **Owned Mappings**

(none)

##### **Applicable filters**

(none)

##### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Annotation::annotatedElement (in annotatedElement : Element) : Element [1]

```
ElementMain_Mapping.getMapped(annotatedElement)
```
- Annotation::annotatingElement () : AnnotatingElement [1]

```
Comment_Mapping.getMapped(from)
```

- Annotation::owningAnnotatedElement () : Element [0..1]

    null

#### **7.7.6.2.4 CommentOwnership\_Mapping**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### **Description**

That mapping class creates an ownership relation that is convenient for a Comment. In SysMLv1/UML can be owned by any kind of element, including some that are not translated to SysMLv2 Namespaces.

##### **General Mappings**

GenericToAnnotation\_Mapping  
UniqueMapping

##### **Mapping Source**

Comment

##### **Mapping Target**

Annotation

##### **Owned Mappings**

(none)

##### **Applicable filters**

(none)

##### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Annotation::annotatedElement () : Element [1]
 

```
ElementMain_Mapping.getMapped(from.owner)
```
- Annotation::annotatingElement () : AnnotatingElement [1]
 

```
Comment_Mapping.getMapped(from)
```
- Annotation::ownedRelatedElement () : Element [0..\*]
 

```
Set{self.annotatingElement()}
```

#### **7.7.6.2.5 Constraint\_Mapping**

##### **Description**

A UML4SysML::Constraint is mapped to a SysML v2 ConstraintDefinition and AssertConstraintUsages for the constrained elements.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block {
 constraint def SysMLv1Constraint {
 calc sysMLv1Constraint {
 language "English"
 /*
 * constraint specification
 */
 }
 }
 assert constraint assert_sysMLv1Constraint : SysMLv1Constraint;
}
```

## General Mappings

GenericToConstraintDefinition\_Mapping  
NamedElementMain\_Mapping

### Mapping Source

Constraint

### Mapping Target

ConstraintDefinition

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConstraintDefinition::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->union(Set{ElementFeatureMembership_Mapping.getMapped(from.specification),
CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from.specification) })
```

## 7.7.6.2.6 ConstrainedElementFeatureMembership\_Mapping

### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

Constraint

### **Mapping Target**

FeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

    ConstraintUsage\_Mapping.getMapped(from)

## **7.7.6.2.7 ConstraintUsageFeatureTyping\_Mapping**

### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

## **General Mappings**

GenericToFeatureTyping\_Mapping

### **Mapping Source**

Constraint

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

from

### 7.7.6.2.8 ConstraintUsage\_Mapping

#### Description

The mapping class creates the SysML v2 AssertConstraintUsage elements for the constrained elements of the UML4SysML::Constraint mapping.

#### General Mappings

GenericToUsage\_Mapping

#### Mapping Source

Constraint

#### Mapping Target

AssertConstraintUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- AssertConstraintUsage::declaredName () : String [0..1]

'assert\_' + from.name

- AssertConstraintUsage::ownedRelationship () : Relationship [0..\*]

```
from.ownedComment->reject(c | c.annotatedElement->includes(from))>>collect(c| CommentOwnership->union(Set{ConstraintUsageFeatureTyping_Mapping.getMapped(from), CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from)}))
```

### 7.7.6.2.9 Dependency\_Mapping

#### Description

A UML4SysML::Dependency relationship is mapped to a SysML v2 Dependency relationship.

#### General Mappings

## DirectedRelationship\_Mapping

### Mapping Source

Dependency

### Mapping Target

Dependency

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Dependency::client () : Element [0..\*]  

```
from.source->collect(e | ElementMain_Mapping.getMapped(e))
```
- Dependency::declaredName () : String [0..1]  

```
from.name
```
- Dependency::supplier () : Element [0..\*]  

```
from.target->collect(e | ElementMain_Mapping.getMapped(e))
```

## 7.7.6.2.10 DirectedRelationship\_Mapping

### Description

The mapping class is the abstract base class for all UML4SysML::DirectedRelationship mappings.

### General Mappings

Relationship\_Mapping

### Mapping Source

DirectedRelationship

### Mapping Target

Relationship

### Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Relationship::target () : Element [0..\*]

```
from.target->collect(e | ElementMain_Mapping.getMapped(e))
```

- Relationship::source () : Element [0..\*]

```
from.source->collect(e | ElementMain_Mapping.getMapped(e))
```

## 7.7.6.2.11 ElementMain\_Mapping

### [SYSML2-280: ElementMain\\_Mapping::ownedRelationship is wrong](#)

#### Description

This is the general abstract class to be used as an ancestor for any class mapping specification.

#### General Mappings

GenericToElement\_Mapping  
MainMapping

#### Mapping Source

Element

#### Mapping Target

Element

#### Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Element::ownedRelationship () : Relationship [0..\*]

```
from.ownedComment->reject(c | c.annotatedElement->includes(from))>collect(c| CommentOwnership)
```

- Element::elementId () : String [1]

```
Helper.getID(from)
```

### 7.7.6.12 ElementMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToMembership\_Mapping

#### Mapping Source

Element

#### Mapping Target

Membership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::visibility () : VisibilityKind [1]

```
if (from.oclIsKindOf(UML::NamedElement)) then
 from.oclAsType(UML::NamedElement).visibility
else
 KerML::VisibilityKind::public
endif
```
- Membership::membershipOwningNamespace () : Element [0..\*]

```
Set{ElementMain_Mapping(from)}
-- will not be used since corresponding attribute is derived,
-- but required for redefinition
```
- Membership::memberElement () : Element [1]

```
ElementMain_Mapping.getMapped(from)
```

### 7.7.6.13 ElementOwnership\_Mapping

#### Description

The mapping class is the abstract base class for mappings that target ownership relationships.

## **General Mappings**

GenericToRelationship\_Mapping  
UniqueMapping

### **Mapping Source**

Element

### **Mapping Target**

Relationship

## **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Relationship::target () : Element [0..\*]  

```
OrderedSet{ElementMain_Mapping.getMapped(from)}
```
- Relationship::source () : Element [0..\*]  

```
OrderedSet{ElementMain_Mapping.getMapped(from.owner)}
```
- Relationship::ownedRelatedElement () : Element [0..\*]  

```
self.target()
```

## **7.7.6.2.14 ElementOwningMembership\_Mapping**

### **Description**

Creates a owning membership relationship for *ownedMemberElement()*.

## **General Mappings**

ElementMembership\_Mapping  
ElementOwnership\_Mapping

### **Mapping Source**

Element

### **Mapping Target**

OwningMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedRelatedElement () : Element [0..\*]  

```
Set{self.ownedMemberElement () }
```
- OwningMembership::membershipOwningNamespace () : Element [0..\*]  

```
Set{ElementMain_Mapping(from) }
-- will not be used since corresponding attribute is derived,
-- but required for redefinition
```
- OwningMembership::ownedMemberElement () : Element [1]  

```
ElementMain_Mapping.getMapped(from)
```

## **7.7.6.2.15 NamedElementMain\_Mapping**

### **Description**

The mapping class is the abstract base class for mappings of UML4SysML::NamedElements.

### **General Mappings**

ElementMain\_Mapping

### **Mapping Source**

NamedElement

### **Mapping Target**

Element

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Element::declaredName () : String [0..1]

from.name

### 7.7.6.2.16 Namespace\_Mapping

#### Description

The mapping class is the abstract base class for UML4SysML::Namespace mappings.

#### General Mappings

GenericToNamespace\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

Namespace

#### Mapping Target

Namespace

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Namespace::ownedImport () : Import [0..\*]

Set {}

### 7.7.6.2.17 Relationship\_Mapping

#### Description

The mapping class is the abstract base class for UML4SysML::Relationship mappings.

#### General Mappings

GenericToRelationship\_Mapping  
ElementMain\_Mapping

#### Mapping Source

Relationship

**Mapping Target**

Relationship

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Relationship::ownedRelatedElement () : Element [0..\*]

```
from.relatedElement->select(e | from.ownedElement->includes(e))
->collect(e | ElementMain_Mapping.getMapped(e))
```
- Relationship::owningRelatedElement () : Element [0..1]

```
ElementMain_Mapping.getMapped(from.owner)
```

### **7.7.6.2.18 Usage\_Mapping**

**Description**

A UML4SysML::Usage relationship is mapped to a SysML v2 Dependency relationship.

**General Mappings**

Dependency\_Mapping

**Mapping Source**

Usage

**Mapping Target**

Dependency

**Owned Mappings**

(none)

## **7.7.7 InformationFlows**

This chapter lists all mapping specifications of UML4SysML::InformationFlows model elements.

**SYSML2-513: Missing text in some main mapping sections**

### 7.7.7.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**Table 10. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax |
|-------------------------------------|--------------------------|
| InformationFlow                     | FlowConnectionDefinition |
| InformationItem                     | ItemDefinition           |

The following table gives an overview of which SysML v2 elements the UML4SysML::InformationFlows elements are transformed with which mapping class. The mapping details are in [7.7.7.2](#).

### 7.7.7.2 Mapping Specifications

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

#### 7.7.7.2.1 InformationFlow\_Mapping

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### Description

A UML4SysML::InformationFlow is mapped to a FlowConnectionDefinition. If the UML4SysML::InformationFlow has defined realizingConnectors an additional FlowConnectionUsage element is created. The transformation rule is specified in the BehavioredClassifier::ownedRelationship operation. Then transformation also considers SysMLv1::ItemFlows which is handled by the factory class FlowConnectionUsage\_Factory.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block {
 part partA : SysMLv1BlockA;
 part partB : SysMLv1BlockB;
 part itemC : SysMLv1BlockC;

 connection sysMLv1Connector connect partA to partB;
 message : SysMLv1InformationFlowB :> sysMLv1Connector of itemC from partA to partB;
}

part def SysMLv1BlockA;
part def SysMLv1BlockB;
part def SysMLv1BlockC;
part def SysMLv1BlockD;

connection def SysMLv1Association {
 end : SysMLv1BlockA;
 end : SysMLv1BlockB;
}
```

```

flow def SysMLv1InformationFlowA :> SysMLv1Association {
 item : SysMLv1BlockC;
 item : SysMLv1BlockD;
}
flow def SysMLv1InformationFlowB {
 end partA : SysMLv1BlockA;
 end partB : SysMLv1BlockB;
}

```

## General Mappings

Relationship\_Mapping

### Mapping Source

InformationFlow

### Mapping Target

FlowConnectionDefinition

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FlowConnectionDefinition::ownedRelationship () : Relationship [0..\*]

```

from.source
 ->collect(s | InformationFlowEndFeatureMembership_Mapping.getMapped(from, s))->asSet()
->union(from.target
 ->collect(t | InformationFlowEndFeatureMembership_Mapping.getMapped(from, t))->asSet())
->union(from.conveyed
 ->collect(i | InformationFlowConveyedFeatureMembership_Mapping.getMapped(i))->asSet())
->union(from.realization->select(a | a.oclIsKindOf(UML::Association))
 ->collect(r | InformationFlowSubclassification_Mapping.getMapped(from, r))->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
->asOrderedSet()

```

## 7.7.7.2.2 InformationFlowConveyedFeatureMembership\_Mapping

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

Classifier

### **Mapping Target**

FeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

InformationItemFlowConveyedItemUsage\_Mapping.getMapped (from)

## **7.7.7.2.3 InformationFlowEnd\_Mapping**

**SYML2-420: InformationFlow mapping classes should use GenericTo mapping classes**

**SYML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

### **Description**

The mapping class creates the source feature of the FlowConnectionDefinition for the mapping of UML4SysML::InformationFlow.

## **General Mappings**

GenericToFeature\_Mapping

UniqueMapping

### **Mapping Source**

InformationFlow

### **Mapping Target**

Feature with qualifier: end:NamedElement

### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]  
true
- Feature::ownedRelationship () : Relationship [0..\*]  

```
Set<InformationFlowFeatureTyping_Mapping.getMapped(from, end) }
```

### **7.7.7.2.4 InformationFlowEndFeatureMembership\_Mapping**

**SYSML2-420: InformationFlow mapping classes should use GenericTo mapping classes**  
**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

#### **Description**

The mapping class creates the source and the target membership relationships of theFlowConnectionDefinition for the UML4SysML::InformationFlow mapping.

#### **General Mappings**

GenericToFeatureMembership\_Mapping  
UniqueMapping

#### **Mapping Source**

InformationFlow

#### **Mapping Target**

FeatureMembership with qualifier: end:NamedElement

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature (in end : NamedElement) : Feature [1]

```
InformationFlowEnd_Mapping.getMapped(from, end)
```

#### 7.7.7.2.5 InformationFlowFeatureTyping\_Mapping

**SYSML2-420: InformationFlow mapping classes should use GenericTo mapping classes**

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

##### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

##### General Mappings

GenericToFeatureTyping\_Mapping

UniqueMapping

##### Mapping Source

InformationFlow

##### Mapping Target

FeatureTyping with qualifier: element:NamedElement

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type (in source : NamedElement) : Type [1]

```
ElementMain_Mapping.getMapped(element)
```

#### 7.7.7.2.6 InformationFlowSubclassification\_Mapping

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

##### Description

Creates a Subclassification relationship between the target element of the UML4SysML::InformationFlow mapping and the target element of the UML4SysML::Association which realizes the flow.

### **General Mappings**

GenericToSubclassification\_Mapping

#### **Mapping Source**

InformationFlow

#### **Mapping Target**

Subclassification with qualifier: element:Relationship

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subclassification::subclassifer () : Classifier [1]  
    from  
    • Subclassification::superclassifier () : Classifier [1]  
        element

## **7.7.7.2.7 InformationItem\_Mapping**

### **Description**

A UML4SysML::InformationItem is mapped to a SysML v2 ItemDefinition.

### **General Mappings**

Classifier\_Mapping

#### **Mapping Source**

InformationItem

#### **Mapping Target**

ItemDefinition

### **Owned Mappings**

(none)

#### **7.7.7.2.8 InformationItemFlowConveyedItemUsage\_Mapping**

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

##### **Description**

Creates an ItemUsage element representing the conveyed classifier of an UML4SysML::InformationFlow.

##### **General Mappings**

GenericToItemUsage

##### **Mapping Source**

Classifier

##### **Mapping Target**

ItemUsage

##### **Owned Mappings**

(none)

##### **Applicable filters**

(none)

##### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ItemUsage::ownedRelationship () : Relationship [0..\*]

```
Set{InformationItemFlowConveyedItemUsageFeatureTyping_Mapping.getMapped(from)}
```

#### **7.7.7.2.9 InformationItemFlowConveyedItemUsageFeatureTyping\_Mapping**

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

##### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

##### **General Mappings**

GenericToFeatureTyping\_Mapping

##### **Mapping Source**

Classifier

**Mapping Target**

FeatureTyping

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

from

## 7.7.8 Interactions

This chapter lists all mapping specifications of UML4SysML::Interactions model elements.

**SYSML2-513: Missing text in some main mapping sections**

### 7.7.8.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

Table 11. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax                          |
|-------------------------------------|---------------------------------------------------|
| ActionExecutionSpecification        | ActionUsage                                       |
| BehaviorExecutionSpecification      | ActionUsage                                       |
| CombinedFragment                    | Interaction                                       |
| ConsiderIgnoreFragment              | not mapped; see next section                      |
| Continuation                        | not mapped; see next section                      |
| DestructionOccurrenceSpecification  | not mapped; see next section                      |
| ExecutionOccurrenceSpecification    | not mapped; see next section                      |
| Gate                                | not mapped; see next section                      |
| GeneralOrdering                     | not mapped; see next section                      |
| Interaction                         | ViewDefinition<br>Interaction<br>RequirementUsage |

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax     |
|-------------------------------------|------------------------------|
| InteractionConstraint               | not mapped; see next section |
| InteractionOperand                  | Interaction                  |
| InteractionUse                      | Step                         |
| Lifeline                            | PartUsage                    |
| Message                             | ItemFlow                     |
| MessageOccurrenceSpecification      | not mapped; see next section |
| OccurrenceSpecification             | not mapped; see next section |
| PartDecomposition                   | not mapped; see next section |
| StateInvariant                      | Invariant                    |

The following table gives an overview of which SysML v2 elements the UML4SysML::Interactions elements are transformed with which mapping class. The mapping details are in [7.7.8.3](#).

The justifications for the elements without mapping are given in [7.7.8.2](#).

### 7.7.8.2 UML4SysML::Interactions elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566: Section containing tables about elements not mapped should get an introductory text**

**Table 12. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept                   | Rationale                     |
|------------------------------------|-------------------------------|
| ConsiderIgnoreFragment             | Mapping is not specified yet. |
| Continuation                       | Mapping is not specified yet. |
| DestructionOccurrenceSpecification | Mapping is not specified yet. |
| ExecutionOccurrenceSpecification   | Mapping is not specified yet. |
| Gate                               | Mapping is not specified yet. |
| GeneralOrdering                    | Mapping is not specified yet. |
| InteractionConstraint              | Mapping is not specified yet. |
| MessageOccurrenceSpecification     | Mapping is not specified yet. |
| OccurrenceSpecification            | Mapping is not specified yet. |
| PartDecomposition                  | Mapping is not specified yet. |

### 7.7.8.3 Mapping Specifications

#### 7.7.8.3.1 ActionExecutionSpecification\_Mapping

##### Description

A UML4SysML::ActionExecutionSpecification is mapped to a SysML v2 ActionUsage.

### **General Mappings**

GenericToActionUsage\_Mapping  
NamedElementMain\_Mapping

### **Mapping Source**

ActionExecutionSpecification

### **Mapping Target**

ActionUsage

### **Owned Mappings**

(none)

## **7.7.8.3.2 BehaviorExecutionSpecification\_Mapping**

### **Description**

A UML4SysML::BehaviorExecutionSpecification is mapped to a SysML v2 ActionUsage.

### **General Mappings**

GenericToActionUsage\_Mapping  
NamedElementMain\_Mapping

### **Mapping Source**

BehaviorExecutionSpecification

### **Mapping Target**

ActionUsage

### **Owned Mappings**

(none)

## **7.7.8.3.3 CombinedFragment\_Mapping**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

### **Description**

A UML4SysML::CombinedFragment is mapped to a SysMLv2 Interaction.

### **General Mappings**

NamedElementMain\_Mapping  
GenericToInteraction\_Mapping

### **Mapping Source**

CombinedFragment

### Mapping Target

Interaction

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Interaction::ownedRelationship () : Relationship [0..\*]

```
let operands: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::InteractionOperand)) in
let occurrencesSpecs: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::OccurrenceSpecification)) in
let elements: Set(UML::Element) =
 (from.ownedElement - operands) - occurrencesSpecs in
elements->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(operands->collect(e | InteractionOperandMembership_Mapping.getMapped(e))->asSet())
->union(self.ocliAsType(ElementMain_Mapping).ownedRelationship())
```

## 7.7.8.3.4 CombinedFragmentMembership\_Mapping

### Description

Creates a membership relationship for *memberElement()*.

### General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

CombinedFragment

### Mapping Target

FeatureMembership

### Owned Mappings

(none)

### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]  

```
self.memberFeature()
```
- FeatureMembership::memberFeature () : Feature [1]  

```
ElementMain_Mapping.getMapped(from)
```

### 7.7.8.3.5 ExecutionSpecificationMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToEndFeatureMembership\_Mapping

#### Mapping Source

ExecutionSpecification

#### Mapping Target

EndFeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::memberFeature () : Feature [1]  

```
ElementMain_Mapping.getMapped(from)
```
- EndFeatureMembership::ownedMemberFeature () : Feature [0..1]  

```
self.memberFeature()
```

### 7.7.8.3.6 Interaction\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

## Description

A UML4SysML::Interaction is mapped to a SysMLv2 Interaction.

## General Mappings

Namespace\_Mapping

GenericToInteraction\_Mapping

## Mapping Source

Interaction

## Mapping Target

Interaction

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Interaction::ownedRelationship () : Relationship [0..\*]

```
let lifelines: Set(UML::Element) = from.lifeline in
let messageOccurrences: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::MessageOccurrenceSpecification)) in
let executionOccurrences: Set(UML::Element) =
 from.fragment->select(e | e.ocliIsKindOf(UML::ExecutionSpecification)) in
let occurrencesSpecs: Set(UML::Element) =
 from.fragment->select(e | e.ocliIsKindOf(UML::OccurrenceSpecification)) in
let messages: Set(UML::Element) = from.message in
let invariants: Set(UML::Element) =
 from.fragment->select(e | e.ocliIsKindOf(UML::StateInvariant)) in
let interactionUsages: Set(UML::Element) =
 from.fragment->select(e | e.ocliIsKindOf(UML::InteractionUse)) in
let combinedFragments: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::CombinedFragment)) in
let continuations: Set(UML::Element) =
 from.ownedElement->select(e | e.ocliIsKindOf(UML::Continuation)) in
let elements: Set(UML::Element) =
 (((((((from.ownedElement - lifelines) - messageOccurrences)
 - executionOccurrences) - occurrencesSpecs) - messages) -
 combinedFragments) - invariants) -
 interactionUsages) - continuations) - from.ownedComment in

elements->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(lifelines->collect(e | LifelineMembership_Mapping.getMapped(e))->asSet())
```

```

->union(executionOccurrences
 ->collect(e | ExecutionSpecificationMembership_Mapping.getMapped(e)) ->asSet())
->union(messages->collect(e | MessageMembership_Mapping.getMapped(e)) ->asSet())
->union(combinedFragments
 ->collect(e | CombinedFragmentMembership_Mapping.getMapped(e)) ->asSet())
->union(invariants
 ->collect(e | StateInvariantMembership_Mapping.getMapped(e)) ->asSet())
->union(interactionUsages
 ->collect(e | InteractionUseMembership_Mapping.getMapped(e)) ->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())

```

### 7.7.8.3.7 InteractionOperand\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A UML4SysML::InteractionOperand is mapped to a SysML v2 Interaction.

##### General Mappings

NamedElementMain\_Mapping  
GenericToInteraction\_Mapping

##### Mapping Source

InteractionOperand

##### Mapping Target

Interaction

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Interaction::ownedRelationship () : Relationship [0..\*]

```

let executionOccurrences: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::ExecutionSpecification)) in
let occurrencesSpecs: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::OccurrenceSpecification)) in
let continuations: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Continuation)) in
let elements: Set(UML::Element) =
 (((from.ownedElement - executionOccurrences) - occurrencesSpecs) -
continuations) - from.ownedComment in

```

```

elements->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
->union(executionOccurrences
 ->collect(e | ExecutionSpecificationMembership_Mapping.getMapped(e))->asSet())

```

### **7.7.8.3.8 InteractionOperandMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

InteractionOperand

#### **Mapping Target**

FeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]
 

```
self.memberFeature()
```
- FeatureMembership::memberFeature () : Feature [1]
 

```
ElementMain_Mapping.getMapped(from)
```

### **7.7.8.3.9 InteractionUse\_Mapping**

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### **Description**

A UML4SysML::InteractionUse is mapped to a SysML v2 Step.

#### **General Mappings**

GenericToStep\_Mapping  
Namespace\_Mapping

**Mapping Source**

InteractionUse

**Mapping Target**

Step

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Step::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship() -> including(InteractionUseFeatureTypi
```

### 7.7.8.3.10 InteractionUseMembership\_Mapping

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToFeatureMembership\_Mapping

**Mapping Source**

InteractionUse

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::memberFeature () : Feature [1]  
ElementMain\_Mapping.getMapped(from)
- FeatureMembership::ownedMemberFeature () : Feature [0..1]  
self.memberFeature()

### 7.7.8.3.11 InteractionUseFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

InteractionUse

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  
ElementMain\_Mapping.getMapped(from.refersTo)

### 7.7.8.3.12 LifelineMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

Lifeline

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]  

```
self.memberFeature()
```
- FeatureMembership::memberFeature () : Feature [1]  

```
ElementMain_Mapping.getMapped(from)
```

### 7.7.8.3.13 LifelinePartUsage\_Mapping

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

**Description**

A UML4SysML::Lifeline is mapped to a SysML v2 PartUsage.

**General Mappings**

GenericToPartUsage\_Mapping  
NamedElementMain\_Mapping

**Mapping Source**

Lifeline

**Mapping Target**

PartUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PartUsage::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship() -> including(LifelineFeatureTyping_Mapping)
```

### 7.7.8.3.14 LifelineFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

Lifeline

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
ElementMain_Mapping.getMapped(from.represents.type)
```

### 7.7.8.3.15 Message\_Mapping

#### Description

A UML4SysML::Message is mapped to a SysML v2 ItemFlow.

#### General Mappings

GenericToItemFlow\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

Message

**Mapping Target**

ItemFlow

**Owned Mappings**

(none)

**7.7.8.3.16 MessageMembership\_Mapping**

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToFeatureMembership\_Mapping

**Mapping Source**

Message

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]  

```
self.memberFeature()
```
- FeatureMembership::memberFeature () : Feature [1]  

```
ElementMain_Mapping.getMapped(from)
```

**7.7.8.3.17 StateInvariant\_Mapping**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

**Description**

A UML4SysML::StateInvariant is mapped to a SysML v2 Invariant.

## **General Mappings**

GenericToExpression\_Mapping  
Namespace\_Mapping

### **Mapping Source**

StateInvariant

### **Mapping Target**

Invariant

## **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Invariant::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->including(StateInvariantFeatureTyping_Mapping.getMapped(from))
```

## **7.7.8.3.18 StateInvariantMembership\_Mapping**

### **Description**

Creates a membership relationship for *memberElement()*.

## **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

StateInvariant

### **Mapping Target**

FeatureMembership

## **Owned Mappings**

(none)

### **Applicable filters**

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]  

```
self.memberFeature()
```
- FeatureMembership::memberFeature () : Feature [1]  

```
ElementMain_Mapping.getMapped(from)
```

### 7.7.8.3.19 StateInvariantFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

StateInvariant

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  

```
ElementMain_Mapping.getMapped(from.invariant)
```

## 7.7.9 Packages

This chapter lists all mapping specifications of UML4SysML::Packages model elements.

**SYSML2-513: Missing text in some main mapping sections**

### 7.7.9.1 Overview

**SYSML2-441:** Change the table header of the overview tables in the mapping class specification chapters

**SYSML2-564:** Mapping tables in the overview sections show duplicates in the SysML v2 column

Table 13. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax     |
|-------------------------------------|------------------------------|
| Extension                           | not mapped; see next section |
| ExtensionEnd                        | not mapped; see next section |
| Image                               | not mapped; see next section |
| Model                               | Package                      |
| Package                             | Package                      |
| PackageMerge                        | not mapped; see next section |
| Profile                             | Package                      |
| ProfileApplication                  | not mapped; see next section |
| Stereotype                          | MetadataDefinition           |

The following table gives an overview of which SysML v2 elements the UML4SysML::Packages elements are transformed with which mapping class. The mapping details are in [7.7.9.3](#).

The justifications for the elements without mapping are given in [7.7.9.2](#).

### 7.7.9.2 UML4SysML::Packages elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566:** Section containing tables about elements not mapped should get an introductory text

Table 14. List of SysML v1 elements not mapped of this section

| SysML v1 Concept | Rationale                                                                                    |
|------------------|----------------------------------------------------------------------------------------------|
| Extension        | The mapping of the extension relationship is performed in the context of Stereotype_Mapping. |
| ExtensionEnd     | The mapping of the extension end property is performed in the context of Stereotype_Mapping. |
| Image            | Mapping is not specified yet.                                                                |
| PackageMerge     | The concept of the PackageMerge relationship is not supported by SysML v2.                   |

### 7.7.9.3 Mapping Specifications

#### 7.7.9.3.1 ElementImport\_Mapping

**SYSML2-7:** Pin\_Mapping::filter: property src should be from

## Description

A UML4SysML::ElementImport is mapped to a SysMLv2 MembershipImport. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
package SysMLv1Package1 {
 import SysMLv1Package2::SysMLv1Block;
 import SysMLv1Package2::SysMLv1ValueType;
}
package SysMLv1Package2 {
 part def SysMLv1Block;
 attribute def SysMLv1ValueType;
}
```

## General Mappings

GenericToMembershipImport\_Mapping  
NamedElementMain\_Mapping

### Mapping Source

ElementImport

### Mapping Target

MembershipImport

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
if src.oclIsKindOf(UML::ElementImport) then
 Helper.hasMainMapping(src.oclAsType(UML::ElementImport).importedElement)
else
 false
endif
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MembershipImport::importedMemberName () : String [0..1]  
  
from.alias
- MembershipImport::visibility () : VisibilityKind [1]  
  
Helper.getKerMLVisibilityKind(from.visibility)
- MembershipImport::importedMembership () : Namespace [1]

```
ElementOwningMembership_Mapping.getMapped(from.importedElement)
```

### 7.7.9.3.2 Model\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

SysMLv2 has no explicit model element for a model. The UML4SysML::Model element is mapped to a SysMLv2 Package. The property "viewpoint" is mapped to a metadata defined in the SysML v1 library. The expected SysML v2 textual notation of a UML4SysML::Model with URI and viewpoint is as follows. If URI or viewpoint are not set in the source model, the metadata is not generated.

```
package SysMLv1Model {
 @SysMLv1Library::PackageData {URI="https://omg.org";}
 @SysMLv1Library::ModelData {'viewpoint'="The viewpoint of the model element."}
}
```

##### General Mappings

Package\_Mapping

##### Mapping Source

Model

##### Mapping Target

Package

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Package::ownedRelationship () : Relationship [0..\*]

```
let relationships : Set(KerML::Relationship) =
 self.oclAsType(Package_Mapping).ownedRelationship() in
if from.viewpoint.oclIsUndefined() or from.viewpoint = '' then
 relationships
else
 relationships
 ->including(ModelViewpointMetadataMembership_Mapping.getMapped(from))
endif
```

### 7.7.9.3.3 ModelViewpointMetadataUsage\_Mapping

#### **7.7.9.3.4 ModelViewpointMetadataFeatureMembership\_Mapping**

##### **Description**

The mapping class creates the feature membership relationship for the metadata feature to store the UML4SysML::Model::viewpoint property.

##### **General Mappings**

GenericToFeatureMembership\_Mapping

##### **Mapping Source**

Model

##### **Mapping Target**

FeatureMembership

##### **Owned Mappings**

(none)

##### **Applicable filters**

(none)

##### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]

```
ModelViewpointMetadataReferenceUsage_Mapping.getMapped(from)
```

#### **7.7.9.3.5 ModelViewpointMetadataReferenceUsage\_Mapping**

##### **Description**

The mapping class creates the MetadataFeature for the mapping of the property UML4SysML::Model::viewpoint.

##### **General Mappings**

GenericToReferenceUsage\_Mapping

##### **Mapping Source**

Model

##### **Mapping Target**

ReferenceUsage

##### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ModelViewpointMetadataRedefinition_Mapping.getMapped(from),
ModelViewpointMetadataFeatureValue_Mapping.getMapped(from)}
```

### **7.7.9.3.6 ModelViewpointMetadataFeatureTyping\_Mapping**

#### **Description**

The mapping class creates the FeatureTyping relationship for the AnnotatingFeature for the metadata to store the UML4SysML::Model::viewpoint property.

#### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

Model

#### **Mapping Target**

FeatureTyping

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SysMLv2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::ModelData')
```

### **7.7.9.3.7 ModelViewpointMetadataMembership\_Mapping**

#### **Description**

The mapping class creates a membership relationship for the metadata feature value for the UML4SysML::Model::viewpoint property.

### **General Mappings**

GenericToOwningMembership\_Mapping

### **Mapping Source**

Model

### **Mapping Target**

OwningMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
ModelViewpointMetadataUsage_Mapping.getMapped(from)
```

## **7.7.9.3.8 ModelViewpointMetadataFeatureValue\_Mapping**

### **Description**

The mapping class maps the value of the property UML4SysML::Model::viewpoint.

### **General Mappings**

GenericToFeatureValue\_Mapping

### **Mapping Source**

Model

### **Mapping Target**

FeatureValue

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
ModelViewpointValue\_Mapping.getMapped(from)

### 7.7.9.3.9 ModelViewpointMetadataRedefinition\_Mapping

#### Description

The mapping class creates the redefinition of the attribute for the metadata UML4SysML::Model::viewpoint.

#### General Mappings

GenericToRedefinition\_Mapping

#### Mapping Source

Model

#### Mapping Target

Redefinition

#### Owned Mappings

(none)

#### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  

```
let m : SYSML2::Membership =
 SYSML2::AttributeUsage.allInstances()
 ->collect(dt | dt.owningRelationship)
 ->select(r | r.oclIsKindOf(SYSML2::Membership))
 ->any(m | m.memberName = 'viewpoint') in
if (m.oclIsUndefined()) then
 invalid
else
 m.memberElement
endif
```

### 7.7.9.3.10 ModelViewpointValue\_Mapping

## Description

The mapping class maps the value expression of the property UML4SysML::Model::viewpoint.

## General Mappings

GenericToExpression\_Mapping

### Mapping Source

Model

### Mapping Target

LiteralString

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralString::value () : String [1]

```
LiteralString_Factory.create(from.viewpoint)
```

## 7.7.9.3.11 Package\_Mapping

### Description

A UML4SysML::Package is mapped to a SysML v2 Package. The property "URI" is mapped to a metadata if it has a value. The expected SysML v2 textual notation of a UML4SysML::Package is as follows:

```
package ThisIsAPackageWithURI {
 metadata SysMLv1Library::PackageData {URI="https://omg.org";}
}
```

## General Mappings

Namespace\_Mapping

### Mapping Source

Package

### Mapping Target

Package

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Package::ownedRelationship () : Relationship [0..\*]

```
Helper.packageOwnedRelationship(from)
```

### **7.7.9.3.12 PackageImport\_Mapping**

#### **[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

##### **Description**

A UML4SysML::PackageImport is mapped to a SysML v2 NamespaceImport. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
import SysMLv1Package::*;


```

### **General Mappings**

GenericToNamespaceImport\_Mapping  
ElementMain\_Mapping

### **Mapping Source**

PackageImport

### **Mapping Target**

NamespaceImport

### **Owned Mappings**

(none)

### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
if src.oclIsKindOf(UML::PackageImport) then
 Helper.isInScope(src.oclAsType(UML::PackageImport).importedPackage)
else
 false
endif
```

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- NamespaceImport::visibility () : VisibilityKind [0..1]  
Helper.getKerMLVisibilityKind(from.visibility)
- NamespaceImport::importedNamespace () : Namespace [1]  
Namespace\_Mapping.getMapped(from.importedPackage)

### **7.7.9.3.13 PackageURIMetadataUsage\_Mapping**

#### **Description**

The mapping class creates the annotating feature to annotate the generated Package element with metadata to store the UML4SysML::Package::URI property.

#### **General Mappings**

GenericToMetadataUsage\_Mapping

#### **Mapping Source**

Package

#### **Mapping Target**

MetadataUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]  
Set{ PackageURIFeatureTyping\_Mapping.getMapped(from),  
PackageURIFeatureMembership\_Mapping.getMapped(from) }
- MetadataUsage::declaredName () : String [0..1]  
'URI'

### **7.7.9.3.14 PackageURIFeatureMembership\_Mapping**

#### **Description**

The mapping class creates the feature membership relationship for the metadata feature to store the UML4SysML::Package::URI property.

### **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

Package

### **Mapping Target**

FeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
PackageURIMetadataReferenceUsage_Mapping.getMapped(from)
```

## **7.7.9.3.15 PackageURIFeatureTyping\_Mapping**

### **Description**

The mapping class creates the FeatureTyping relationship for the AnnotatingFeature for the metadata to store the UML4SysML::Package::URI property.

### **General Mappings**

GenericToFeatureTyping\_Mapping

### **Mapping Source**

Package

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
let m: SysMLv2::Membership = SysMLv2::AttributeDefinition.allInstances()
->collect(dt | dt.owningRelationship)
->select(r | r.oclIsKindOf(SysMLv2::Membership))
->any(m | m.memberName = 'PackageData') in

if (m.oclIsUndefined()) then
 invalid
else
 m.memberElement
endif
```

### 7.7.9.3.16 PackageURIMetadataReferenceUsage\_Mapping

#### Description

The mapping class creates the MetadataFeature for the mapping of the property UML4SysML::Package::URI.

#### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

Package

#### Mapping Target

ReferenceUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{PackageURIRedefinition_Mapping.getMapped(from),
PackageURIMetadataFeatureValue_Mapping.getMapped(from)}
```

### 7.7.9.3.17 PackageURIMetadataFeatureValue\_Mapping

## Description

The mapping class maps the value of the property UML4SysML::Package::URI.

## General Mappings

GenericToFeatureValue\_Mapping

### Mapping Source

Package

### Mapping Target

FeatureValue

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::featureWithValue () : Feature [1]  
    packageURIMetadataReferenceUsage.to
- FeatureValue::value () : Expression [1]  
    PackageURIValue\_Mapping.getMapped(from)

## 7.7.9.3.18 PackageURIMetadataMembership\_Mapping

## Description

The mapping class creates a membership relationship for the metadata feature value for the UML4SysML::Package::URI property.

## General Mappings

GenericToOwningMembership\_Mapping

### Mapping Source

Package

### Mapping Target

OwningMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
PackageURI1MetadataUsage_Mapping.getMapped(from)
```

## **7.7.9.3.19 PackageURIRedefinition\_Mapping**

### **Description**

The mapping class creates the redefinition of the attribute for the metadata UML4SysML::Package::URI.

### **General Mappings**

GenericToRedefinition\_Mapping

### **Mapping Source**

Package

### **Mapping Target**

Redefinition

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
let m : SysMLv2::Membership =
 SysMLv2::AttributeUsage.allInstances()
 ->collect(dt | dt.owningRelationship)
 ->select(r | r.oclIsKindOf(SYSML2::Membership))
 ->any(m | m.memberName = 'URI') in
if (m.oclIsUndefined()) then
```

```
 invalid
else
 m.memberElement
endif
```

### 7.7.9.3.20 PackageURIValue\_Mapping

#### Description

The mapping class maps the value expression of the property UML4SysML::Package::URI.

#### General Mappings

GenericToExpression\_Mapping

#### Mapping Source

Package

#### Mapping Target

LiteralString

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralString::value () : String [1]

    from.URI

### 7.7.9.3.21 Profile\_Mapping

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### Description

A UML4SysML::Profile is mapped to a SysML v2 Package.

#### General Mappings

Package\_Mapping

#### Mapping Source

Profile

## **Mapping Target**

Package

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Package::ownedRelationship () : Relationship [0..\*]  

```
self.oclAsType(Package_Mapping).ownedRelationship()
->including(ProfileMetadataMembership_Mapping.getMapped(from))
```

## **7.7.9.3.22 ProfileMetadataMembership\_Mapping**

### **Description**

The mapping class creates a membership relationship for the metadata feature value for the UML4SysML::Model::viewpoint property.

### **General Mappings**

GenericToOwningMembership\_Mapping

### **Mapping Source**

Profile

### **Mapping Target**

OwningMembership

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
ProfileMetadataUsage_Mapping.getMapped(from)
```

### 7.7.9.3.23 ProfileMetadataUsage\_Mapping

#### Description

The mapping class creates the annotating feature to annotate the generated Package element with metadata to store the UML4SysML::Model::viewpoint property.

#### General Mappings

GenericToMetadataUsage\_Mapping

#### Mapping Source

Profile

#### Mapping Target

MetadataUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::declaredName () : String [0..1]  
    'Profile'

### 7.7.9.3.24 StereotypeMetadataDefinition\_Mapping

#### Description

A UML4SysML::Stereotype is mapped to a SysML v2 MetadataDefinition.

#### General Mappings

Class\_Mapping

#### Mapping Source

Stereotype

#### Mapping Target

MetadataDefinition

## **Owned Mappings**

(none)

### **7.7.9.3.25 StereotypeMetadataDefinitionMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

ElementOwningMembership\_Mapping

#### **Mapping Source**

Stereotype

#### **Mapping Target**

OwningMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [0..1]

ElementMain\_Mapping.getMapped(from)

### **7.7.9.3.26 StereotypeOccurrenceUsage\_Mapping**

#### **Description**

The mapping class maps the usage of a stereotype to a SysML v2 OccurrenceUsage.

#### **General Mappings**

GenericToOccurrenceUsage\_Mapping

#### **Mapping Source**

Stereotype

#### **Mapping Target**

OccurrenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OccurrenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{StereotypeOccurenceUsageFeatureTyping_Mapping.getMapped(from),
StereotypeOccurenceUsageMultiplicityMembership_Mapping.getMapped(from)}
```

## **7.7.9.3.27 StereotypeOccurenceUsageFeatureTyping\_Mapping**

### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

### **General Mappings**

GenericToFeatureTyping\_Mapping

### **Mapping Source**

Stereotype

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
StereotypeOccurenceDefinition_Mapping.getMapped(from)
```

## **7.7.9.3.28 StereotypeOccurenceUsageMembership\_Mapping**

### **Description**

Creates a membership relationship for *memberElement()*.

### **General Mappings**

GenericToMembership\_Mapping

#### **Mapping Source**

Stereotype

#### **Mapping Target**

Membership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]  
StereotypeOccurrenceUsage\_Mapping.getMapped (from)

### **7.7.9.3.29 StereotypeOccurrenceUsageMultiplicityMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

### **General Mappings**

GenericToMembership\_Mapping

#### **Mapping Source**

Stereotype

#### **Mapping Target**

Membership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::ownedMemberElement () : Element [0..1]  
    StereotypeOccurrenceUsageMultiplicityRange\_Mapping.getMapped(from)
- Membership::memberElement () : Element [1]  
    self.ownedMemberElement()

### **7.7.9.3.30 StereotypeOccurrenceUsageMultiplicityRange\_Mapping**

#### **Description**

The mapping class creates the multiplicity range element for the UML4SysML::Stereotype mapping.

#### **General Mappings**

GenericToFeature\_Mapping

#### **Mapping Source**

Stereotype

#### **Mapping Target**

MultiplicityRange

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MultiplicityRange::ownedRelationship () : Relationship [0..\*]  
    Set{StereotypeOccurrenceUsageMultiplicityRangeMembership\_Mapping.getMapped(from)}

### **7.7.9.3.31 StereotypeOccurrenceUsageMultiplicityRangeInfinity\_Mapping**

#### **Description**

The mapping class creates the literal infinity element for the multiplicity range element for the UML4SysML::Stereotype mapping.

#### **General Mappings**

GenericToExpression\_Mapping

**Mapping Source**

Stereotype

**Mapping Target**

LiteralInfinity

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralInfinity::ownedRelationship () : Relationship [0..\*]

```
Set{StereotypeOccurrenceUsageInfinityReturnParameterMembership_Mapping.getMapped(from)}
```

### 7.7.9.3.32 StereotypeOccurrenceUsageInfinityReturnParameter\_Mapping

**Description**

The mapping class creates the return parameter relationship for the literal infinity element for the multiplicity range element for the UML4SysML::Stereotype mapping.

**General Mappings**

GenericToFeature\_Mapping

**Mapping Source**

Stereotype

**Mapping Target**

Feature

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::direction () : FeatureDirectionKind [0..1]

```
SysMLv2::FeatureDirectionKind::out
```

### 7.7.9.3.33 StereotypeOccurrenceUsageInfinityReturnParameterMembership\_Mapping

#### Description

#### General Mappings

GenericToReturnParameterMembership\_Mapping

#### Mapping Source

Stereotype

#### Mapping Target

ReturnParameterMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReturnParameterMembership::ownedMemberParameter () : Feature [0..1]

```
StereotypeOccurrenceUsageInfinityReturnParameter_Mapping.getMapped(from)
```

- ReturnParameterMembership::ownedRelatedElement () : Element [0..\*]

```
let member: KerML::Element = self.ownedMemberParameter() in
if member.oclIsUndefined() then
 Set{}
else
 Set{self.ownedMemberParameter()}
endif
```

- ReturnParameterMembership::memberParameter () : Feature [1]

```
self.ownedMemberParameter()
```

### 7.7.9.3.34 StereotypeOccurrenceUsageMultiplicityRangeMembership\_Mapping

## Description

Creates a membership relationship for *memberElement()*.

## General Mappings

GenericToMembership\_Mapping

### Mapping Source

Stereotype

### Mapping Target

Membership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::ownedMemberElement () : Element [0..1]  

```
StereotypeOccurrenceUsageMultiplicityRangeInfinity_Mapping.getMapped(from)
```
- Membership::memberElement () : Element [1]  

```
self.ownedMemberElement()
```

## 7.7.10 SimpleClassifiers

This chapter lists all mapping specifications of UML4SysML::SimpleClassifiers model elements.

**SYSML2-513: Missing text in some main mapping sections**

### 7.7.10.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column**

Table 15. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax |
|-------------------------------------|--------------------------|
| DataType                            | AttributeDefinition      |
| Enumeration                         | EnumerationDefinition    |

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax                        |
|-------------------------------------|-------------------------------------------------|
| EnumerationLiteral                  | EnumerationUsage                                |
| Interface                           | PortDefinition                                  |
| InterfaceRealization                | SatisfyRequirementUsage<br>AllocationDefinition |
| PrimitiveType                       | AttributeDefinition                             |
| Reception                           | ItemUsage                                       |
| Signal                              | ItemDefinition                                  |

The following table gives an overview of which SysML v2 elements the UML4SysML::SimpleClassifiers elements are transformed with which mapping class. The mapping details are in [7.7.10.2](#).

## 7.7.10.2 Mapping Specifications

### 7.7.10.2.1 Attribute\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

An UML4SysML::Property is mapped to a SysMLv2 AttributeUsage.

##### General Mappings

PropertyCommon\_Mapping  
NamedElementMain\_Mapping

##### Mapping Source

Property

##### Mapping Target

AttributeUsage

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
if src.oclIsKindOf(UML::Property) and not
 Helper.hasStereotypeApplied(src.owner,
 'SysML::ConstraintBlocks::ConstraintBlock') then
 let p: UML::Property = src.oclAsType(UML::Property) in
 if p.type.oclIsUndefined() then
 false
 else
 p.type.oclIsKindOf(UML::DataType) and
```

```

 (p.association.oclIsUndefined() or p.association.ownedEnd->excludes (p))
 endif
else
 false
endif

```

## Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.7.10.2.2 AttributeRedefined\_Mapping

#### Description

An UML4SysML::SimpleClassifiers::Property is mapped to a SysML v2 AttributeUsage.

#### General Mappings

PropertyCommon\_Mapping

#### Mapping Source

Property

#### Mapping Target

ReferenceUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```

let typing: KerML::FeatureTyping =
 AssociationToFeatureTyping_Mapping.getMapped(from) in
let subsetting: Set(KerML::Subsetting) =
 from.subsettedProperty
 ->collect(p | PropertySubsetting_Mapping.getMapped(from, p))->asSet() in
let subsettingMultiplicityTyping: Set(KerML::Relationship) =
 subsetting
 ->union(Set{AttributeRedefinedRedefinition_Mapping.getMapped(from)})->union(
 if typing.oclIsUndefined() then
 Set{MultiplicityMembership_Mapping.getMapped(from)} }
 else
 Set{MultiplicityMembership_Mapping.getMapped(from), typing})
 endif)->asSet() in

```

```

if from.defaultValue.oclIsUndefined() then
 subsettingMultiplicityTyping
else
 subsettingMultiplicityTyping
 ->including(PropertyDefaultValue_Mapping.getMapped(from))
endif

```

### **7.7.10.2.3 AttributeRedefinedRedefinition\_Mapping**

#### **Description**

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

#### **General Mappings**

GenericToRedefinition\_Mapping

#### **Mapping Source**

Property

#### **Mapping Target**

Redefinition

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
from.redefinedProperty.get(0)
```

### **7.7.10.2.4 AttributeRedefinedMembership\_Mapping**

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

ElementFeatureMembership\_Mapping

#### **Mapping Source**

Element

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.oclIsKindOf(UML::Property)
and (src.oclAsType(UML::Property).redefinedElement->size() > 0)
```

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]

```
AttributeRedefined_Mapping.getMapped(from)
```

### 7.7.10.2.5 AttributeRedefinedFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

StructuralFeatureToFeatureTyping\_Mapping

#### Mapping Source

StructuralFeature

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

### 7.7.10.2.6 BehavioredClassifier\_Mapping

**SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported**

**SYSML2-208: A ConnectionUsage should be owned by a FeatureMembership relationship**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

## Description

The abstract mapping class maps the abstract metaclass UML4SysML::BehavioredClassifiers to a SysMLv2 Classifier. The mapping class is used by concrete mapping classes, for example, Block\_Mapping.

## General Mappings

Classifier\_Mapping

### Mapping Source

BehavioredClassifier

### Mapping Target

Classifier

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Classifier::ownedRelationship () : Relationship [0..\*]

```
let toElementFMS: Set(UML::Element) =
 from.ownedElement->select(e | (e.oclIsKindOf(UML::Property) and
 (e.oclAsType(UML::Property).redefinedProperty->size() = 0)) or
 e.oclIsKindOf(UML::Operation) or e.oclIsKindOf(UML::Connector)) in
let redefinedAttributes: Set(UML::Element) =
 from.ownedElement->select(e | from.oclIsKindOf(UML::DataType) and
 (e.oclAsType(UML::Property).redefinedProperty->size() > 0)) in
let generalizations : Set(UML::Generalization) =
 from.ownedElement
 ->select(e | e.oclIsKindOf(UML::Generalization)) in
let constraints : Set(UML::Constraint) =
 UML::Constraint.allInstances()
 ->select(c | c.constrainedElement->includes(from)) in
let toElementOMS: Set(UML::Element) =
 (((from.ownedElement - toElementFMS) - redefinedAttributes) -
 generalizations) - from.ownedComment in
let relationships: Sequence(KerML::Relationship) =
 toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
 ->union(toElementFMS->collect(e |
 ElementFeatureMembership_Mapping.getMapped(e))->asSet())
 ->union(constraints->collect(e |
 ConstrainedElementFeatureMembership_Mapping.getMapped(e))->asSet())
 ->union(redefinedAttributes->collect(e |
 AttributeRedefinedMembership_Mapping.getMapped(e))->asSet())
```

```

->union(generalizations->collect(e |
 Generalization_Mapping.getMapped(e))->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship()) in
if from.classifierBehavior.oclIsUndefined() then
 relationships
else
 relationships
->including(BehavioredClassifierFeatureMembership_Mapping.getMapped(from))
endif

```

### **7.7.10.2.7 BehavioredClassifierFeatureMembership\_Mapping**

#### **Description**

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

BehavioredClassifier

#### **Mapping Target**

FeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]

BehavioredClassifierActionUsage\_Mapping.getMapped(from)

### **7.7.10.2.8 BehavioredClassifierFeatureTyping\_Mapping**

#### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

#### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

BehavioredClassifier

**Mapping Target**

FeatureTyping

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

from

### 7.7.10.2.9 BehavioredClassifierActionUsage\_Mapping

**Description**

The BehavioredClassifierToPerformActionUsage\_Mapping class creates a PerformActionUsage element to call the transformed SysML v1 classifier behavior.

**General Mappings**

GenericToActionUsage\_Mapping

**Mapping Source**

BehavioredClassifier

**Mapping Target**

ActionUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::declaredName () : String [0..1]

```

'classifierBehavior'

• ActionUsage::ownedRelationship () : Relationship [0..*]

Set{BehavioredClassifierFeatureTyping_Mapping.getMapped (from) }

```

### **7.7.10.2.10 DataType\_Mapping**

#### **Description**

A UML4SysML::SimpleClassifiers::DataType is mapped to a SysML v2 AttributeDefinition. The mapping also cover the transformation of UML4SysML::PrimitiveType elements.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```

part def SysMLv1Block {
 attribute sysMLv1Property : ScalarValues::Integer;
}

```

#### **General Mappings**

Classifier\_Mapping

#### **Mapping Source**

DataType

#### **Mapping Target**

AttributeDefinition

#### **Owned Mappings**

(none)

### **7.7.10.2.11 Enumeration\_Mapping**

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### **Description**

A UML4SysML::Enumeration is mapped to a SysML v2 EnumerationDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```

enum def SysMLv1Enumeration {
 enum sysMLv1Literal1;
 enum sysMLv1Literal2;
}

```

#### **General Mappings**

DataType\_Mapping

### **Mapping Source**

Enumeration

### **Mapping Target**

EnumerationDefinition

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EnumerationDefinition::isVariation () : Boolean [1]  
true
- EnumerationDefinition::ownedRelationship () : Relationship [0..\*]  

```
self.oclAsType(Classifier_Mapping).ownedRelationship()
->union(from.ownedLiteral->collect(e | EnumerationVariantMembership_Mapping.getMapped(e))->as
```

## **7.7.10.2.12 EnumerationLiteral\_Mapping**

### **Description**

A UML4SysML::EnumerationLiteral is mapped to a SysML v2 EnumerationUsage.

### **General Mappings**

GenericToFeature\_Mapping  
InstanceSpecification\_Mapping

### **Mapping Source**

EnumerationLiteral

### **Mapping Target**

EnumerationUsage

### **Owned Mappings**

(none)

## **7.7.10.2.13 EnumerationVariantMembership\_Mapping**

### **Description**

The EnumerationVariantMembership\_Mapping class creates the variant membership relationship between the enumeration definition and a enumeration usage.

### General Mappings

GenericToOwningMembership\_Mapping

### Mapping Source

EnumerationLiteral

### Mapping Target

VariantMembership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- VariantMembership::ownedMemberElement () : Element [1]

from

### 7.7.10.2.14 Interface\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A UML4SysML::Interface is mapped to a SysMLv2 PortDefinition. The mapping also includes the generation of an appropriate ConjugatedPortDefinition. That mappings is performed by the mapping classes InterfaceConjugatedPortDefinitionMembership\_Mapping, InterfacePortConjugation\_Mapping, and InterfaceConjugatedPortDefinition\_Mapping.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
port def SysMLv1Interface {
 attribute sysMLv1Property;
}
```

### General Mappings

GenericToPortDefinition\_Mapping  
Classifier\_Mapping

## **Mapping Source**

Interface

## **Mapping Target**

PortDefinition

## **Owned Mappings**

- conjugatedPortDefinitionMembership : InterfaceConjugatedPortDefinitionMembership\_Mapping

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PortDefinition::ownedRelationship () : Relationship [0..\*]  

```
self.oclAsType(Classifier_Mapping).ownedRelationship()
->including(conjugatedPortDefinitionMembership)
```

## **7.7.10.2.15 InterfaceConjugatedPortDefinition\_Mapping**

### **Description**

As part of the mapping from a UML4SysML::Interface to a SysMLv2 PortDefinition, this mapping class is used to create the appropriate ConjugatedPortDefinition.

## **General Mappings**

GenericToPortDefinition\_Mapping

## **Mapping Source**

Interface

## **Mapping Target**

ConjugatedPortDefinition

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConjugatedPortDefinition::declaredName () : String [0..1]  
`'~' + from.name`
- ConjugatedPortDefinition::ownedRelationship () : Relationship [0..\*]  
`Set{InterfacePortConjugation_Mapping.getMapped(from)}`

### 7.7.10.2.16 InterfaceConjugatedPortDefinitionMembership\_Mapping

#### Description

As part of the mapping from a UML4SysML::Interface to a SysML v2 PortDefinition, this mapping class is used to create the membership relationship for the ConjugatedPortDefinition.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Interface

#### Mapping Target

OwningMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
`InterfaceConjugatedPortDefinition_Mapping.getMapped(from)`

### 7.7.10.2.17 InterfacePortConjugation\_Mapping

#### Description

As part of the mapping from a UML4SysML::Interface to a SysML v2 PortDefinition, this mapping class is used to create the appropriate PortConjugation relationship.

#### General Mappings

GenericToRelationship\_Mapping

**Mapping Source**

Interface

**Mapping Target**

PortConjugation

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PortConjugation::conjugatedType () : Type [1]  

```
SysMLv2::ConjugatedPortDefinition.allInstances()
->collect(cpd | cpd.owningRelationship)
->select(r | r.oclIsKindOf(SysMLv2::Membership))
->any(m | m.memberName = from.name)
```
- PortConjugation::originalPortDefinition () : PortDefinition [1]  

```
from
```

### 7.7.10.2.18 InterfaceRealization\_Mapping

**Description**

A UML4SysML::InterfaceRealization is mapped to a SysMLv2 Subclassification relationship.

**General Mappings**

GenericToSpecialization\_Mapping

**Mapping Source**

InterfaceRealization

**Mapping Target**

Subclassification

**Owned Mappings**

(none)

**Applicable filters**

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subclassification::subclassifer () : Type [1]  

```
Classifier_Mapping.getMapped(from_specific)
```
- Subclassification::superclassifier () : Type [1]  

```
Classifier_Mapping.getMapped(from_general)
```

## 7.7.10.2.19 PrimitiveType\_Mapping

### Description

The PrimitiveType\_Mapping class maps a UML4SysML::PrimitiveType to a SysML v2 AttributeDefinition.

### General Mappings

DataType\_Mapping

### Mapping Source

PrimitiveType

### Mapping Target

AttributeDefinition

### Owned Mappings

(none)

## 7.7.10.2.20 Reception\_Mapping

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

### Description

A UML4SysML::Reception is mapped to a SysML v2 AttributeUsage with feature direction "in".

### General Mappings

BehavioralFeature\_Mapping

### Mapping Source

Reception

### Mapping Target

ItemUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ItemUsage::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship() -> including(ReceptionFeatureTyping_Mapping)
```

- ItemUsage::direction () : FeatureDirectionKind [0..1]

```
SysMLv2::FeatureDirectionKind::in
```

## **7.7.10.2.21 ReceptionFeatureTyping\_Mapping**

### **Description**

A UML4SysML::Reception is mapped to SysML v2 AttributeUsage. The ReceptionToFeatureTyping\_Mapping class creates the type of the AttributeUsage which is the Signal of the Reception.

### **General Mappings**

TypedElementFeatureTyping\_Mapping

### **Mapping Source**

Reception

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
Classifier_Mapping.getMapped(from.signal)
```

### 7.7.10.2.22 Signal\_Mapping

#### Description

A UML4SysML::Signal is mapped to a SysML v2 AttributeDefinition.

#### General Mappings

Classifier\_Mapping

#### Mapping Source

Signal

#### Mapping Target

ItemDefinition

#### Owned Mappings

(none)

## 7.7.11 StateMachines

**SYSML2-1: "Elements not mapped" table sections are empty**

**SYSML2-513: Missing text in some main mapping sections**

#### 7.7.11.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column**

**SYSML2-511: Remove sentence in StateMachines overview section**

Table 16. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax                              |
|-------------------------------------|-------------------------------------------------------|
| ConnectionPointReference            | StateUsage                                            |
| FinalState                          | StateUsage                                            |
| Pseudostate                         | StateUsage                                            |
| Region                              | StateUsage                                            |
| State                               | StateUsage                                            |
| StateMachine                        | ViewDefinition<br>StateDefinition<br>RequirementUsage |
| Transition                          | TransitionUsage                                       |

The following table gives an overview of which SysML v2 elements the UML4SysML::StateMachines elements are transformed with which mapping class. The mapping details are in [7.7.11.2](#).

#### 7.7.11.2 Mapping Specifications

### **7.7.11.2.1 ConnectionPointReference\_Mapping**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### **Description**

A UML4SysML::ConnectionPointReference element is mapped to a SysML v2 StateUsage.

#### **General Mappings**

Namespace\_Mapping

GenericToStateUsage\_Mapping

#### **Mapping Source**

ConnectionPointReference

#### **Mapping Target**

StateUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- StateUsage::isComposite () : Boolean [1]  
    false
- StateUsage::ownedRelationship () : Relationship [0..\*]  

```
let toFeatureMS : Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Region)) in
let toElementOMS : Set(UML::Element) =
 (from.ownedElement - toFeatureMS) - from.ownedComment in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(toFeatureMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e))->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

### **7.7.11.2.2 FinalState\_Mapping**

#### **Description**

A UML4SysML::FinalState is mapped to a SysML v2 StateUsage. The details of the mapping are not defined yet.

#### **General Mappings**

State\_Mapping

**Mapping Source**

FinalState

**Mapping Target**

StateUsage

**Owned Mappings**

(none)

**Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
src.oclIsTypeOf(UML::FinalState)
```

**Mapping rules**

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.7.11.2.3 PseudoState\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

**Description**

A UML4SysML::PseudoState is mapped to a SysML v2 StateUsage.

**General Mappings**

Namespace\_Mapping

GenericToStateUsage\_Mapping

**Mapping Source**

Pseudostate

**Mapping Target**

StateUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- StateUsage::ownedRelationship () : Relationship [0..\*]

```
let toFeatureMS : Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Region))->asSet() in
let toElementOMS : Set(UML::Element) =
 from.ownedElement - toFeatureMS in
toElementOMS
->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(toFeatureMS
->collect(e | ElementFeatureMembership_Mapping.getMapped(e))->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

#### 7.7.11.2.4 Region\_Mapping

##### **[SYSML2-280: ElementMain\\_Mapping::ownedRelationship is wrong](#)**

###### Description

A UML4SysML::Region is mapped to SysML v2 StateUsage.

###### General Mappings

Namespace\_Mapping  
GenericToStateUsage\_Mapping

###### Mapping Source

Region

###### Mapping Target

StateUsage

###### Owned Mappings

(none)

###### Applicable filters

(none)

###### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- StateUsage::ownedRelationship () : Relationship [0..\*]

```
let toFeatureMS : Set(UML::Element) =
 from.ownedElement
 ->select(e | e.oclIsKindOf(UML::State) or e.oclIsKindOf(UML::Transition)) in
let toElementOMS : Set(UML::Element) =
 (from.ownedElement - toFeatureMS) - from.ownedComment in
```

```

toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(toFeatureMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e))->asSet())
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())

```

### 7.7.11.2.5 State\_Mapping

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### Description

A UML4SysML::State is mapped to a SysML v2 StateUsage.

#### General Mappings

Namespace\_Mapping  
GenericToStateUsage\_Mapping

#### Mapping Source

State

#### Mapping Target

StateUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- StateUsage::ownedRelationship () : Relationship [0..\*]

```

let toFeatureMS : Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Region))->asSet() in
let toElementOMS : Set(UML::Element) =
 (from.ownedElement - toFeatureMS) - from.ownedComment in
toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))->asSet()
->union(toFeatureMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))->asSet()
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())

```

### 7.7.11.2.6 StateDefinition\_Mapping

**SYSML2-202: Filter for mapping class Behavior\_Mapping is useless**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-221: UML4SysML::Activities and StateMachines owned by blocks should be mapped to definition elements**

## Description

A UML4SysML::StateMachine is mapped to a SysML v2 StateDefinition.

## General Mappings

Behavior\_Mapping

### Mapping Source

StateMachine

### Mapping Target

StateDefinition

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- StateDefinition::ownedRelationship () : Relationship [0..\*]

```
let initialState : Set(UML::Element) =
 from.ownedElement
 ->select(e | e.oclIsKindOf(UML::Pseudostate) and
 e.oclaType(UML::Pseudostate).kind = UML::PseudostateKind::initial) in
let toParameterMS : Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Parameter)) in
let parameterSets: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::ParameterSet)) in
let toFeatureMS : Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Region)) in
let toElementOMS : Set(UML::Element) =
 ((from.ownedElement - toFeatureMS) - initialState in
 toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e)))
 ->union(toFeatureMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
 ->union(toParameterMS->collect(e | ParameterMembership_Mapping.getMapped(e)))
 ->union(parameterSets->collect(e | ParameterSetMembership_Mapping.getMapped(e)))
 ->union(initialState->collect(e | InitialStateMembership_Mapping.getMapped(e)))
```

- StateDefinition::isParallel () : Boolean [1]

```
from.region->size() > 1
```

### 7.7.11.2.7 Transition\_Mapping

[SYSML2-211: Introduce GenericToTransitionUsage\\_Mapping class](#)

[SYSML2-280: ElementMain\\_Mapping::ownedRelationship is wrong](#)

## Description

A UML4SysML::Transition is mapped to a SysML v2 TransitionUsage.

## General Mappings

Namespace\_Mapping  
GenericToTransitionUsage\_Mapping

## Mapping Source

Transition

## Mapping Target

TransitionUsage

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TransitionUsage::target () : ActionUsage [1]  

```
from.target
```
- TransitionUsage::ownedRelationship () : Relationship [0..\*]  

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->union((from.ownedElement - from.ownedComment)->collect(e | ElementOwningMembership_Mapping.
->including(TransitionSuccession_Mapping.getMapped(from)))
```
- TransitionUsage::source () : ActionUsage [1]  

```
from.source
```

## 7.7.11.2.8 TransitionSuccession\_Mapping

### Description

The mapping class creates the source Feature element of the Succession that is part of the TransitionUsage that is the target element of the UML4SysML::Transition mapping.

### General Mappings

GenericToConnector\_Mapping  
GenericToMembership\_Mapping

**Mapping Source**

Transition

**Mapping Target**

Succession

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Succession::ownedRelationship () : Relationship [0..\*]

```
OrderedSet{TransitionSuccessionSourceMembership_Mapping.getMapped(from),
TransitionSuccessionTargetMembership_Mapping.getMapped(from)}
```

### 7.7.11.2.9 TransitionSourceToSubsetting\_Mapping

#### **SYSML2-200: Description of Subsetting mapping classes is not correct**

**Description**

Creates a subsetting relationship.

**General Mappings**

GenericToSubsetting\_Mapping

**Mapping Source**

Transition

**Mapping Target**

Subsetting

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subsetting::subsettingFeature () : Feature [1]  
TransitionSuccessionSource\_Mapping.getMapped(from)
- Subsetting::subsettedFeature () : Feature [1]  
ElementMain\_Mapping.getMapped(from.source)

### 7.7.11.2.10 TransitionSuccessionSource\_Mapping

#### Description

The mapping class creates the Succession element that is part of the TransitionUsage that is the target element of the UML4SysML::Transition mapping.

#### General Mappings

GenericToFeature\_Mapping

#### Mapping Source

Transition

#### Mapping Target

Feature

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]  
Set{TransitionSourceToSubsetting\_Mapping.getMapped(from)}
- Feature::declaredName () : String [0..1]  
'source'
- Feature::isEnd () : Boolean [1]  
true

### 7.7.11.2.11 TransitionSuccessionSourceMembership\_Mapping

## Description

Creates a membership relationship for *memberElement()*.

## General Mappings

GenericToEndFeatureMembership\_Mapping

## Mapping Source

Transition

## Mapping Target

EndFeatureMembership

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

```
TransitionSuccessionSource_Mapping.getMapped(from)
```

## 7.7.11.2.12 TransitionSuccessionTarget\_Mapping

### Description

The mapping class creates the target Feature element of the Succession that is part of the TransitionUsage that is the target element of the UML4SysML::Transition mapping.

## General Mappings

GenericToFeature\_Mapping

## Mapping Source

Transition

## Mapping Target

Feature

## Owned Mappings

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]  
    true
- Feature::declaredName () : String [0..1]  
    'target'
- Feature::ownedRelationship () : Relationship [0..\*]  
    Set{TransitionTargetToSubsetting\_Mapping.getMapped(from)}

### **7.7.11.2.13 TransitionSuccessionTargetMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

GenericToEndFeatureMembership\_Mapping

#### **Mapping Source**

Transition

#### **Mapping Target**

EndFeatureMembership

#### **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]  
    TransitionSuccessionTarget\_Mapping.getMapped(from)

### **7.7.11.2.14 TransitionTargetToSubsetting\_Mapping**

**[SYSML2-200: Description of Subsetting mapping classes is not correct](#)**

#### **Description**

Creates a subsetting relationship.

#### **General Mappings**

GenericToSubsetting\_Mapping

#### **Mapping Source**

Transition

#### **Mapping Target**

Subsetting

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

This chapter lists all mapping specifications of UML4SysML::StateMachines model elements.

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subsetting::subsettingFeature () : Feature [1]  
    TransitionSuccessionTarget\_Mapping.getMapped (from)
- Subsetting::subsettedFeature () : Feature [1]  
    ElementMain\_Mapping.getMapped (from.target)

### **7.7.12 StructuredClassifiers**

This chapter lists all mapping specifications of UML4SysML::StructuredClassifiers model elements.

**[SYSML2-513: Missing text in some main mapping sections](#)**

#### **7.7.12.1 Overview**

**[SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters](#)**

**[SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column](#)**

**Table 17. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax           |
|-------------------------------------|------------------------------------|
| Association                         | not mapped; see next section       |
| AssociationClass                    | ConnectionDefinition               |
| Class                               | ViewDefinition<br>RequirementUsage |
| Connector                           | ConnectionUsage                    |
| ConnectorEnd                        | not mapped; see next section       |
| Port                                | PartUsage                          |

The following table gives an overview of which SysML v2 elements the UML4SysML::StructuredClassifiers elements are transformed with which mapping class. The mapping details are in [7.7.12.2](#).

## 7.7.12.2 Mapping Specifications

### 7.7.12.2.1 AssociationClass\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

A UML4SysML::AssociationClass is mapped to a SysML v2 ConnectionDefinition. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block1;
part def SysMLv1Block2;
connection def SysMLv1AssociationBlock {
 end : SysMLv1Block1;
 end : SysMLv1Block2;
}
```

##### General Mappings

AssociationCommon\_Mapping

##### Mapping Source

AssociationClass

##### Mapping Target

ConnectionDefinition

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
not Helper.hasStereotypeApplied(src, 'SysML::Blocks::Block')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `ConnectionDefinition::ownedRelationship () : Relationship [0..*]`

```
let nonOwnedEnds: OrderedSet(UML::Property) =
 (from.memberEnd-from.ownedEnd)->asOrderedSet() in
let generalizations : Set(UML::Generalization) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Generalization)) in
let others: OrderedSet(UML::Element) =
 ((from.ownedElement-from.memberEnd)-generalizations)->asOrderedSet() in
nonOwnedEnds->collect(e | NonOwnedEndMembership_Mapping.getMapped(e))
->union(from.ownedEnd->collect(e | OwnedEndMembership_Mapping.getMapped(e)))
->union(generalizations->collect(e | Generalization_Mapping.getMapped(e)))
->union(others->collect(e | ElementOwningMembership_Mapping.getMapped(e)))
->asOrderedSet()
```

#### 7.7.12.2 AssociationCommon\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A UML4SysML::Association is mapped to a SysML v2 ConnectionDefinition. This is the abstract base class of all concrete association mapping classes.

##### General Mappings

Classifier\_Mapping  
Relationship\_Mapping

##### Mapping Source

Association

##### Mapping Target

Association

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
src.memberEnd->select(m | m.type.oclIsKindOf(UML::UseCase))->isEmpty()
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Association::ownedRelationship () : Relationship [0..\*]

```
let nonOwnedEnds: OrderedSet(UML::Property) =
 (from.memberEnd-from.ownedEnd)->asOrderedSet() in
nonOwnedEnds->collect(e | NonOwnedEndMembership_Mapping.getMapped(e))->asOrderedSet()
->union(self.oclAsType(Classifier_Mapping).ownedRelationship()->asOrderedSet())
->asOrderedSet()
```

### 7.7.12.2.3 AssociationMetadataUsage\_Mapping

#### Description

The mapping class creates the MetadataUsage element to annotate a ConnectionDefinition that its mapping source element is a derived association.

#### General Mappings

GenericToMetadataUsage\_Mapping

#### Mapping Source

Association

#### Mapping Target

MetadataUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{AssociationToFeatureTyping_Mapping.getMapped(from),
AssociationMetadataUsageFeatureMembership_Mapping.getMapped(from)}
```

### 7.7.12.2.4 AssociationMetadataUsageFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

Association

#### **Mapping Target**

FeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
    AssociationMetadataUsageFeature\_Mapping.getMapped(from)

### **7.7.12.2.5 AssociationMetadataUsageFeatureTyping\_Mapping**

#### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

Association

#### **Mapping Target**

FeatureTyping

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  

```
SYSML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::AssociationData')
```

### 7.7.12.2.6 AssociationMetadataUsageFeature\_Mapping

#### Description

The mapping class creates the feature of the MetadataUsage.

#### General Mappings

GenericToFeature\_Mapping

#### Mapping Source

Association

#### Mapping Target

Feature

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]  

```
Set{AssociationMetadataUsageRedefinition_Mapping.getMapped(from),
AssociationMetadataUsageFeatureValue_Mapping.getMapped(from)}
```

### 7.7.12.2.7 AssociationMetadataUsageFeatureValue\_Mapping

#### Description

Creates a feature value relationship.

#### General Mappings

GenericToFeatureValue\_Mapping

**Mapping Source**

Association

**Mapping Target**

FeatureValue

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  

```
LiteralBoolean_Factory.create(from.isDerived)
```

**7.7.12.2.8 AssociationMetadataUsageMembership\_Mapping****Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToOwningMembership\_Mapping

**Mapping Source**

Association

**Mapping Target**

OwningMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
AssociationMetadataUsage\_Mapping.getMapped (from)

### 7.7.12.2.9 AssociationMetadataUsageRedefinition\_Mapping

#### Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

#### General Mappings

GenericToRedefinition\_Mapping

#### Mapping Source

Association

#### Mapping Target

Redefinition

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  
SYSML2::AttributeUsage.allInstances()  
->any (m | m.qualifiedName = 'SysMLv1Library::AssociationData::isDerived')

### 7.7.12.2.10 Class\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

A UML4SysML::Class is mapped to a SysML v2 OccurrenceDefinition. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
occurrence def UML4SysMLClass;
```

#### General Mappings

BehavioredClassifier\_Mapping

**Mapping Source**

Class

**Mapping Target**

OccurrenceDefinition

**Owned Mappings**

(none)

**Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
not Helper.isRequirement(src) and not src.oclIsTypeOf(UML::AssociationClass)
```

**Mapping rules**

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.7.12.2.11 ConnectionEndToSubsetting\_Mapping

**[SYSML2-200: Description of Subsetting mapping classes is not correct](#)**

**Description**

Creates a subsetting relationship.

**General Mappings**

GenericToSubsetting\_Mapping

**Mapping Source**

ConnectorEnd

**Mapping Target**

Subsetting

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subsetting::ownedRelationship () : Relationship [0..\*]

```
let propertyPath: OrderedSet(UML::Property) =
 Helper.getTagValueAsElementColl
 (from, 'SysML::Blocks::NestedConnectorEnd', 'propertyPath')
 ->asOrderedSet() in
if propertyPath->notEmpty() then
 OrderedSet{ConnectorEndToSubsettedFeatureMembership_Mapping.getMapped(from)}
else
 OrderedSet{}
endif
```

- Subsetting::subsettedFeature () : Feature [1]

```
let propertyPath: OrderedSet(UML::Property) =
 Helper.getTagValueAsElementColl
 (src, 'SysML::Blocks::NestedConnectorEnd', 'propertyPath')
 ->asOrderedSet() in
if propertyPath->isEmpty() then
 ElementMain_Mapping.getMapped(from.role)
else
 ConnectorEndToSubsettedFeature_Mapping.getMapped(from)
endif
```

- Subsetting::subsettingFeature () : Feature [1]

```
ConnectorEndToOwnedFeature_Mapping.getMapped(from)
```

### 7.7.12.2.12 Connector\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A UML4SysML::Connector is mapped to a SysMLv2 ConnectionUsage. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block3 {
 part sysMLv1PartProperty1 : SysMLv1Block1;
 part sysMLv1PartProperty2 : SysMLv1Block2;
 connection sysMLv1Connector connect sysMLv1PartProperty1 to sysMLv1PartProperty2;
}
part def SysMLv1Block1;
part def SysMLv1Block2;
```

##### General Mappings

NamedElementMain\_Mapping  
GenericToConnector\_Mapping

##### Mapping Source

Connector

### **Mapping Target**

ConnectionUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConnectionUsage::ownedRelationship () : Relationship [0..\*]

```
from.end->collect(e | ConnectorEndToMembership_Mapping.getMapped(e))->asset()
->including(ConnectorMultiplicityMembership_Mapping.getMapped(from))
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

## **7.7.12.2.13 ConnectorEndToFeatureCommon\_Mapping**

### **Description**

The mapping class is the abstract base class for UML4SysML::ConnectorEnd mapping classes.

### **General Mappings**

GenericToFeature\_Mapping

### **Mapping Source**

ConnectorEnd

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isOrdered () : Boolean [1]

```
from.isOrdered
```

#### 7.7.12.2.14 ConnectorEndToMembership\_Mapping

##### Description

Creates a membership relationship for *memberElement()*.

##### General Mappings

GenericToFeatureMembership\_Mapping

##### Mapping Source

ConnectorEnd

##### Mapping Target

EndFeatureMembership

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

```
ConnectorEndToOwnedFeature_Mapping.getMapped(from)
```

#### 7.7.12.2.15 ConnectorEndToOwnedFeature\_Mapping

##### Description

The mapping class creates the SysML v2 Feature element for the UML4SysML::ConnectorEnd mapping.

##### General Mappings

ConnectorEndToFeatureCommon\_Mapping  
ElementMain\_Mapping

##### Mapping Source

ConnectorEnd

##### Mapping Target

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
let subsetting: KerML::Subsetting =
 ConnectionEndToSubsetting_Mapping.getMapped(from) in
if subsetting.oclIsUndefined() then
 OrderedSet{MultiplicityMembership_Mapping.getMapped(from)}
else
 OrderedSet{MultiplicityMembership_Mapping.getMapped(from), subsetting}
endif
```

## **7.7.12.2.16 ConnectorEndToSubsettedFeature\_Mapping**

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

### **Description**

The mapping class maps UML4SysML::ConnectorEnd that are part of a SysML::Ports&Flows::NestedConnectorEnd.

### **General Mappings**

ConnectorEndToFeatureCommon\_Mapping

### **Mapping Source**

ConnectorEnd

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
let propertyPath: OrderedSet(UML::Property) =
Helper.getTagValueAsElementColl(src, 'SysML::Blocks::NestedConnectorEnd', 'propertyPath')
```

```
->asOrderedSet() in
propertyPath->notEmpty()
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::declaredName () : String [0..1]  
  
'featureChain'
- Feature::ownedRelationship () : Relationship [0..\*]  
  
let propertyPath: OrderedSet(UML::Property) =  
 Helper.getTagValueAsElementColl  
(from, 'SysML::Blocks::NestedConnectorEnd', 'propertyPath')  
->asOrderedSet() in  
let chain: OrderedSet(KerML::FeatureChaining) =  
 propertyPath->collect(p | PropertyToFeatureChaining\_Mapping.getMapped(p))  
->asOrderedSet()  
->including(PropertyToFeatureChaining\_Mapping.getMapped(from.role)) in  
chain->union(OrderedSet{MultiplicityMembership\_Mapping.getMapped(from)})

## 7.7.12.2.17 ConnectorEndToSubsettedFeatureMembership\_Mapping

### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

### General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

ConnectorEnd

### Mapping Target

EndFeatureMembership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

```
ConnectorEndToSubsettedFeature_Mapping.getMapped(from)
```

### 7.7.12.2.18 ConnectorMultiplicityMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

DefaultMultiplicityMembership\_Mapping

#### Mapping Source

Connector

#### Mapping Target

OwningMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::memberName () : String [0..1]

```
from.name+'_Connector_multiplicity'
```

### 7.7.12.2.19 ConnectorType\_Mapping

#### Description

A UML4SysML::Association is mapped to a SysML v2 ConnectionDefinition.

#### General Mappings

AssociationCommon\_Mapping

#### Mapping Source

Association

#### Mapping Target

ConnectionDefinition

#### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
let this: UML::Association = src.oclAsType(UML::Association) in
if this.oclIsUndefined() then
 false
else
 not src.memberEnd->exists(m | m.type.oclIsKindOf(UML::UseCase)) and
 not src.isDerived and
 not src.oclIsTypeOf(UML::AssociationClass) and
 Helper.isConnectionDef(src)
endif
```

### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

#### 7.7.12.2.20 ConnectorTypeDerived\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

### Description

The mapping class is a concrete mapping class of the abstract AssociationCommon\_Mapping class for mappings of derived associations. The UML4SysML::Association::isDerived property is not supported in SysML v2. To preserve the information, it is stored in a metadata annotation.

### General Mappings

AssociationCommon\_Mapping

### Mapping Source

Association

### Mapping Target

ConnectionDefinition

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
(src.memberEnd->select(m | m.type.oclIsKindOf(UML::UseCase))->isEmpty()) and
(let this: UML::Association = src.oclAsType(UML::Association) in
if this.oclIsUndefined() then
```

```

 false
else
 this.isDerived and
 not this.oclIsTypeOf(UML::AssociationClass) and
 Helper.isConnectionDef(this)
endif)

```

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConnectionDefinition::ownedRelationship () : Relationship [0..\*]

```

self.oclAsType(AssociationCommon_Mapping).ownedRelationship()
->including(AssociationMetadataUsageMembership_Mapping.getMapped(from))

```

### **7.7.12.2.21 End\_Mapping**

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### **Description**

The mapping class is the abstract base class of mapping classes for properties that are defined by association ends.

##### **General Mappings**

PropertyCommon\_Mapping

##### **Mapping Source**

Property

##### **Mapping Target**

Feature

##### **Owned Mappings**

(none)

##### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```

src.oclIsKindOf(UML::Property) and
not src.oclAsType(UML::Property).association.oclIsUndefined()

```

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::isEnd () : Boolean [1]

true

### 7.7.12.2.22 EndMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

StructuralFeatureMembership\_Mapping

#### Mapping Source

Property

#### Mapping Target

EndFeatureMembership

#### Owned Mappings

(none)

### 7.7.12.2.23 EndToSubsettedFeature\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

The mapping class creates a feature element for the UML4SysML::ConnectorEnd mapping.

#### General Mappings

PropertyCommon\_Mapping

#### Mapping Source

Property

#### Mapping Target

Feature

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
let property: UML::Property = src.oclAsType(UML::Property) in
not property.association.oclIsUndefined()
and property.association.ownedEnd->excludes(property)
```

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
let chain: OrderedSet(KerML::FeatureChaining) =
 OrderedSet{EndToSubsettedFeatureChaining_Mapping.getMapped(from)} in
chain->including(MultiplicityMembership_Mapping.getMapped(from))
```

### 7.7.12.2.24 EndToSubsettedFeatureChaining\_Mapping

**SYSML2-443: Property\_Mapping should map to ItemUsage and the class name is misleading**

#### Description

The mapping class creates a feature chaining element for the UML4SysML::ConnectorEnd mapping.

#### General Mappings

GenericToRelationship\_Mapping

#### Mapping Source

Property

#### Mapping Target

FeatureChaining

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::declaredName () : String [0..1]

```
'featureChain'
```

- FeatureChaining::chainingFeature () : Feature [1]

```
from
```

### 7.7.12.2.25 NonOwnedEndSubsetting\_Mapping

**SYSML2-200: Description of Subsetting mapping classes is not correct**

**SYSML2-443: Property\_Mapping should map to ItemUsage and the class name is misleading**

## Description

Creates a subsetting relationship.

## General Mappings

GenericToSubsetting\_Mapping

### Mapping Source

Property

### Mapping Target

Subsetting

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Subsetting::subsettFeature () : Feature [1]

from

## 7.7.12.2.26 NonOwnedEndToSubsettedFeatureMembership\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

## Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

Property

### Mapping Target

FeatureMembership

## Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
src.oclIsKindOf(UML::Property) and
not src.oclAsType(UML::Property).association.oclIsUndefined()
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
EndToSubsettedFeature_Mapping.getMapped(from)
```

## 7.7.12.2.27 NonOwnedEnd\_Mapping

### Description

The mapping class maps UML4SysML::Property elements that are not owned by an association to a SysML v2 Feature element.

### General Mappings

End\_Mapping

### Mapping Source

Property

### Mapping Target

Feature

### Owned Mappings

- nonOwnedEndTyping : NonOwnedEndFeatureTyping\_Mapping

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{MultiplicityMembership_Mapping.getMapped(from),
nonOwnedEndTyping.to,
NonOwnedEndSubsettingMembership_Mapping.getMapped(from),
```

```

NonOwnedEndToSubsettedFeatureMembership_Mapping.getMapped(from) }
->union(from.qualifier
->collect(q | ElementFeatureMembership_Mapping.getMapped(q)) ->asSet())

```

- Feature::declaredName () : String [0..1]

'nonOwnedEnd'

### **7.7.12.2.28 NonOwnedEndMembership\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

EndMembership\_Mapping

#### **Mapping Source**

Property

#### **Mapping Target**

EndFeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```

src.oclIsKindOf(UML::Property)
and not src.oclAsType(UML::Property).association.oclIsUndefined()
and src.oclAsType(UML::Property).association.ownedEnd->excludes(src)

```

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

NonOwnedEnd\_Mapping.getMapped(from)

### **7.7.12.2.29 NonOwnedEndSubsettingMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

GenericToOwningMembership\_Mapping

**Mapping Source**

Property

**Mapping Target**

OwningMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
NonOwnedEndSubsetting_Mapping.getMapped (from)
```

### 7.7.12.2.30 NonOwnedEndFeatureTyping\_Mapping

**Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

**General Mappings**

StructuralFeatureToFeatureTyping\_Mapping

**Mapping Source**

Property

**Mapping Target**

FeatureTyping

**Owned Mappings**

- nonOwnedEnd : NonOwnedEnd\_Mapping

### 7.7.12.2.31 OwnedEnd\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**Description**

The mapping class maps UML4SysML::Property elements that are owned by an association to a SysML v2 Feature element.

### General Mappings

End\_Mapping  
NamedElementMain\_Mapping

### Mapping Source

Property

### Mapping Target

Feature

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
let p: UML::Property = src.oclAsType(UML::Property) in
not p.oclisUndefined() and
(not p.association.oclisUndefined()
 and p.association.ownedEnd->includes(p)) and
(not p.association.memberEnd
->select(m | (not m.type.oclisUndefined()
 and m.type.oclisTypeOf(UML::UseCase))->notEmpty())
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
let qualifiers: Set(KerML::FeatureMembership) =
 from.qualifier
 ->collect(q | ElementFeatureMembership_Mapping.getMapped(q))->asSet() in
let typing: KerML::FeatureTyping =
 StructuralFeatureToFeatureTyping_Mapping.getMapped(from) in
let subsetting: Set(KerML::Subsetting) =
 from.subsettedProperty
 ->collect(p | PropertySubsetting_Mapping.getMapped(from, p))->asSet() in
let subsettingMultiplicityTyping: Set(KerML::Relationship) =
 subsetting->union(if typing.oclisUndefined() then
 Set{MultiplicityMembership_Mapping.getMapped(from)}
 else
 Set{MultiplicityMembership_Mapping.getMapped(from), typing}
 endif)->asSet() in
let relationships: Set(KerML::Relationship) = qualifiers->union(
 if from.defaultValue.oclisTypeOf(UML::OpaqueExpression) then
 subsettingMultiplicityTyping
```

```

 ->including(ElementOwningMembership_Mapping.getMapped(from.defaultValue))
 else
 subsettingMultiplicityTyping
 endif) in

 if from.defaultValue.oclIsUndefined() then
 relationships
 else
 relationships->including(
 if from.defaultValue.oclIsTypeOf(UML::OpaqueExpression) then
 DefaultValueOpaqueExpression_Mapping.getMapped(from.defaultValue)
 else
 DefaultValue_Mapping.getMapped(from.defaultValue)
 endif)
 endif

```

### 7.7.12.2.32 OwnedEndMembership\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

EndMembership\_Mapping

#### Mapping Source

Property

#### Mapping Target

EndFeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```

src.oclIsKindOf(UML::Property)
and not src.oclAsType(UML::Property).association.oclIsUndefined()
and src.oclAsType(UML::Property).association.ownedEnd->includes(src)

```

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

```
OwnedEnd_Mapping.getMapped(from)
```

### 7.7.12.2.33 Port\_Mapping

**SYSML2-443: Property\_Mapping should map to ItemUsage and the class name is misleading**

#### Description

A UML4SysML::Port that is typed by an interface block is mapped to a SysML v2 PortUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
port sysMLv1Port : SysMLv1InterfaceBlock;
port def SysMLv1InterfaceBlock
```

#### General Mappings

PropertyCommon\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

Port

#### Mapping Target

PortUsage

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
if src.oclIsTypeOf(UML::Port) and
not Helper.hasStereotypeApplied(src.owner,
'SysML::ConstraintBlocks::ConstraintBlock') then
 let p: UML::Port = src.oclAsType(UML::Port) in
 if p.type.oclIsUndefined() then
 false
 else
 true
 endif
 else
 false
 endif
endif
```

#### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.7.12.2.34 PortUntyped\_Mapping

#### Description

A UML4SysML::Port that is untyped is mapped to a SysML v2 PortUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
port sysMLv1Port;
```

### **General Mappings**

PropertyUntyped\_Mapping

#### **Mapping Source**

Port

#### **Mapping Target**

PortUsage

#### **Owned Mappings**

(none)

### **7.7.12.2.35 PropertyToFeatureChaining\_Mapping**

#### **Description**

The mapping class creates the SysML v2 FeatureChaining for the UML4SysML::Property mapping.

### **General Mappings**

GenericToRelationship\_Mapping

#### **Mapping Source**

Property

#### **Mapping Target**

FeatureChaining

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]

```
ElementMain_Mapping.getMapped(from)
```

### 7.7.12.2.36 QualifierMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

StructuralFeatureMembership\_Mapping

#### Mapping Source

StructuralFeature

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

## 7.7.13 UseCases

This chapter lists all mapping specifications of UML4SysML::UseCases model elements.

**SYSML2-513: Missing text in some main mapping sections**

### 7.7.13.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**Table 18. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax     |
|-------------------------------------|------------------------------|
| Actor                               | ItemDefinition               |
| Extend                              | not mapped; see next section |
| ExtensionPoint                      | not mapped; see next section |
| Include                             | IncludeUseCaseUsage          |
| UseCase                             | UseCaseDefinition            |

The following table gives an overview of which SysML v2 elements the UML4SysML::UseCases elements are transformed with which mapping class. The mapping details are in [7.7.13.3](#).

The justifications for the elements without mapping are given in [7.7.13.2](#).

### 7.7.13.2 UML4SysML::UseCases elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566: Section containing tables about elements not mapped should get an introductory text**

**Table 19. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept | Rationale                                                                                                                                                        |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Extend           | The semantics of the UML4SysML::Extend relationship is not supported by SysML v2.                                                                                |
| ExtensionPoint   | The semantics of the UML4SysML::Extend relationship is not supported by SysML v2 Therefore, UML4SysML::ExtensionPoint is also not covered by the transformation. |

### 7.7.13.3 Mapping Specifications

#### 7.7.13.3.1 Actor\_Mapping

##### Description

A UML4SysML::Actor is mapped to a SysML v2 ItemDefinition. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
item def SysMLv1Actor;
```

##### General Mappings

ElementMain\_Mapping  
BehavioredClassifier\_Mapping

##### Mapping Source

Actor

##### Mapping Target

ItemDefinition

##### Owned Mappings

(none)

#### 7.7.13.3.2 Include\_Mapping

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A UML4SysML::Include is mapped to a SysML v2 IncludeUseCaseUsage. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```

use case def SysMLv1UseCase1 {
 include use case : SysMLv1UseCase2;
}
use case def SysMLv1UseCase2;

```

### **General Mappings**

GenericToOccurrenceUsage\_Mapping  
NamedElementMain\_Mapping

#### **Mapping Source**

Include

#### **Mapping Target**

IncludeUseCaseUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- IncludeUseCaseUsage::ownedRelationship () : Relationship [0..\*]

```

Set<IncludeFeatureTyping_Mapping.getMapped(from),
ReturnParameterFeatureMembership_Factory.create(),
EmptySubjectMembership_Factory.create()
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())

```

### **7.7.13.3.3 IncludeFeatureTyping\_Mapping**

#### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

#### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

Include

#### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

from.addition

### **7.7.13.3.4 UseCase\_Mapping**

**[SYSML2-178: ClassifierBehaviorFeatureMembership\\_Mapping does not exist](#)**

#### **Description**

A UML4SysML::UseCase is mapped to a SysML v2 UseCaseDefinition. The expected SysML v2 textual syntax of a mapped UML4SysML::UseCase with a defined subject is as follows.

```
use case def SysMLv1UseCase {
 subject subject_SysMLv1Block : SysMLv1Block;
}
part def SysMLv1Block;
```

Currently, only one use case subject is supported by the mapping class. Since the UML4SysML::Extend relationship is not considered by the SysML v1 to SysML v2 transformation, the extension points of a use case are also not mapped.

#### **General Mappings**

BehavioredClassifier\_Mapping  
NamedElementMain\_Mapping

#### **Mapping Source**

UseCase

#### **Mapping Target**

UseCaseDefinition

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- UseCaseDefinition::ownedRelationship () : Relationship [0..\*]

```

let properties : Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Property) and
 e.oclAsType(UML::Property).association.oclIsUndefined()) in
let actors : Set(UML::Property) =
 UML::Association.allInstances()
 ->collect(m | m.memberEnd)
 ->flatten()
 ->select(m | m.type = from)->collect(a | a.owningAssociation)
 ->collect(p | p.memberEnd->select(m | not (m.type = from)))->flatten() in
let extensionPoints : Sequence(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::ExtensionPoint)) in
let extend : Sequence(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Extend)) in
let include : Sequence(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Include)) in
let elements : Set(UML::Element) =
 (((from.ownedElement-properties) - extensionPoints) - extend) - include) in
let relationships : Sequence(KerML::Relationship) =
elements->collect(e | ElementOwningMembership_Mapping.getMapped(e))
->union(properties->collect(e | PropertyMembership_Mapping.getMapped(e)))
->including(UseCaseSubjectMembership_Mapping.getMapped(from))
->including(UseCaseObjectiveMembership_Mapping.getMapped(from))
->including(CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from))
->union(actors->collect(e | UseCaseActorMembership_Mapping.getMapped(e))) in
if from.classifierBehavior.oclIsUndefined() then
 relationships
else
 relationships
 ->including(BehavioredClassifierFeatureMembership_Mapping.getMapped(from))
endif

```

### 7.7.13.3.5 UseCaseActor\_Mapping

#### Description

The mapping class creates the PartUsage representing an actor of the use case.

#### General Mappings

GenericToPartUsage\_Mapping

#### Mapping Source

Property

#### Mapping Target

PartUsage

#### Owned Mappings

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PartUsage::declaredName () : String [0..1]  
    from.name
- PartUsage::ownedRelationship () : Relationship [0..\*]  
    Set{UseCaseActorFeatureTyping\_Mapping.getMapped(from)}

### **7.7.13.3.6 UseCaseActorFeatureTyping\_Mapping**

#### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

#### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

Property

#### **Mapping Target**

FeatureTyping

#### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  
    from.type

### **7.7.13.3.7 UseCaseActorMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

## **General Mappings**

GenericToActorMembership\_Mapping

### **Mapping Source**

Property

### **Mapping Target**

ActorMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActorMembership::ownedMemberParameter () : Feature [1]

```
UseCaseActor_Mapping.getMapped(from)
```

## **7.7.13.3.8 UseCaseEmptySubjectReferenceUsage\_Mapping**

### **Description**

The mapping class creates an "empty" ReferenceUsage for the subject, if the subject is not given at the SysML v1 UseCase element.

## **General Mappings**

GenericToReferenceUsage\_Mapping

### **Mapping Source**

UseCase

### **Mapping Target**

ReferenceUsage

### **Owned Mappings**

(none)

## **7.7.13.3.9 UseCaseObjectiveMembership\_Mapping**

### **Description**

Creates a membership relationship for *memberElement()*.

### **General Mappings**

GenericToObjectiveMembership\_Mapping

#### **Mapping Source**

UseCase

#### **Mapping Target**

ObjectiveMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ObjectiveMembership::ownedMemberFeature () : Feature [1]  
    UseCaseObjectiveRequirementUsage\_Mapping.getMapped(from)

### **7.7.13.3.10 UseCaseObjectiveRequirementUsage\_Mapping**

#### **Description**

The mapping class creates the RequirementUsage element for the use case objective. The element is not set by an element from the SysML v1 UseCase.

### **General Mappings**

GenericToRequirementUsage\_Mapping

#### **Mapping Source**

UseCase

#### **Mapping Target**

RequirementUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- RequirementUsage::ownedRelationship () : Relationship [0..\*]

```
Set{UseCaseObjectiveSubjectMembership_Mapping.getMapped(from),
CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from)}
```

## 7.7.13.3.11 UseCaseObjectiveSubjectMembership\_Mapping

### Description

Creates a membership relationship for *memberElement()*.

### General Mappings

GenericToSubjectMembership\_Mapping

### Mapping Source

UseCase

### Mapping Target

SubjectMembership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SubjectMembership::ownedMemberParameter () : Feature [1]

```
UseCaseEmptySubjectReferenceUsage_Mapping.getMapped(from)
```

## 7.7.13.3.12 UseCaseSubjectFeatureTyping\_Mapping

### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

### General Mappings

GenericToFeatureTyping\_Mapping

**Mapping Source**

UseCase

**Mapping Target**

FeatureTyping

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  

```
if from.subject->size() > 0 then from.subject->get(0) else invalid endif
```

### 7.7.13.3.13 UseCaseSubjectMembership\_Mapping

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToSubjectMembership\_Mapping

**Mapping Source**

UseCase

**Mapping Target**

SubjectMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SubjectMembership::ownedMemberParameter () : Feature [1]

```
if from.subject->size() > 0 then
 UseCaseSubjectReferenceUsage_Mapping.getMapped(from)
else
 UseCaseEmptySubjectReferenceUsage_Mapping.getMapped(from)
endif
```

### 7.7.13.3.14 UseCaseSubjectReferenceUsage\_Mapping

#### Description

The mapping class creates the ReferenceUsage element for the subject.

#### General Mappings

UseCaseEmptySubjectReferenceUsage\_Mapping

#### Mapping Source

UseCase

#### Mapping Target

ReferenceUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{UseCaseSubjectFeatureTyping_Mapping.getMapped(from)}
```
- ReferenceUsage::declaredName () : String [0..1]

```
'subject_' + from.subject->get(0).name
```

### 7.7.14 Values

This chapter lists all mapping specifications of UML4SysML::Values model elements.

#### **SYSML2-513: Missing text in some main mapping sections**

#### 7.7.14.1 Overview

**SYSML2-441:** Change the table header of the overview tables in the mapping class specification chapters

**SYSML2-564:** Mapping tables in the overview sections show duplicates in the SysML v2 column

Table 20. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax     |
|-------------------------------------|------------------------------|
| Duration                            | not mapped; see next section |
| DurationConstraint                  | ConstraintDefinition         |
| DurationInterval                    | not mapped; see next section |
| DurationObservation                 | not mapped; see next section |
| Expression                          | OperatorExpression           |
| Interval                            | not mapped; see next section |
| IntervalConstraint                  | not mapped; see next section |
| LiteralBoolean                      | LiteralBoolean               |
| LiteralInteger                      | LiteralInteger               |
| LiteralNull                         | NullExpression               |
| LiteralReal                         | LiteralRational              |
| LiteralString                       | LiteralString                |
| LiteralUnlimitedNatural             | LiteralInteger               |
| OpaqueExpression                    | CalculationUsage             |
| StringExpression                    | not mapped; see next section |
| TimeConstraint                      | ConstraintDefinition         |
| TimeExpression                      | TriggerInvocationExpression  |
| TimeInterval                        | not mapped; see next section |
| TimeObservation                     | not mapped; see next section |

The following table gives an overview of which SysML v2 elements the UML4SysML::Values elements are transformed with which mapping class. The mapping details are in [7.7.14.3](#).

The justifications for the elements without mapping are given in [7.7.14.2](#).

#### 7.7.14.2 UML4SysML::Values elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566:** Section containing tables about elements not mapped should get an introductory text

**Table 21. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept    | Rationale                     |
|---------------------|-------------------------------|
| Duration            | Mapping is not specified yet. |
| DurationConstraint  | Mapping is not specified yet. |
| DurationInterval    | Mapping is not specified yet. |
| DurationObservation | Mapping is not specified yet. |
| Interval            | Mapping is not specified yet. |
| IntervalConstraint  | Mapping is not specified yet. |
| StringExpression    | Mapping is not specified yet. |
| TimeConstraint      | Mapping is not specified yet. |
| TimeInterval        | Mapping is not specified yet. |
| TimeObservation     | Mapping is not specified yet. |

### 7.7.14.3 Mapping Specifications

#### 7.7.14.3.1 EqualOperatorExpressionFeature\_Mapping

##### Description

The mapping class creates the feature element for the equal operator.

##### General Mappings

GenericToFeature\_Mapping

##### Mapping Source

TypedElement

##### Mapping Target

Feature

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{EqualOperatorExpressionFeatureValue_Mapping.getMapped(from)}
```

### **7.7.14.3.2 EqualOperatorExpressionFeatureValue\_Mapping**

#### **Description**

Creates a feature value relationship.

#### **General Mappings**

GenericToFeatureValue\_Mapping

#### **Mapping Source**

TypedElement

#### **Mapping Target**

FeatureValue

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
CommonFeatureReferenceExpression\_Mapping.getMapped(from)

### **7.7.14.3.3 EqualOperatorExpressionOperandParameterMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

GenericToParameterMembership\_Mapping

#### **Mapping Source**

TypedElement

#### **Mapping Target**

ParameterMembership

#### **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]  
    EqualOperatorExpressionFeature\_Mapping.getMapped(from)
- ParameterMembership::visibility () : VisibilityKind [1]  
    KerML::VisibilityKind::private

### **7.7.14.3.4 Expression\_Mapping**

#### **Description**

A UML4SysML::Expression element is mapped to a SysML v2 OperatorExpression element.

#### **General Mappings**

GenericToExpression\_Mapping  
NamedElementMain\_Mapping

#### **Mapping Source**

Expression

#### **Mapping Target**

OperatorExpression

#### **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::operator () : String [1]  
    from.symbol

### **7.7.14.3.5 ExpressionElse\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### **Description**

A UML4SysML::Expression element with operator "else" is mapped to a SysML v2 TextualRepresentation element with language set to "SysMLv1" and body set to "else".

#### **General Mappings**

Expression\_Mapping

#### **Mapping Source**

Expression

#### **Mapping Target**

OperatorExpression

#### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.symbol = 'else'
```

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship() -> including(ExpressionElseMembership_
```

### **7.7.14.3.6 ExpressionElseMembership\_Mapping**

#### **Description**

Creates the membership relationship for the textual representation for the else guard condition specification.

#### **General Mappings**

GenericToOwningMembership\_Mapping

#### **Mapping Source**

Expression

**Mapping Target**

OwningMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
ExpressionElseSpecification\_Mapping.getMapped(from)

### **7.7.14.3.7 ExpressionElseSpecification\_Mapping**

**Description**

Creates the textual representation for the else guard condition specification.

**General Mappings**

GenericToTextualRepresentation\_Mapping

**Mapping Source**

Expression

**Mapping Target**

TextualRepresentation

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TextualRepresentation::body () : String [1]  
'else'

- TextualRepresentation::language () : String [1]

'SysMLv1'

### 7.7.14.3.8 LiteralBoolean\_Mapping

#### Description

The mapping class maps UML4SysML::LiteralBoolean to SysML v2 LiteralBoolean.

#### General Mappings

LiteralSpecificationCommon\_Mapping

#### Mapping Source

LiteralBoolean

#### Mapping Target

LiteralBoolean

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralBoolean::value () : Boolean [1]

from.value

### 7.7.14.3.9 LiteralInteger\_Mapping

#### Description

The mapping class maps UML4SysML::LiteralInteger to SysML v2 LiteralInteger.

#### General Mappings

LiteralSpecificationCommon\_Mapping

#### Mapping Source

LiteralInteger

#### Mapping Target

LiteralInteger

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralInteger::value () : Integer [1]  
    from.value

## **7.7.14.3.10 LiteralNull\_Mapping**

### **Description**

The mapping class maps UML4SysML::LiteralNull to SysML v2 NullExpression.

### **General Mappings**

LiteralSpecificationCommon\_Mapping

### **Mapping Source**

LiteralNull

### **Mapping Target**

NullExpression

### **Owned Mappings**

(none)

## **7.7.14.3.11 LiteralReal\_Mapping**

### **Description**

The mapping class maps UML4SysML::LiteralReal to SysML v2 LiteralRational.

### **General Mappings**

LiteralSpecificationCommon\_Mapping

### **Mapping Source**

LiteralReal

### **Mapping Target**

LiteralRational

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralRational::value () : Real [1]

from.value

### **7.7.14.3.12 LiteralSpecificationCommon\_Mapping**

#### **Description**

The mapping class the is abstract base class for all concrete UML4SysML::LiteralSpecification mappings.

#### **General Mappings**

ValueSpecification\_Mapping

#### **Mapping Source**

LiteralSpecification

#### **Mapping Target**

LiteralExpression

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralExpression::ownedRelationship () : Relationship [0..\*]

```
let ownerships: Set(SYMSL2::Relationship) =
 self.oclAsType(ElementMain_Mapping).ownedRelationship()
 ->including(CommonReturnParameterFeatureMembership_Mapping.getMapped(from)) in
 if from.type.oclIsUndefined() then
 ownerships
 else
```

```
 ownerships->including(LiteralSpecificationTyping_Mapping.getMapped(from))
endif
```

### 7.7.14.3.13 LiteralSpecificationFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

TypedElementFeatureTyping\_Mapping

#### Mapping Source

LiteralSpecification

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

### 7.7.14.3.14 LiteralString\_Mapping

#### Description

The mapping class maps UML4SysML::LiteralString to the SysML v2 LiteralString.

#### General Mappings

LiteralSpecificationCommon\_Mapping

#### Mapping Source

LiteralString

#### Mapping Target

LiteralString

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralString::value () : String [1]  
if from.value.oclIsUndefined() then '' else from.value endif

#### 7.7.14.3.15 LiteralUnlimitedUnbounded\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

The mapping class maps UML4SysML::LiteralUnlimited to SysML v2 LiteralInfinity if it is the unlimited value.

##### General Mappings

LiteralUnlimitedInteger\_Mapping

##### Mapping Source

LiteralUnlimitedNatural

##### Mapping Target

LiteralInfinity

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

(from.value = -1)

##### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

#### 7.7.14.3.16 LiteralUnlimitedInteger\_Mapping

##### Description

The mapping class maps UML4SysML::LiteralUnlimited to SysML v2 LiteralInteger if it is not the unlimited value.

##### General Mappings

LiteralSpecificationCommon\_Mapping

##### Mapping Source

LiteralUnlimitedNatural

**Mapping Target**

LiteralInteger

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- LiteralInteger::value () : Integer [1]

from.value

### **7.7.14.3.17 OpaqueExpressionAsValue\_Mapping**

**Description**

The mapping class maps a UML4SysML::OpaqueExpression if it is used as a value to a SysML v2 FeatureChainExpression.

**General Mappings**

GenericToExpression\_Mapping

**Mapping Source**

OpaqueExpression

**Mapping Target**

FeatureChainExpression

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChainExpression::ownedRelationship () : Relationship [0..\*]

```
Set{OpaqueExpressionParameterMembership_Mapping.getMapped(from),
CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
```

### 7.7.14.3.18 OpaqueExpression\_Mapping

#### **SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A UML4SysML::OpaqueExpression element is mapped to a SysMLv2 CalculationUsage element.. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
calc sysMLv1OpaqueExpression {
 return result : ScalarValues::Integer;
 language "Built-in Math"
 /*
 * result = 42 + 23;
 */
}
```

##### General Mappings

CommonAction\_Mapping  
ValueSpecification\_Mapping

##### Mapping Source

OpaqueExpression

##### Mapping Target

CalculationUsage

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- CalculationUsage::ownedRelationship () : Relationship [0..\*]

```
Set{OpaqueExpressionMembership_Mapping.getMapped(from),
OpaqueExpressionReferenceUsageReturnParameterMembership_Mapping.getMapped(from)}
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

### 7.7.14.3.19 OpaqueExpressionFeature\_Mapping

##### Description

The mapping class creates the feature of the FeatureChainExpression.

### **General Mappings**

GenericToFeature\_Mapping

### **Mapping Source**

OpaqueExpression

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{OpaqueExpressionFeatureValue_Mapping.getMapped(from),
OpaqueExpressionFeatureMembership_Mapping.getMapped(from)}
```

## **7.7.14.3.20 OpaqueExpressionFeatureFeature\_Mapping**

### **Description**

The mapping class creates the Feature of the FeatureReferenceExpression.

### **General Mappings**

GenericToFeature\_Mapping

### **Mapping Source**

OpaqueExpression

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

## **7.7.14.3.21 OpaqueExpressionFeatureFeatureMembership\_Mapping**

## Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

OpaqueExpression

### Mapping Target

FeatureMembership

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
OpaqueExpressionFeatureFeature_Mapping.getMapped(from)
```

## 7.7.14.3.22 OpaqueExpressionFeatureValue\_Mapping

## Description

Creates a feature value relationship.

## General Mappings

GenericToFeatureValue\_Mapping

### Mapping Source

OpaqueExpression

### Mapping Target

FeatureValue

## Owned Mappings

(none)

## Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
OpaqueExpressionFeatureValueExpression_Mapping.getMapped(from)
```

### 7.7.14.3.23 OpaqueExpressionFeatureValueExpression\_Mapping

**SYSML2-174: EmptyReturnParameterFeatureMembership\_Mapping does not exist**

#### Description

The mapping class creates the value of the FeatureChainExpression that is a FeatureReferenceExpression.

#### General Mappings

GenericToExpression\_Mapping

#### Mapping Source

OpaqueExpression

#### Mapping Target

FeatureReferenceExpression

#### Owned Mappings

(none)

#### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..\*]

```
Set{OpaqueExpressionFeatureValueExpressionMembership_Mapping.getMapped(from),
ReturnParameterFeatureMembership_Factory.create()}
```

### 7.7.14.3.24 OpaqueExpressionFeatureValueExpressionMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToMembership\_Mapping

**Mapping Source**

OpaqueExpression

**Mapping Target**

Membership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Membership::memberElement () : Element [1]

from

### 7.7.14.3.25 OpaqueExpressionMembership\_Mapping

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToOwningMembership\_Mapping

**Mapping Source**

OpaqueExpression

**Mapping Target**

OwningMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
OpaqueExpressionSpecification\_Mapping.getMapped(from)

### 7.7.14.3.26 OpaqueExpressionParameterMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToParameterMembership\_Mapping

#### Mapping Source

OpaqueExpression

#### Mapping Target

ParameterMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ParameterMembership::ownedMemberParameter () : Feature [1]  
OpaqueExpressionFeature\_Mapping.getMapped(from)

### 7.7.14.3.27 OpaqueExpressionReferenceUsageReturnParameterMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToReturnParameterMembership\_Mapping

#### Mapping Source

OpaqueExpression

### **Mapping Target**

ReturnParameterMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReturnParameterMembership::ownedMemberParameter () : Feature [1]

```
if from.type.oclIsUndefined() then
 OpaqueExpressionReferenceUsageUntyped_Mapping.getMapped(from)
else
 OpaqueExpressionReferenceUsage_Mapping.getMapped(from)
endif
```

## **7.7.14.3.28 OpaqueExpressionReferenceUsage\_Mapping**

### **Description**

The mapping class creates the return parameter reference usage of the calculation usage.

### **General Mappings**

GenericToReferenceUsage\_Mapping

### **Mapping Source**

OpaqueExpression

### **Mapping Target**

ReferenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]  
Set{OpaqueExpressionReferenceUsageFeatureTyping\_Mapping.getMapped(from)}
- ReferenceUsage::direction () : FeatureDirectionKind [0..1]  
KerML::FeatureDirectionKind::\_'out'

#### **7.7.14.3.29 OpaqueExpressionReferenceUsageFeatureTyping\_Mapping**

##### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

##### **General Mappings**

TypedElementFeatureTyping\_Mapping

##### **Mapping Source**

OpaqueExpression

##### **Mapping Target**

FeatureTyping

##### **Owned Mappings**

(none)

#### **7.7.14.3.30 OpaqueExpressionReferenceUsageUntyped\_Mapping**

##### **Description**

The mapping class creates the return parameter reference usage of the calculation usage, if the UML4SysML::OpaqueExpression is untyped.

##### **General Mappings**

GenericToReferenceUsage\_Mapping

##### **Mapping Source**

OpaqueExpression

##### **Mapping Target**

ReferenceUsage

##### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

```
KerML::FeatureDirectionKind::_out'
```

## **7.7.14.3.31 OpaqueExpressionSpecification\_Mapping**

### **Description**

The mapping class creates the specification of the calculation usage based on the language and body of the UML4SysML::OpaqueExpression.

### **General Mappings**

GenericToTextualRepresentation\_Mapping

### **Mapping Source**

OpaqueExpression

### **Mapping Target**

TextualRepresentation

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TextualRepresentation::body () : String [1]

```
if from.body->size() = 0 then invalid else from.body.get(0) endif
```

- TextualRepresentation::language () : String [1]

```
if from.language->size() = 0 then invalid else from.language.get(0) endif
```

## **7.7.14.3.32 TimeExpression\_Mapping**

### **Description**

A UML4SysML::TimeExpression is mapped to a SysML v2 TriggerInvocationExpression. The details of the mapping are not specified yet.

### General Mappings

ValueSpecification\_Mapping

#### Mapping Source

TimeExpression

#### Mapping Target

TriggerInvocationExpression

### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- TriggerInvocationExpression::kind () : TriggerKind [1]

SysMLv2::TriggerKind::at

### 7.7.14.3.33 ValueSpecification\_Mapping

#### **[SYSML2-280: ElementMain\\_Mapping::ownedRelationship is wrong](#)**

##### Description

The mapping class is the abstract base class of all mapping classes for special value specifications.

### General Mappings

NamedElementMain\_Mapping  
GenericToExpression\_Mapping

#### Mapping Source

ValueSpecification

#### Mapping Target

Expression

### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Expression::ownedRelationship () : Relationship [0..\*]

```
if from.type.oclIsUndefined() then
 Set{CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
else
 Set{LiteralSpecificationTyping_Mapping.getMapped(from),
 CommonReturnParameterFeatureMembership_Mapping.getMapped(from)}
endif) ->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

## 7.8 Mappings from SysML v1.7 stereotypes

### 7.8.1 Overview

The following subclauses of Mappings from SysML v1.7 stereotypes are organized according to the main packages of SysML v1.

### 7.8.2 Activities

This chapter lists all mapping specifications of SysML::Activities model elements.

#### 7.8.2.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

Table 22. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax |
|-------------------------------------|--------------------------|
| Continuous                          | MetadataUsage            |
| ControlOperator                     |                          |
| Discrete                            | MetadataUsage            |
| NoBuffer                            |                          |
| Optional                            |                          |
| Overwrite                           |                          |
| Probability                         | MetadataUsage            |
| Rate                                | MetadataUsage            |

The following table gives an overview of which SysML v2 elements the SysML::Activities elements are transformed with which mapping class. The mapping details are specified in [7.8.2.3](#).

The justifications for the elements without mapping are given in [7.8.2.2](#).

### 7.8.2.2 SysML::Activities elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566: Section containing tables about elements not mapped should get an introductory text**

**Table 23. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept | Rationale                                                                                                                                                                                                                                                                 |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ControlOperator  | The concept that an action can control other actions is not supported by SysML v2.                                                                                                                                                                                        |
| NoBuffer         | Mapping is not specified yet.                                                                                                                                                                                                                                             |
| Optional         | The stereotype states that the lower multiplicity of the parameter is 0. Since the multiplicity of the parameter is transformed, the additional statement that the parameter is optional is redundant. Therefore, the stereotype is not considered in the transformation. |
| Overwrite        | Mapping is not specified yet.                                                                                                                                                                                                                                             |

### 7.8.2.3 Mapping Specifications

#### 7.8.2.3.1 ProbabilityMetadataUsage\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

A SysML::Activities::Probability is mapped to a SysML v2 MetadataUsage owned by the appropriate target element of the UML4SysML::ActivityEdge or UML4SysML::ParameterSet.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 action sysMLv1Action1;
 succession sysMLv1ControlFlow1 first sysMLv1Action1 then sysMLv1Action2 {
 @SysMLv1Library::ProbabilityData {probability = 0.42;}
 }
 action sysMLv1Action2;
}
```

##### General Mappings

GenericToMetadataUsage\_Mapping

##### Mapping Source

Element

## **Mapping Target**

MetadataUsage

## **Owned Mappings**

(none)

## **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Probability')
```

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ProbabilityMetadataUsageFeatureTyping_Mapping.getMapped(from),
ProbabilityMetadataUsageFeatureMembership_Mapping.getMapped(from)}
```

### **7.8.2.3.2 ProbabilityMetadataUsageFeatureMembership\_Mapping**

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

## **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

## **General Mappings**

GenericToFeatureMembership\_Mapping

## **Mapping Source**

Element

## **Mapping Target**

FeatureMembership

## **Owned Mappings**

(none)

## **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Probability')
```

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
ProbabilityMetadataUsageReferenceUsage\_Mapping.getMapped(from)

### 7.8.2.3.3 ProbabilityMetadataUsageFeatureTyping\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

Element

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Probability')
```

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  
SYSML2::MetadataDefinition.allInstances()  
->any(m | m.qualifiedName = 'SysMLv1Library::ProbabilityData')

### 7.8.2.3.4 ProbabilityMetadataUsageReferenceUsage\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

Creates a reference usage.

### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

Element

#### Mapping Target

ReferenceUsage

### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Probability')
```

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ProbabilityMetadataUsageReferenceUsageRedefinition_Mapping.getMapped(from),
ProbabilityMetadataUsageReferenceUsageFeatureValue_Mapping.getMapped(from)}
```

### 7.8.2.3.5 ProbabilityMetadataUsageReferenceUsageFeatureValue\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

Creates a feature value relationship.

### General Mappings

GenericToFeatureValue\_Mapping

#### Mapping Source

Element

#### Mapping Target

FeatureValue

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Probability')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
let probability : OclAny =
Helper.getTagValue(from, 'SysML::Activities::Probability', 'probability') in
LiteralRational_Factory.create(probability)
```

## 7.8.2.3.6 ProbabilityMetadataUsageReferenceUsageRedefinition\_Mapping

### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

Creates a redefinition relationship for the `redefiningFeature()` and the `redefinedFeature()`.

#### General Mappings

GenericToRedefinition\_Mapping

#### Mapping Source

Element

#### Mapping Target

Redefinition

#### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Probability')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::ProbabilityData::probability')
```

### 7.8.2.3.7 ProbabilityOwningMembership\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### **Description**

Creates a owning membership relationship for *ownedMemberElement()*.

##### **General Mappings**

GenericToOwningMembership\_Mapping

##### **Mapping Source**

Element

##### **Mapping Target**

OwningMembership

##### **Owned Mappings**

(none)

##### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Probability')
```

##### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  

```
ProbabilityMetadataUsage_Mapping.getMapped(from)
```

### 7.8.2.3.8 RateMetadataUsage\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### **Description**

A SysML::Activities::Rate and the specializations SysML::Activities::Discrete and SysML::Activities::Continuous are mapped to a SysML v2 MetadataUsage owned by the appropriate target element of the UML4SysML::ActivityEdge or UML4SysML::Parameter.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
succession flow sysMLv1ObjectFlow of SysMLv1Block
 from sysMLv1Action1.outputValue to sysMLv1Action1.inputValue {
 @SysMLv1Library::RateData {isDiscrete = true;}
 }
```

The mapping of the rate instance value is not supported yet.

## General Mappings

GenericToMetadataUsage\_Mapping

### Mapping Source

Element

### Mapping Target

MetadataUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Rate')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Continuous')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Discrete')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
let relationships : Set(KerML::Relationship) =
 Set{RateMetadataUsageFeatureTyping_Mapping.getMapped(from)} in
if Helper.hasStereotypeApplied(from, 'SysML::Activities::Discrete') then
 relationships
->including(
 RateMetadataUsageDiscreteFeatureMembership_Mapping.getMapped(from))
else if Helper.hasStereotypeApplied(from, 'SysML::Activities::Continuous') then
 relationships
->including(
```

```

 RateMetadataUsageContinuousFeatureMembership_Mapping.getMapped(from)
 else
 relationships
 endif
endif

```

### **7.8.2.3.9 RateMetadataUsageContinuousFeatureMembership\_Mapping**

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

Element

#### **Mapping Target**

FeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Continuous')
```

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
RateMetadataUsageContinuousReferenceUsage_Mapping.getMapped(from)
```

### **7.8.2.3.10 RateMetadataUsageFeatureValue\_Mapping**

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

#### **Description**

Creates a feature value relationship.

#### **General Mappings**

GenericToFeatureValue\_Mapping

**Mapping Source**

Element

**Mapping Target**

FeatureValue

**Owned Mappings**

(none)

**Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Rate')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Continuous')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Discrete')
```

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
    LiteralBoolean\_Factory.create(true)

### 7.8.2.3.11 RateMetadataUsageContinuousReferenceUsage\_Mapping

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

**Description**

Creates a reference usage.

**General Mappings**

GenericToReferenceUsage\_Mapping

**Mapping Source**

Element

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

## Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Continuous')
```

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `ReferenceUsage::ownedRelationship () : Relationship [0..*]`

```
Set{RateMetadataUsageContinuousReferenceUsageRedefinition_Mapping.getMapped(from),
RateMetadataUsageFeatureValue_Mapping.getMapped(from)}
```

### 7.8.2.3.12 RateMetadataUsageContinuousReferenceUsageRedefinition\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

Creates a redefinition relationship for the `redefiningFeature()` and the `redefinedFeature()`.

##### General Mappings

GenericToRedefinition\_Mapping

##### Mapping Source

Element

##### Mapping Target

Redefinition

##### Owned Mappings

(none)

## Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Continuous')
```

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `Redefinition::redefinedFeature () : Feature [1]`

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::RateData::isContinuous')
```

### 7.8.2.3.13 RateMetadataUsageDiscreteFeatureMembership\_Mapping

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

Element

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Discrete')
```

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
  
RateMetadataUsageDiscreteReferenceUsage\_Mapping.getMapped(from)

### 7.8.2.3.14 RateMetadataUsageDiscreteReferenceUsage\_Mapping

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

#### Description

Creates a reference usage.

#### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

Element

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

**Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Discrete')
```

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{RateMetadataUsageDiscreteReferenceUsageRedefinition_Mapping.getMapped(from),
RateMetadataUsageFeatureValue_Mapping.getMapped(from)}
```

### 7.8.2.3.15 RateMetadataUsageDiscreteReferenceUsageRedefinition\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**Description**

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

**General Mappings**

GenericToRedefinition\_Mapping

**Mapping Source**

Element

**Mapping Target**

Redefinition

**Owned Mappings**

(none)

**Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Discrete')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::RateData::isDiscrete')
```

### 7.8.2.3.16 RateMetadataUsageFeatureTyping\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

Element

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Rate')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Continuous')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Discrete')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  

```
SYSML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::RateData')
```

### **7.8.2.3.17 RateOwningMembership\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### **Description**

Creates a owning membership relationship for *ownedMemberElement()*.

#### **General Mappings**

GenericToOwningMembership\_Mapping

#### **Mapping Source**

Element

#### **Mapping Target**

OwningMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Activities::Rate')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Continuous')
or Helper.hasStereotypeApplied(src, 'SysML::Activities::Discrete')
```

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
RateMetadataUsage\_Mapping.getMapped(from)

### **7.8.2.3.18 Model Libraries**

#### **7.8.2.3.18.1 ControlValues**

##### **7.8.2.3.18.1.1 ControlValueKind**

The enumeration ControlValueKind is mapped to the SysML v2 enumeration definition SysMLv1Library::Enumerations::ControlValueKind (see [7.3.2](#)).

## **7.8.3 Allocations**

This chapter lists all mapping specifications of SysML::Allocations model elements.

### 7.8.3.1 Overview

**SYSML2-441:** Change the table header of the overview tables in the mapping class specification chapters

Table 24. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax |
|-------------------------------------|--------------------------|
| Allocate                            | AllocationUsage          |
| AllocateActivityPartition           |                          |

The following table gives an overview of which SysML v2 elements the SysML::Allocations elements are transformed with which mapping class. The mapping details are in [7.8.3.3](#).

The justifications for the elements without mapping are given in [7.8.3.2](#).

### 7.8.3.2 SysML::Allocations elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566:** Section containing tables about elements not mapped should get an introductory text

Table 25. List of SysML v1 elements not mapped of this section

| SysML v1 Concept          | Rationale                     |
|---------------------------|-------------------------------|
| AllocateActivityPartition | Mapping is not specified yet. |

### 7.8.3.3 Mapping Specifications

#### 7.8.3.3.1 Allocation\_Mapping

**SYSML2-258:** Mapping of allocation between usage and definition or definition and usage elements does not work

**SYSML2-7:** Pin\_Mapping::filter: property src should be from

**SYSML2-280:** ElementMain\_Mapping::ownedRelationship is wrong

**SYSML2-88:** Mapping of allocation between usage elements is not specified yet

#### Description

A SysML::Allocations::Allocate is mapped to a SysML v2 AllocationDefinition if it is an allocation between definition elements.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
action def SysMLv1Activity {
 action sysMLv1Action;
}
part def SysMLv1Block {
 part sysMLv1PartProperty : AnotherSysMLv1Block;
}
part def AnotherSysMLv1Block;
```

```

// Allocation of definition
allocation def SysMLv1Allocation {
 end :>> source : SysMLv1Activity;
 end :>> target : SysMLv1Block;
}

// Allocation of usage
allocation def {
 end :>> source : SysMLv1Activity;
 end :>> target : SysMLv1Block;
 allocate source.sysMLv1Action to target.sysMLv1PartProperty;
}
// Allocation of usage to definition
allocation def {
 end :>> source : SysMLv1Activity;
 end :>> target : SysMLv1Block;
 allocate source.sysMLv1Action to target;
}

```

## General Mappings

Abstraction\_Mapping

### Mapping Source

Abstraction

### Mapping Target

AllocationDefinition

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
(Helper.hasStereotypeApplied(src, 'SysML::Allocations::Allocate'))
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- AllocationDefinition::ownedRelationship () : Relationship [0..\*]

```

let relationships : Set(KerML::Relationship) =
 Set{AllocationSourceFeatureMembership_Mapping.getMapped(from.client.get(0)),
 AllocationTargetFeatureMembership_Mapping.getMapped(from.supplier.get(0)) }
 ->union(self.oclastype(ElementMain_Mapping).ownedRelationship())
 if from.client.get(0).oclIsKindOf(UML::Type) then

```

```

 relationships
 else
 relationships->including(AllocationUsageFeatureMembership_Mapping.getMapped(from))
 endif

```

### 7.8.3.3.2 AllocationFeatureMembership\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

NamedElement

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
AllocationSourceReferenceUsage_Mapping.getMapped(from)
```

### 7.8.3.3.3 AllocationFeatureTyping\_Mapping

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

**Mapping Source**

NamedElement

**Mapping Target**

FeatureTyping

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if from.oclIsKindOf(UML::Type) then
 from
else
 from.owner
endif
```

### 7.8.3.3.4 AllocationReferenceUsage\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

**Description**

Creates a reference usage.

**General Mappings**

GenericToReferenceUsage\_Mapping  
UniqueMapping

**Mapping Source**

NamedElement

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::isEnd () : Boolean [1]  
true
- ReferenceUsage::ownedRelationship () : Relationship [0..\*]  

```
Set<AllocationFeatureTyping_Mapping.getMapped(from),
AllocationSourceReferenceUsageRedefinition_Mapping.getMapped(from) }
```

### **7.8.3.3.5 AllocationSourceReferenceUsageRedefinition\_Mapping**

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

#### **Description**

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

#### **General Mappings**

GenericToRedefinition\_Mapping

#### **Mapping Source**

NamedElement

#### **Mapping Target**

Redefinition

#### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  

```
SYSML2::ReferenceUsage.allInstances()
->any(m | m.qualifiedName = 'Allocations::Allocation::source')
```

### **7.8.3.3.6 AllocationTargetFeatureMembership\_Mapping**

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

NamedElement

#### **Mapping Target**

FeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
AllocationTargetReferenceUsage\_Mapping.getMapped(from)

### **7.8.3.3.7 AllocationTargetReferenceUsage\_Mapping**

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

#### **Description**

Creates a reference usage.

#### **General Mappings**

GenericToReferenceUsage\_Mapping  
UniqueMapping

#### **Mapping Source**

NamedElement

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::isEnd () : Boolean [1]  
    true
- ReferenceUsage::ownedRelationship () : Relationship [0..\*]  

```
Set<AllocationFeatureTyping_Mapping.getMapped(from),
AllocationTargetReferenceUsageRedefinition_Mapping.getMapped(from) }
```

### 7.8.3.3.8 AllocationTargetReferenceUsageRedefinition\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**Description**

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

**General Mappings**

GenericToRedefinition\_Mapping

**Mapping Source**

NamedElement

**Mapping Target**

Redefinition

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  

```
SYSML2::ReferenceUsage.allInstances()
->any(m | m.qualifiedName = 'Allocations::Allocation::target')
```

### 7.8.3.3.9 AllocationUsage\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

#### Description

A SysML::Allocations::Allocate is mapped to a SysML v2 AllocationUsage owned by a AllocationDefinition if a usage element is source or target of the allocation relationship.

#### General Mappings

GenericToUsage\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

AllocationUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- AllocationUsage::ownedRelationship () : Relationship [0..\*]  

```
Set{AllocationUsageSourceEndFeatureMembership_Mapping.getMapped(from.client.get(0)),
AllocationUsageTargetEndFeatureMembership_Mapping.getMapped(from.target.get(0))}
```

### 7.8.3.3.10 AllocationUsageEndFeatureMembership\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

## Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## General Mappings

GenericToEndFeatureMembership\_Mapping

### Mapping Source

NamedElement

### Mapping Target

EndFeatureMembership

## Owned Mappings

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

AllocationUsageSourceFeature\_Mapping.getMapped (from)

### 7.8.3.3.11 AllocationUsageFeature\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

## Description

Creates a feature element as an end of the allocation usage relationship.

## General Mappings

GenericToFeature\_Mapping

### Mapping Source

NamedElement

### Mapping Target

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{AllocationUsageSourceFeatureSubsetting_Mapping.getMapped(from)}
```

### **7.8.3.3.12 AllocationUsageFeatureChaining\_Mapping**

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

### **Description**

Creates the first feature chaining element for the subsetting feature for the feature element which represents an end of the allocation usage relationship.

### **General Mappings**

GenericToFeatureChaining\_Mapping

### **Mapping Source**

NamedElement

### **Mapping Target**

FeatureChaining

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]

```
AllocationSourceReferenceUsage_Mapping.getMapped(from)
```

### **7.8.3.3.13 AllocationUsageFeatureChainingChainedFeature\_Mapping**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

#### **Description**

Creates the second feature chaining element for the subsetting feature for the feature element which represents an end of the allocation usage relationship.

#### **General Mappings**

GenericToFeatureChaining\_Mapping

#### **Mapping Source**

NamedElement

#### **Mapping Target**

FeatureChaining

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]

from

### **7.8.3.3.14 AllocationUsageFeatureMembership\_Mapping**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

Abstraction

#### **Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

AllocationUsage\_Mapping.getMapped(from)

### 7.8.3.3.15 AllocationUsageFeatureSubsetting\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

**Description**

Creates a subsetting relationship.

**General Mappings**

GenericToReferenceSubsetting\_Mapping

**Mapping Source**

NamedElement

**Mapping Target**

ReferenceSubsetting

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::ownedRelatedElement () : Element [0..\*]

```

if from.oclIsKindOf(UMLO::Type) then
 Set{}
else
 Set{AllocationUsageSourceFeatureSubsettingFeature_Mapping.getMapped(from)}
endif

```

### **7.8.3.3.16 AllocationUsageFeatureSubsettingFeature\_Mapping**

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**SYSML2-88: Mapping of allocation between usage elements is not specified yet**

#### **Description**

Creates the subsetting feature for the feature element which represents an end of the allocation usage relationship.

#### **General Mappings**

GenericToFeature\_Mapping

#### **Mapping Source**

NamedElement

#### **Mapping Target**

Feature

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```

Set{AllocationUsageSourceFeatureChaining_Mapping.getMapped(from),
AllocationUsageFeatureChainingChainedFeature_Mapping.getMapped(from)}

```

### **7.8.3.3.17 AllocationUsageTargetEndFeatureMembership\_Mapping**

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToEndFeatureMembership\_Mapping

**Mapping Source**

NamedElement

**Mapping Target**

EndFeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]  
AllocationUsageTargetFeature\_Mapping.getMapped(from)

### 7.8.3.3.18 AllocationUsageTargetFeature\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

**Description**

Creates a feature element as an end of the allocation usage relationship.

**General Mappings**

GenericToFeature\_Mapping

**Mapping Source**

NamedElement

**Mapping Target**

Feature

**Owned Mappings**

(none)

**Applicable filters**

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{AllocationUsageTargetFeatureSubsetting_Mapping.getMapped(from)}
```

### 7.8.3.3.19 AllocationUsageTargetFeatureChaining\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

#### Description

Creates the first feature chaining element for the subsetting feature for the feature element which represents an end of the allocation usage relationship.

#### General Mappings

GenericToFeatureChaining\_Mapping

#### Mapping Source

NamedElement

#### Mapping Target

FeatureChaining

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]

```
AllocationTargetReferenceUsage_Mapping.getMapped(from)
```

### 7.8.3.3.20 AllocationUsageTargetFeatureSubsetting\_Mapping

**SYSML2-258: Mapping of allocation between usage and definition or definition and usage elements does not work**

#### Description

Creates a subsetting relationship.

## **General Mappings**

GenericToReferenceSubsetting\_Mapping

### **Mapping Source**

NamedElement

### **Mapping Target**

ReferenceSubsetting

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::ownedRelatedElement () : Element [0..\*]

```
if from.oclIsKindOf(UML::Type) then
 Set{}
else
 Set{AllocationUsageTargetFeatureSubsettingFeature_Mapping.getMapped(from)}
endif
```

## **7.8.3.3.21 AllocationUsageTargetFeatureSubsettingFeature\_Mapping**

**SYSML2-258: Mapping of allcation between usage and definition or definition and usage elements does not work**

### **Description**

Creates the subsetting feature for the feature element which represents an end of the allocation usage relationship.

## **General Mappings**

GenericToFeature\_Mapping

### **Mapping Source**

NamedElement

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{AllocationUsageTargetFeatureChaining_Mapping.getMapped(from),
AllocationUsageFeatureChainingChainedFeature_Mapping.getMapped(from) }
```

## **7.8.4 Blocks**

This chapter lists all mapping specifications of SysML::Blocks model elements.

### **7.8.4.1 Overview**

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**SYSML2-446: Document how SysML v1 properties are mapped to SysML v2**

**SYSML2-564: Mapping tables in the overview sections show duplicates in the SysML v2 column**

**Table 26. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax         |
|-------------------------------------|----------------------------------|
| AdjunctProperty                     |                                  |
| BindingConnector                    | BindingConnectorAsUsage          |
| Block                               | PartDefinition<br>PartDefinition |
| BoundReference                      |                                  |
| ClassifierBehaviorProperty          |                                  |
| ConnectorProperty                   |                                  |
| DistributedProperty                 |                                  |
| EndPathMultiplicity                 |                                  |
| NestedConnectorEnd                  |                                  |
| ParticipantProperty                 |                                  |
| PropertySpecificType                |                                  |
| ValueType                           | AttributeDefinition              |

The following table gives an overview of which SysML v2 elements the SysML::Blocks elements are transformed with which mapping class. The mapping details are in [7.8.4.3](#)

SysML v1 defines special property concepts, but they are not stereotypes or metamodel elements and thus do not all have an explicit mapping class. The following table shows how they are mapped.

| SysML v1 Property Concept                     | SysML v2 Element                       | Main Mapping Class                    |
|-----------------------------------------------|----------------------------------------|---------------------------------------|
| Property typed by a Class or Interface        | OccurrenceUsage with isComposite=false | PropertyTypedByClassInterface_Mapping |
| Part Property                                 | PartUsage with isComposite=true        | PartProperty_Mapping                  |
| Value Property                                | AttributeUsage with isComposite=true   | Attribute_Mapping                     |
| ConstraintProperty                            | AssertConstraintUsage                  | not defined yet                       |
| ReferenceProperty typed by a Block            | PartUsage with isComposite=false       | PartProperty_Mapping                  |
| ReferenceProperty typed by a ValueType        | AttributeUsage with isComposite=false  | Attribute_Mapping                     |
| ReferenceProperty typed by Class or Interface | OccurrenceUsage with isComposite=false | PropertyTypedByClassInterface_Mapping |

The justifications for the elements without mapping are given in [7.8.4.2](#).

#### 7.8.4.2 SysML::Blocks elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566: Section containing tables about elements not mapped should get an introductory text**

**Table 27. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept                 | Rationale                                                                                                                                                                                         |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AdjunctProperty                  | The concept of adjunct properties is not needed in SysML v2, where the principal of the adjunct property can be used directly in the appropriate place.                                           |
| BoundReference                   | Mapping is not specified yet.                                                                                                                                                                     |
| ClassifierBehaviorProperty       | The classifier behavior is already mapped to a property which also plays the role of the classifier behavior property. Therefore, there is no explicit mapping of a classifier behavior property. |
| ConnectorProperty                | The connector property is a special case of an adjunct property and is not mapped, just like the adjunct property.                                                                                |
| DirectedRelationshipPropertyPath | The stereotype is abstract is therefore not mapped. The concept of the DirectedRelationshipPropertyPath is included in the SysML v2 language.                                                     |
| DistributedProperty              | Mapping is not specified yet.                                                                                                                                                                     |

| SysML v1 Concept     | Rationale                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------|
| ElementPropertyPath  | The stereotype is abstract is therefore not mapped. The concept of the ElementPropertyPath is included in the SysML v2 language. |
| EndPathMultiplicity  | Mapping is not specified yet.                                                                                                    |
| NestedConnectorEnd   | The concept of NestedConnectorEnd is already included in the SysML v2 language. It is not required to do an explicit mapping.    |
| ParticipantProperty  | Mapping is not specified yet.                                                                                                    |
| PropertySpecificType | Mapping is not specified yet.                                                                                                    |

### 7.8.4.3 Mapping Specifications

#### 7.8.4.3.1 AssociationBlock\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

An AssociationBlock is mapped to a SysML v2 ConnectionDefinition.

The SysML::Blocks::ParticipantProperties transformation is not defined yet. Therefore, the mapping is currently identical with the mapping of UML4SysML::AssociationClass.

##### General Mappings

AssociationClass\_Mapping

##### Mapping Source

AssociationClass

##### Mapping Target

ConnectionDefinition

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Blocks::Block')
```

##### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.8.4.3.2 BindingConnector\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

A SysML::Blocks::BindingConnector is mapped to a SysML v2 BindingConnectorAsUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block1 {
 part sysMLv1PartProperty1 : SysMLv1Block2;
 part sysMLv1PartProperty2 : SysMLv1Block2;

 binding sysMLv1BindingConnector
 bind sysMLv1PartProperty1 = sysMLv1PartProperty2;
}
part def SysMLv1Block2;
```

#### General Mappings

Connector\_Mapping

#### Mapping Source

Connector

#### Mapping Target

BindingConnectorAsUsage

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Blocks::BindingConnector')
```

#### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.8.4.3.3 Block\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

A SysML::Blocks::Block is mapped to a SysML v2 PartDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part definition SysMLv1Block;
```

### General Mappings

Class\_Mapping

#### Mapping Source

Class

#### Mapping Target

PartDefinition

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
not src.oclIsTypeOf(UML::AssociationClass)
 and Helper.hasStereotypeApplied(src, 'SysML::Blocks::Block')
 and not Helper.hasStereotypeApplied(src, 'SysML::ConstraintBlocks::ConstraintBlock')
 and not Helper.hasStereotypeApplied(src, 'SysML::Ports&Flows::InterfaceBlock')
```

#### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.8.4.3.4 EncapsulatedBlock\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-178: ClassifierBehaviorFeatureMembership\_Mapping does not exist**

#### Description

A SysML::Block with *isEncapsulated=true* is mapped to a SysML v2 PartDefinition, and, additionally, gets a metadata feature defined by the SysML v1 library which represents the SysML v1 *isEncapsulated* property.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1EncapsulatedBlock {
 @SysMLv1Library::BlockData {isEncapsulated = true;}
}
```

### General Mappings

## Block\_Mapping

### Mapping Source

Class

### Mapping Target

PartDefinition

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
not src.oclIsTypeOf(UML::AssociationClass) and
 Helper.hasStereotypeApplied(src, 'SysML::Blocks::Block') and
 not Helper.hasStereotypeApplied(src, 'SysML::ConstraintBlocks::ConstraintBlock') and
 not Helper.hasStereotypeApplied(src, 'SysML::Ports&Flows::InterfaceBlock') and
 Helper.getTagValue(src, 'SysML::Blocks::Block', 'isEncapsulated')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PartDefinition::ownedRelationship () : Relationship [0..\*]

```
let toElementFMS: Set(UML::Element) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Property) and
 (e.oclaType(UML::Property).redefinedProperty->size() = 0)) in
let redefinedAttributes: Set(UML::Element) =
 from.ownedElement->select(e | from.oclIsKindOf(UML::DataType) and
 (e.oclaType(UML::Property).redefinedProperty->size() > 0)) in
let generalizations : Set(UML::Generalization) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Generalization)) in
let toElementOMS: Set(UML::Element) =
 (((from.ownedElement - toElementFMS) - redefinedAttributes) -
 generalizations) in
let relationships: Sequence(UML::Element) =
 toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
 ->union(toElementFMS
 ->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
 ->union(redefinedAttributes
 ->collect(e | AttributeRedefinedMembership_Mapping.getMapped(e)))
 ->union(generalizations->collect(e | Generalization_Mapping.getMapped(e)))
 ->including(EncapsulatedBlockMetadataMembership_Mapping.getMapped(from)) in
 if from.classifierBehavior.oclIsUndefined() then
 relationships
 else
 relationships
 ->append(BehavioredClassifierFeatureMembership_Mapping.getMapped(from))
endif
```

### **7.8.4.3.5 EncapsulatedBlockMetadataMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

GenericToOwningMembership\_Mapping

#### **Mapping Source**

Class

#### **Mapping Target**

OwningMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
    EncapsulatedBlockMetadata\_Mapping.getMapped (from)

### **7.8.4.3.6 EncapsulatedBlockMetadata\_Mapping**

#### **Description**

The mapping class creates the metadata for the property SysML::Blocks::Block::isEncapsulated.

#### **General Mappings**

GenericToMetadataUsage\_Mapping

#### **Mapping Source**

Class

#### **Mapping Target**

MetadataUsage

#### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{EncapsulatedBlockMetadataFeatureTyping_Mapping.getMapped(from),
EncapsulatedBlockMetadataFeatureMembership_Mapping.getMapped(from)}
```

## **7.8.4.3.7 EncapsulatedBlockMetadataFeatureMembership\_Mapping**

### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

### **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

Class

### **Mapping Target**

FeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]

```
EncapsulatedBlockMetadataReferenceUsage_Mapping.getMapped(from)
```

## **7.8.4.3.8 EncapsulatedBlockMetadataFeatureTyping\_Mapping**

### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

### **General Mappings**

GenericToFeatureTyping\_Mapping

**Mapping Source**

Class

**Mapping Target**

FeatureTyping

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  

```
SYSML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::BlockData')
```

### 7.8.4.3.9 EncapsulatedBlockMetadataReferenceUsage\_Mapping

**Description**

Creates a reference usage.

**General Mappings**

GenericToReferenceUsage\_Mapping

**Mapping Source**

Class

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{EncapsulatedBlockMetadataRedefinition_Mapping.getMapped(from),
EncapsulatedBlockMetadataFeatureValue_Mapping.getMapped(from)}
```

#### 7.8.4.3.10 EncapsulatedBlockMetadataFeatureValue\_Mapping

##### Description

Creates a feature value relationship.

##### General Mappings

GenericToFeatureValue\_Mapping

##### Mapping Source

Class

##### Mapping Target

FeatureValue

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
LiteralBoolean_Factory.create(true)
```

#### 7.8.4.3.11 EncapsulatedBlockMetadataRedefinition\_Mapping

##### Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

##### General Mappings

GenericToRedefinition\_Mapping

##### Mapping Source

Class

## **Mapping Target**

Redefinition

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::BlockData::isEncapsulated')
```

### **7.8.4.3.12 PartProperty\_Mapping**

**SYSML2-432: Part properties with AggregationKind::none or shared are not mapped to PartUsage with isComposite=false**

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

## **Description**

A UML4SysML::Property which is typed by a block is mapped to a SysML::PartUsage. The derived property Property::isComposite is directly mapped to PartUsage::isComposite.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part def SysMLv1Block1 {
 part sysMLv1PartProperty1 : SysMLv1Block2;
 ref part sysMLv1ReferencedPartProperty2 : SysMLv1Block2;
}
part def SysMLv1Block2;
```

## **General Mappings**

PropertyTypedByClassInterface\_Mapping

## **Mapping Source**

Property

## **Mapping Target**

PartUsage

## **Owned Mappings**

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
if src.oclIsKindOf(UML::Property) and not src.oclIsKindOf(UML::Port) then
 let p: UML::Property = src.oclaType(UML::Property) in
 not p.type.oclIsUndefined() and
 Helper.hasStereotypeApplied(p.type, 'SysML::Blocks::Block') and
 (p.association.oclIsUndefined() or p.association.ownedEnd->excludes(p))
else
 false
endif
```

### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

## 7.8.4.3.13 Model Libraries

### 7.8.4.3.13.1 PrimitiveValueTypes

The SysML v1 model library PrimitiveValueTypes contains primitive types that are mapped to the appropriate scalar values in SysML v2.

#### 7.8.4.3.13.1.1 Boolean

The SysML v1 primitive type Boolean is mapped to the SysML v2 ScalarValues::Boolean element.

#### 7.8.4.3.13.1.2 Complex

The SysML v1 primitive type Complex is mapped to the SysML v2 ScalarValues::Complex element.

#### 7.8.4.3.13.1.3 Integer

The SysML v1 primitive type Integer is mapped to the SysML v2 ScalarValues::Integer element.

#### 7.8.4.3.13.1.4 Number

The SysML v1 primitive type Number is abstract. Therefore, no mapping is defined for it.

#### 7.8.4.3.13.1.5 Real

The SysML v1 primitive type Real is mapped to the SysML v2 ScalarValues::Real element.

#### 7.8.4.3.13.1.6 String

The SysML v1 primitive type String is mapped to the SysML v2 ScalarValues::String element.

### 7.8.4.3.13.2 UnitAndQuantityKind

The SysML v1 model library UnitAndQuantityKind contains the blocks Unit and QuantityKind.

#### **7.8.4.3.13.2.1 QuantityKind**

The mapping of the SysML v1 QuantityKind element is not specified yet.

#### **7.8.4.3.13.2.2 Unit**

The mapping of the SysML v1 QuantityKind element is not specified yet.

#### **7.8.4.3.14 ValueType\_Mapping**

**[SYSML2-437: The transformation specification does not explicitly specify how to map a ValueType](#)**

##### **Description**

A SysML::Blocks::ValueType is mapped to a SysML v2 AttributeDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
attribute definition SysMLv1ValueType;
```

##### **General Mappings**

**DataType\_Mapping**

##### **Mapping Source**

**DataType**

##### **Mapping Target**

**AttributeDefinition**

##### **Owned Mappings**

(none)

##### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(from, 'SysML::Blocks::ValueType')
```

##### **Mapping rules**

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

#### **7.8.5 ConstraintBlocks**

This chapter lists all mapping specifications of SysML::ConstraintBlocks model elements.

**[SYSML2-513: Missing text in some main mapping sections](#)**

### 7.8.5.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**Table 28. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax |
|-------------------------------------|--------------------------|
| ConstraintBlock                     | ConstraintDefinition     |

The following table gives an overview of which SysML v2 elements the SysML::ConstraintBlocks elements are transformed with which mapping class. The mapping details are in [7.8.5.2](#).

### 7.8.5.2 Mapping Specifications

#### 7.8.5.2.1 ConstraintBlock\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

A SysML::ConstraintBlocks::ConstraintBlock is mapped to a SysML v2 ConstraintDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
onstraint def SysMLv1ConstraintBlock {
 in attribute a : ScalarValues::Integer;
 in attribute b : ScalarValues::Integer;
 in attribute c : ScalarValues::Integer;

 constraint constraintExpression {
 language "OCL2.0"
 /*
 * c == a + b
 */
 }
}
```

#### General Mappings

Class\_Mapping

##### Mapping Source

Class

##### Mapping Target

ConstraintDefinition

##### Owned Mappings

(none)

## Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::ConstraintBlocks::ConstraintBlock')
```

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `ConstraintDefinition::ownedRelationship () : Relationship [0..*]`

```
let generalizations : Set(UML::Generalization) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Generalization)) in
let toElementFMS : Set(UML::Element) =
 from.ownedElement
 ->select(e | e.oclIsKindOf(UML::Property) or e.oclIsKindOf(UML::Constraint)) in
let toElementOMS: Set(UML::Element) =
 (from.ownedElement - generalizations) - toElementFMS in
toElementOMS->collect(e | ElementOwnership_Mapping.getMapped(e))
->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
->union(generalizations->collect(e | Generalization_Mapping.getMapped(e)))
->including(CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from))
```

### 7.8.5.2.2 ConstraintParameter\_Mapping

**SYSML2-443: Property\_Mapping should map to ItemUsage and the class name is misleading**  
**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

The mapping class maps SysML v1 constraint parameter to SysML v2 attribute usages.

#### General Mappings

PropertyCommon\_Mapping  
NamedElementMain\_Mapping

#### Mapping Source

Property

#### Mapping Target

AttributeUsage

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```

if src.oclIsKindOf(UML::Property) and
Helper.hasStereotypeApplied(src.owner, 'SysML::ConstraintBlocks::ConstraintBlock') then
 let p: UML::Property = src.oclaType(UML::Property) in
 if p.type.oclIsUndefined() then
 false
 else
 true
 endif
else
 false
endif

```

### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

## 7.8.6 Model Elements

This chapter lists all mapping specifications of SysML::ModelElements model elements.

### 7.8.6.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

**Table 29. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax |
|-------------------------------------|--------------------------|
| Conform                             |                          |
| ElementGroup                        | Package                  |
| Expose                              |                          |
| Problem                             | Comment                  |
| Rationale                           | Comment                  |
| Stakeholder                         | ItemDefinition           |
| View                                |                          |
| Viewpoint                           |                          |

The following table gives an overview of which SysML v2 elements the SysML::ModelElements elements are transformed with which mapping class. The mapping details are in [7.8.6.3](#).

The justifications for the elements without mapping are given in [7.8.6.2](#).

### 7.8.6.2 SysML::ModelElements elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566: Section containing tables about elements not mapped should get an introductory text**

**Table 30. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept | Rationale                     |
|------------------|-------------------------------|
| Conform          | Mapping is not specified yet. |
| Expose           | Mapping is not specified yet. |
| View             | Mapping is not specified yet. |

### 7.8.6.3 Mapping Specifications

#### 7.8.6.3.1 ProblemRationaleMetadataFeatureMembership\_Mapping

##### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

##### General Mappings

GenericToFeatureMembership\_Mapping

##### Mapping Source

Comment

##### Mapping Target

FeatureMembership

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [0..1]

```
ProblemRationaleMetadataReferenceUsage_Mapping.getMapped(from)
```

#### 7.8.6.3.2 ProblemRationaleMetadataFeatureTyping\_Mapping

##### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

##### General Mappings

GenericToFeatureTyping\_Mapping

##### Mapping Source

Comment

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
if Helper.hasStereotypeApplied(from, 'SysML::ModelElements::Problem') then
 SYSML2::MetadataDefinition.allInstances()
 ->any(m | m.qualifiedName = 'ModelingMetadata::Issue')
else if Helper.hasStereotypeApplied(from, 'SysML::ModelElements::Rationale') then
 SYSML2::MetadataDefinition.allInstances()
 ->any(m | m.qualifiedName = 'ModelingMetadata::Rationale')
else invalid endif endif
```

## **7.8.6.3.3 ProblemRationaleMetadataReferenceUsage\_Mapping**

### **Description**

Creates a reference usage.

### **General Mappings**

GenericToReferenceUsage\_Mapping

### **Mapping Source**

Comment

### **Mapping Target**

ReferenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]  

```
Set<ProblemRationaleMetadataRedefinition_Mapping.getMapped(from),
ProblemRationaleMetadataFeatureValue_Mapping.getMapped(from) }
```

### 7.8.6.3.4 ProblemRationaleMetadataFeatureValue\_Mapping

#### Description

Creates a feature value relationship.

#### General Mappings

GenericToFeatureValue\_Mapping

#### Mapping Source

Comment

#### Mapping Target

FeatureValue

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  

```
LiteralString_Factory.create(from.body)
```

### 7.8.6.3.5 ProblemRationaleMetadataMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Comment

### **Mapping Target**

OwningMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
ProblemRationaleMetadataUsage_Mapping.getMapped (from)
```

### **7.8.6.3.6 Concern\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### **Description**

The concern comments of a SysML::ModelElements::Stakeholder or a SysML::ModelElements::Viewpoint are mapped to SysML v2 ConcernUsages. The concern comments of the stakeholder are mapped to ConcernUsages which reference the stakeholder item definition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
item def SysMLv1Stakeholder {
 @SysMLv1Library::StakeholderData {isStakeholder = true;}
}
concern concernCommentXMI_ID {
 doc /* concern string */
 stakeholder : SysMLv1Stakeholder;
}
```

### **General Mappings**

Comment\_Mapping

### **Mapping Source**

Comment

### **Mapping Target**

## ConcernUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
(not Helper.hasStereotypeApplied(src, 'SysML::ModelElements::ElementGroup')) and
((UML::Classifier.allInstances()
->select(s |
 Helper.hasStereotypeApplied(s, 'SysML::ModelElements::Stakeholder'))
->collect(c |
 Helper.getTagValue(c, 'SysML::ModelElements::Stakeholder', 'concernList'))
->flatten()
->includes(src)) or
(UML::Classifier.allInstances()
->select(s |
 Helper.hasStereotypeApplied(s, 'SysML::ModelElements::Viewpoint'))
->collect(c |
 Helper.getTagValue(c, 'SysML::ModelElements::Viewpoint', 'concernList'))
->flatten()->includes(src)))
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConcernUsage::ownedRelationship () : Relationship [0..\*]

```
let toStakeholderMS : Set(UML::Classifier) =
 UML::Classifier.allInstances()
->select(s |
 Helper.hasStereotypeApplied(s, 'SysML::ModelElements::Stakeholder'))
->select(s |
 Helper.getTagValue(s, 'SysML::ModelElements::Stakeholder', 'concernList')
->flatten()->includes(from))->asSet() in
toStakeholderMS
->including(
 CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from))
->including(EmptySubjectMembership_Factory.create())
->union(self.oclAsType(Comment_Mapping).ownedRelationship())
```

### 7.8.6.3.7 ConcernDocumentation\_Mapping

#### Description

The mapping class creates the documentation element with the body string of the UML4SysML::Comment model element representing a concern.

#### General Mappings

## GenericToDocumentation\_Mapping

### Mapping Source

Comment

### Mapping Target

Documentation

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Documentation::body () : String [1]

from.body

## 7.8.6.3.8 ConcernOwningMembership\_Mapping

### Description

Creates a owning membership relationship for *ownedMemberElement()*.

### General Mappings

GenericToOwningMembership\_Mapping

### Mapping Source

Comment

### Mapping Target

OwningMembership

### Owned Mappings

(none)

### Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

ConcernDocumentation\_Mapping.getMapped(from)

### 7.8.6.3.9 ConcernStakeholderMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToParameterMembership\_Mapping

#### Mapping Source

Classifier

#### Mapping Target

StakeholderMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- StakeholderMembership::ownedMemberParameter () : Feature [1]

ConcernStakeholderPartUsage\_Mapping.getMapped(from)

### 7.8.6.3.10 ConcernStakeholderPartUsage\_Mapping

#### Description

In SysML v1, the stakeholder element has concerns. In SysML v2, the Concern element has stakeholders. This mapping class creates a PartUsage of the type of the stakeholder for the concern element.

#### General Mappings

GenericToPartUsage\_Mapping

#### Mapping Source

Classifier

**Mapping Target**

PartUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PartUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ConcernStakeholderPartUsageFeatureTyping_Mapping.getMapped(from),
ConcernStakeholderPartUsageOwningMembership_Mapping.getMapped(from)}
```

### **7.8.6.3.11 ConcernStakeholderPartUsageFeatureTyping\_Mapping**

**Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

**General Mappings**

GenericToFeatureTyping\_Mapping

**Mapping Source**

Classifier

**Mapping Target**

FeatureTyping

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

from

### 7.8.6.3.12 ConcernStakeholderPartUsageOwningMembership\_Mapping

#### Description

Creates a owning membership relationship for *ownedMemberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Classifier

#### Mapping Target

OwningMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
ConcernStakeholderPartUsageFeature_Mapping.getMapped(from)
```

### 7.8.6.3.13 ConcernStakeholderPartUsageFeature\_Mapping

#### Description

The mapping class creates a feature element for the concern stakeholder part usage.

#### General Mappings

GenericToFeature\_Mapping

#### Mapping Source

Classifier

#### Mapping Target

Multiplicity

#### Owned Mappings

(none)

#### 7.8.6.3.14 ElementGroup\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A SysML::ModelElements::ElementGroup element is mapped to a SysML v2 Package with membership import relationships representing the grouping.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
package ElementGroupModel {
 part def SysMLv1Block1;
 attribute def SysMLv1ValueType;
 part def SysMLv1Block2 {
 part sysMLv1PartProperty:SysMLv1Block1;
 }
}

package SysMLv1ElementGroup {
 import ElementGroupModel::SysMLv1Block1;
 import ElementGroupModel::SysMLv1ValueType;
 import ElementGroupModel::SysMLv1Block2::sysMLv1PartProperty;

 @SysMLv1Library::ElementGroupData {criterion = "criterion string";}
}
```

##### General Mappings

Comment\_Mapping

##### Mapping Source

Comment

##### Mapping Target

Package

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::ModelElements::ElementGroup')
```

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Package::declaredName () : String [0..1]  
Helper.getValueAsString(from, 'SysML::ModelElements::ElementGroup', 'name')
- Package::ownedRelationship () : Relationship [0..\*]  

```
let elements : Set(KerML::Relationahip) =
 Helper.getValueAsElementColl(from,
 'SysML::ModelElements::ElementGroup', 'member')
 ->collect(e | CommonElementImport_Mapping.getMapped(e)) in
elements->including(ElementGroupMetadaMembership_Mapping.getMapped(from))
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

### 7.8.6.3.15 ElementGroupMetadaMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Comment

#### Mapping Target

OwningMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
ElementGroupMetadataUsage\_Mapping.getMapped(from)

### 7.8.6.3.16 ElementGroupMetadataFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

Comment

### **Mapping Target**

FeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

ElementGroupMetadataReferenceUsage\_Mapping.getMapped (from)

## **7.8.6.3.17 ElementGroupMetadataFeatureTyping\_Mapping**

### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

## **General Mappings**

GenericToFeatureTyping\_Mapping

### **Mapping Source**

Comment

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYSML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::ElementGroupData')
```

### 7.8.6.3.18 ElementGroupMetadataFeatureValue\_Mapping

#### Description

Creates a feature value relationship.

#### General Mappings

GenericToFeatureValue\_Mapping

#### Mapping Source

Comment

#### Mapping Target

FeatureValue

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
let criterion: String = Helper.getTagValueAsString(from, 'SysML::ModelElements::ElementGroup'
LiteralString_Factory.create(criterion)
```

### 7.8.6.3.19 ElementGroupMetadataRedefinition\_Mapping

#### Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

#### General Mappings

GenericToRedefinition\_Mapping

#### Mapping Source

Comment

### **Mapping Target**

Redefinition

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
let m : SYSML2::Membership =
 SYSML2::AttributeUsage.allInstances()
 ->collect(dt | dt.owningRelationship)
 ->select(r | r.oclIsKindOf(SYSML2::Membership))
 ->any(m | m.memberName = 'criterion') in
if (m.oclIsUndefined()) then
 invalid
else
 m.memberElement
endif
```

## **7.8.6.3.20 ElementGroupMetadataReferenceUsage\_Mapping**

### **Description**

Creates a reference usage.

### **General Mappings**

GenericToReferenceUsage\_Mapping

### **Mapping Source**

Comment

### **Mapping Target**

ReferenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ElementGroupMetadataRedefinition_Mapping.getMapped(from),
ElementGroupMetadataFeatureValue_Mapping.getMapped(from)}
```

## 7.8.6.3.21 ElementGroupMetadataUsage\_Mapping

### Description

The mapping class creates the metadata usage element for the SysML::ModelElements::ElementGroup mapping.

### General Mappings

GenericToMetadataUsage\_Mapping

### Mapping Source

Comment

### Mapping Target

MetadataUsage

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ElementGroupMetadataFeatureTyping_Mapping.getMapped(from),
ElementGroupMetadataFeatureMembership_Mapping.getMapped(from)}
```

## 7.8.6.3.22 ProblemRationale\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

### Description

The mapping class combines the mapping of SysML::ModelElements::Problem and SysML::ModelElements::Rationale. The SysML::ModelElements::Problem is mapped to the library element ModelingMetadata::Issue and the SysML::ModelElements::Rationale is mapped to ModelingMetadata::Rationale.

The expected SysML v2 textual syntax of the mapping is as follows.

```
@ModelingMetadata::Issue {text = "This is a problem statement";}
@ModelingMetadata::Rationale {text = "This is a rationale statement";}
```

## General Mappings

Comment\_Mapping

### Mapping Source

Comment

### Mapping Target

Comment

## Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
(not Helper.hasStereotypeApplied(src, 'SysML::ModelElements::ElementGroup')) and
(Helper.hasStereotypeApplied(src, 'SysML::ModelElements::Problem') or
Helper.hasStereotypeApplied(src, 'SysML::ModelElements::Rationale'))
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Comment::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->including(ProblemRationaleMetadataMembership_Mapping.getMapped(from))
```

## 7.8.6.3.23 ProblemRationaleMetadataRedefinition\_Mapping

### Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

## General Mappings

GenericToRedefinition\_Mapping

### **Mapping Source**

Comment

### **Mapping Target**

Redefinition

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
if Helper.hasStereotypeApplied(from, 'SysML::ModelElements::Problem') then
 SYSML2::AttributeUsage.allInstances()
 ->any(m | m.qualifiedName = 'ModelingMetadata::Issue::text')
else if Helper.hasStereotypeApplied(from, 'SysML::ModelElements::Rationale') then
 SYSML2::AttributeUsage.allInstances()
 ->any(m | m.qualifiedName = 'ModelingMetadata::Rationale::text')
else
 invalid
endif
endif
```

## **7.8.6.3.24 ProblemRationaleMetadataUsage\_Mapping**

### **Description**

The mapping class creates the metadata usage element for the SysML::ModelElements::Problem and SysML::ModelElements::Rationale transformation target.

### **General Mappings**

GenericToMetadataUsage\_Mapping

### **Mapping Source**

Comment

### **Mapping Target**

MetadataUsage

### **Owned Mappings**

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ProblemRationaleMetadataFeatureTyping_Mapping.getMapped(from),
ProblemRationaleMetadataFeatureMembership_Mapping.getMapped(from)}
```

### 7.8.6.3.25 Stakeholder\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-178: ClassifierBehaviorFeatureMembership\_Mapping does not exist**

#### Description

A SysML::ModelElements::Stakeholder is mapped to a SysML v2 ItemDefinition with metadata to tag it as a stakeholder. The concern comments of the stakeholder are mapped to ConcernUsages which reference the stakeholder item definition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
item def SysMLv1Stakeholder {@SysMLv1Library::StakeholderData {isStakeholder = true;}}
concern concernCommentXMI_ID {
 doc /* concern string */
 stakeholder : SysMLv1Stakeholder;
}
```

#### General Mappings

Class\_Mapping

#### Mapping Source

Class

#### Mapping Target

ItemDefinition

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::ModelElements::Stakeholder')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ItemDefinition::ownedRelationship () : Relationship [0..\*]

```
let toElementFMS: Set(UML::Element) =
 from.ownedElement
 ->select(e | (e.oclIsKindOf(UML::Property) and
 (e.oclAsType(UML::Property).redefinedProperty->size() = 0)) or
 e.oclIsKindOf(UML::Operation)) in
let redefinedAttributes: Set(UML::Element) =
 from.ownedElement
 ->select(e | from.oclIsKindOf(UML::DataType) and
 (e.oclAsType(UML::Property).redefinedProperty->size() > 0)) in
let generalizations : Set(UML::Generalization) =
 from.ownedElement
 ->select(e | e.oclIsKindOf(UML::Generalization)) in
let constraints : Set(UML::Constraint) =
 UML::Constraint.allInstances()
 ->select(c | c.constrainedElement->includes(from)) in
let toElementOMS: Set(UML::Element) =
 (((from.ownedElement - toElementFMS) - redefinedAttributes) -
 generalizations) in
let relationships: Sequence(KerML::Relationship) =
 toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
 ->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
 ->union(constraints
 ->collect(e | ConstrainedElementFeatureMembership_Mapping.getMapped(e)))
 ->union(redefinedAttributes
 ->collect(e | AttributeRedefinedMembership_Mapping.getMapped(e)))
 ->union(generalizations->collect(e | Generalization_Mapping.getMapped(e)))
 ->including(StakeholderMetadataOwningMembership_Mapping.getMapped(from)) in
 if from.classifierBehavior.oclIsUndefined() then
 relationships
 else
 relationships->append(BehavioredClassifierFeatureMembership_Mapping.getMapped(from))
 endif
```

### 7.8.6.3.26 StakeholderMetadataUsage\_Mapping

#### Description

The mapping class creates the metadata usage element for the SysML::ModelElements::Stakeholder mapping.

#### General Mappings

GenericToMetadataUsage\_Mapping

#### Mapping Source

Classifier

**Mapping Target**

MetadataUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{StakeholderMetadataFeatureTyping_Mapping.getMapped(from),
StakeholderMetadataFeatureMembership_Mapping.getMapped(from)}
```

### 7.8.6.3.27 StakeholderMetadataFeatureMembership\_Mapping

**Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

**General Mappings**

GenericToFeatureMembership\_Mapping

**Mapping Source**

Classifier

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
StakeholderMetadataReferenceUsage_Mapping.getMapped(from)
```

### 7.8.6.3.28 StakeholderMetadataFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

Classifier

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::StakeholderData')
```

### 7.8.6.3.29 StakeholderMetadataOwningMembership

#### Description

Creates a owning membership relationship for *ownedMemberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Classifier

#### Mapping Target

OwningMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `OwningMembership::ownedMemberElement () : Element [1]`  
`StakeholderMetadataUsage_Mapping.getMapped (from)`

## **7.8.6.3.30 StakeholderMetadataReferenceUsage\_Mapping**

### **Description**

Creates a reference usage.

### **General Mappings**

`GenericToReferenceUsage_Mapping`

### **Mapping Source**

Classifier

### **Mapping Target**

ReferenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `ReferenceUsage::ownedRelationship () : Relationship [0..*]`  
`Set{StakeholderMetadataReferenceUsageRedefinition_Mapping.getMapped (from) , StakeholderMetadataReferenceUsageFeatureValue_Mapping.getMapped (from) }`

## **7.8.6.3.31 StakeholderMetadataReferenceUsageFeatureValue\_Mapping**

### **Description**

Creates a feature value relationship.

### **General Mappings**

GenericToFeatureValue\_Mapping

#### **Mapping Source**

Classifier

#### **Mapping Target**

FeatureValue

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
    LiteralBoolean\_Factory.create(true)

### **7.8.6.3.32 StakeholderMetadataReferenceUsageRedefinition\_Mapping**

#### **Description**

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

### **General Mappings**

GenericToRedefinition\_Mapping

#### **Mapping Source**

Classifier

#### **Mapping Target**

Redefinition

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::StakeholderData::isStakeholder')
```

### 7.8.6.3.33 Viewpoint\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-178: ClassifierBehaviorFeatureMembership\_Mapping does not exist**

#### Description

A SysML::ModelElements::Viewpoint is mapped to a SysML v2 ViewDefinition with an owned SysML v2 ViewpointUsage. In SysML v1, the viewpoint combines the purpose and stakeholder concerns as well as presentation information. This is covered by a SysML v2 ViewDefinition with owned SysML v2 ViewpointUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
view def SysMLv1Viewpoint {
 viewpoint sysMLv1Viewpoint {
 frame concern1XmiID1;
 frame concern2XmiID2;
 metadata SysMLv1Library::ViewpointData {
 languages = ("language1", "language2");
 presentations = ("presentation1", "presentation2");
 }
 require constraint {
 doc /* thisIsThePurpose */
 }
 }
 satisfy sysMLv1Viewpoint;
 rendering {
 action : SysMLv1ViewpointMethodBehavior1;
 action : SysMLv1ViewpointMethodBehavior2;
 }
}
action def SysMLv1ViewpointMethodBehavior1;
action def SysMLv1ViewpointMethodBehavior2;

item def SysMLv1Stakeholder {@SysMLv1Library::StakeholderData {isStakeholder = true;}}

concern concern1XmiID1 {
 doc /* Concern1 */
 stakeholder : SysMLv1Stakeholder;
}
concern concern2XmiID2 {
 doc /* Concern2 */
 stakeholder : SysMLv1Stakeholder;
}
```

## General Mappings

## Class\_Mapping

### Mapping Source

Class

### Mapping Target

ViewDefinition

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::ModelElements::Viewpoint')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ViewDefinition::ownedRelationship () : Relationship [0..\*]

```
let toElementFMS: Set(UML::Element) =
 from.ownedElement->select(e | (e.oclIsKindOf(UML::Property) and
 (e.oclAsType(UML::Property).redefinedProperty->size() = 0)) or
 e.oclIsKindOf(UML::Comment)) in
let redefinedAttributes: Set(UML::Element) =
 from.ownedElement->select(e | from.oclIsKindOf(UML::DataType) and
 (e.oclAsType(UML::Property).redefinedProperty->size() > 0)) in
let generalizations : Set(UML::Generalization) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Generalization)) in
let toElementOMS: Set(UML::Element) =
 (((from.ownedElement - toElementFMS) - redefinedAttributes) -
 generalizations) in
let relationships: Sequence(UML::Element) =
 toElementOMS->collect(e | ElementOwningMembership_Mapping.getMapped(e))
 ->union(toElementFMS->collect(e | ElementFeatureMembership_Mapping.getMapped(e)))
 ->union(redefinedAttributes
 ->collect(e | AttributeRedefinedMembership_Mapping.getMapped(e)))
 ->union(generalizations->collect(e | Generalization_Mapping.getMapped(e)))
 ->including(ViewpointViewpointUsageFeatureMembership_Mapping.getMapped(from))
 ->including(ViewpointSatisfyFeatureMembership_Mapping.getMapped(from))
 ->including(ViewpointRenderingFeatureMembership_Mapping.getMapped(from))
 ->including(
 CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from)) in
if from.classifierBehavior.oclIsUndefined() then
 relationships
else
 relationships
 ->append(BehavioredClassifierFeatureMembership_Mapping.getMapped(from))
endif
```

### **7.8.6.3.34 ViewpointConcernReferenceSubsetting\_Mapping**

**SYSML2-200: Description of Subsetting mapping classes is not correct**

#### **Description**

Creates a subsetting relationship.

#### **General Mappings**

GenericToReferenceSubsetting\_Mapping

#### **Mapping Source**

Comment

#### **Mapping Target**

ReferenceSubsetting

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

from

### **7.8.6.3.35 ViewpointConcernUsage\_Mapping**

#### **Description**

The mapping class creates the concern usage element for the SysML::ModelElements::Viewpoint mapping.

#### **General Mappings**

GenericToRequirementUsage\_Mapping

#### **Mapping Source**

Comment

#### **Mapping Target**

ConcernUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConcernUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ViewpointConcernReferenceSubsetting_Mapping.getMapped(from),
EmptySubjectMembership_Factory.create(),
CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from) }
```

### **7.8.6.3.36 ViewpointConstraintUsage\_Mapping**

#### **Description**

The mapping class creates the constraint usage element for the SysML::ModelElements::Viewpoint mapping.

#### **General Mappings**

GenericToConstraintUsage\_Mapping

#### **Mapping Source**

Class

#### **Mapping Target**

ConstraintUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ConstraintUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ViewpointConstraintUsageOwningMembership_Mapping.getMapped(from),
ReturnParameterFeatureMembership_Factory.create() }
```

### **7.8.6.3.37 ViewpointConstraintUsageDocumentation\_Mapping**

#### **Description**

The mapping class creates the documentation element for the SysML::ModelElements::Viewpoint mapping.

### **General Mappings**

GenericToDocumentation\_Mapping

### **Mapping Source**

Class

### **Mapping Target**

Documentation

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Documentation::body () : String [1]

```
Helper.getTagValueAsString(from, 'SysML::ModelElements::Viewpoint', 'purpose')
```

## **7.8.6.3.38 ViewpointConstraintUsageOwningMembership\_Mapping**

### **Description**

Creates a owning membership relationship for *ownedMemberElement()*.

### **General Mappings**

GenericToOwningMembership\_Mapping

### **Mapping Source**

Class

### **Mapping Target**

OwningMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
ViewpointConstraintUsageDocumentation\_Mapping.getMapped (from)

### 7.8.6.3.39 ViewpointFramedConcernMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

Comment

#### Mapping Target

FramedConcernMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FramedConcernMembership::ownedMemberFeature () : Feature [1]  
ViewpointConcernUsage\_Mapping.getMapped (from)

### 7.8.6.3.40 ViewpointLanguagesMetadataFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

Class

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

ViewpointLanguagesMetadataReferenceUsage\_Mapping.getMapped (from)

**7.8.6.3.41 ViewpointLanguagesMetadataFeatureValue\_Mapping**

**Description**

Creates a feature value relationship.

**General Mappings**

GenericToFeatureValue\_Mapping

**Mapping Source**

Class

**Mapping Target**

FeatureValue

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
ViewpointLanguagesMetadataOperatorExpression_Mapping.getMapped(from)
```

### 7.8.6.3.42 ViewpointLanguagesMetadataRedefinition\_Mapping

#### Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

#### General Mappings

GenericToRedefinition\_Mapping

#### Mapping Source

Class

#### Mapping Target

Redefinition

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::ViewpointData::languages')
```

### 7.8.6.3.43 ViewpointLanguagesMetadataReferenceUsage\_Mapping

#### Description

Creates a reference usage.

#### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

Class

#### Mapping Target

ReferenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]  

```
Set{ViewpointLanguagesMetadataRedefinition_Mapping.getMapped(from),
ViewpointLanguagesMetadataFeatureValue_Mapping.getMapped(from)}
```

## **7.8.6.3.44 ViewpointMetadataFeatureTyping\_Mapping**

### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

### **General Mappings**

GenericToFeatureTyping\_Mapping

### **Mapping Source**

Class

### **Mapping Target**

FeatureTyping

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  

```
SYSML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::ViewpointData')
```

## **7.8.6.3.45 ViewpointLanguagesMetadataOperatorExpression\_Mapping**

## Description

The mapping class creates the operator expression for the list of languages of the SysML::ModelElements::Viewpoint mapping.

## General Mappings

GenericToOperatorExpression\_Mapping

### Mapping Source

Class

### Mapping Target

OperatorExpression

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::operator () : String [1]  
    ', '
- OperatorExpression::ownedRelationship () : Relationship [0..\*]  

```
Helper.getTagValueAsStringColl(from, 'SysML::ModelElements::Viewpoint', 'language')
->collect(e | StringParameterMembership_Factory.create(e))
```

## 7.8.6.3.46 ViewpointMetadataOwningMembership\_Mapping

### Description

Creates a owning membership relationship for *ownedMemberElement()*.

## General Mappings

GenericToOwningMembership\_Mapping

### Mapping Source

Class

### Mapping Target

OwningMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `OwningMembership::ownedMemberElement () : Element [1]`  
`ViewpointMetadataUsage_Mapping.getMapped (from)`

## **7.8.6.3.47 ViewpointMetadataUsage\_Mapping**

### **Description**

The mapping class creates the metadata usage element for the SysML::ModelElements::Viewpoint mapping.

### **General Mappings**

`GenericToMetadataUsage_Mapping`

### **Mapping Source**

Class

### **Mapping Target**

MetadataUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `MetadataUsage::ownedRelationship () : Relationship [0..*]`  
`Set{ViewpointMetadataFeatureTyping_Mapping.getMapped (from),`  
`ViewpointLanguagesMetadataFeatureMembership_Mapping.getMapped (from),`  
`ViewpointPresentationsMetadataFeatureMembership_Mapping.getMapped (from) }`

## **7.8.6.3.48 ViewpointPresentationsMetadataFeatureMembership\_Mapping**

## Description

Creates a feature membership relationship for *ownedMemberFeature()*.

## General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

Class

### Mapping Target

FeatureMembership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ViewpointPresentationsMetadataReferenceUsage_Mapping.getMapped(from)
```

## 7.8.6.3.49 ViewpointPresentationsMetadataFeatureValue\_Mapping

### Description

Creates a feature value relationship.

## General Mappings

GenericToFeatureValue\_Mapping

### Mapping Source

Class

### Mapping Target

FeatureValue

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
ViewpointPresentationsMetadataOperatorExpression\_Mapping.getMapped(from)

## 7.8.6.3.50 ViewpointPresentationsMetadataOperatorExpression\_Mapping

### Description

The mapping class creates the operator expression for the list of presentations of the SysML::ModelElements::Viewpoint mapping.

### General Mappings

GenericToOperatorExpression\_Mapping

### Mapping Source

Class

### Mapping Target

OperatorExpression

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OperatorExpression::ownedRelationship () : Relationship [0..\*]  
Helper.getTagValueAsStringColl(from,  
'SysML::ModelElements::Viewpoint', 'presentation')  
->collect(e | StringParameterMembership\_Factory.create(e))
- OperatorExpression::operator () : String [1]  
,

## 7.8.6.3.51 ViewpointPresentationsMetadataRedefinition\_Mapping

### Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

### General Mappings

GenericToRedefinition\_Mapping

### Mapping Source

Class

### Mapping Target

Redefinition

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::ViewpointData::presentations')
```

## 7.8.6.3.52 ViewpointPresentationsMetadataReferenceUsage\_Mapping

### Description

Creates a reference usage.

### General Mappings

GenericToReferenceUsage\_Mapping

### Mapping Source

Class

### Mapping Target

ReferenceUsage

### Owned Mappings

(none)

### Applicable filters

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ViewpointPresentationsMetadataRedefinition_Mapping.getMapped(from),
ViewpointPresentationsMetadataFeatureValue_Mapping.getMapped(from)}
```

## **7.8.6.3.53 ViewpointRenderingFeatureMembership\_Mapping**

### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

### **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

Class

### **Mapping Target**

FeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ViewpointRenderingUsage_Mapping.getMapped(from)
```

## **7.8.6.3.54 ViewpointRenderingUsage\_Mapping**

### **Description**

The mapping class creates the rendering usage element for the SysML::ModelElements::Viewpoint mapping class.

### **General Mappings**

GenericToPartUsage\_Mapping

**Mapping Source**

Class

**Mapping Target**

RenderingUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- RenderingUsage::ownedRelationship () : Relationship [0..\*]

```
from.ownedOperation
->select(o | Helper.hasStereotypeApplied(o, 'Create'))
->collect(e |
 ViewpointRenderingUsageActionUsageFeatureMembership_Mapping.getMapped(e))
```

### 7.8.6.3.55 ViewpointRenderingUsageActionUsage\_Mapping

**Description**

The mapping class creates the action usage element for the rendering usage element for the SysML::ModelElements::Viewpoint mapping class.

**General Mappings**

GenericToActionUsage\_Mapping

**Mapping Source**

Class

**Mapping Target**

ActionUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ActionUsage::ownedRelationship () : Relationship [0..\*]  
Set{ViewpointRenderingUsageActionUsageFeatureTyping\_Mapping.getMapped(from)}

### 7.8.6.3.56 ViewpointRenderingUsageActionUsageFeatureMembership\_Mapping

#### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

#### General Mappings

GenericToFeatureMembership\_Mapping

#### Mapping Source

Class

#### Mapping Target

FeatureMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
ViewpointRenderingUsageActionUsage\_Mapping.getMapped(from)

### 7.8.6.3.57 ViewpointRenderingUsageActionUsageFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

Class

## **Mapping Target**

FeatureTyping

## **Owned Mappings**

(none)

### **7.8.6.3.58 ViewpointRequirementConstraintMembership\_Mapping**

#### **Description**

Creates a membership relationship for *memberElement()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

Class

## **Mapping Target**

RequirementConstraintMembership

## **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- RequirementConstraintMembership::ownedMemberFeature () : Feature [1]

    ViewpointConstraintUsage\_Mapping.getMapped (from)

### **7.8.6.3.59 ViewpointSatisfyFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

Class

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
ViewpointSatisfyRequirementUsage_Mapping.getMapped(from)
```

**7.8.6.3.60 ViewpointSatisfyRequirementUsage\_Mapping****Description**

The mapping class creates the satisfy requirement usage element for the SysML::ModelElements::Viewpoint mapping.

**General Mappings**

GenericToRequirementUsage\_Mapping

**Mapping Source**

Class

**Mapping Target**

SatisfyRequirementUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SatisfyRequirementUsage::ownedRelationship () : Relationship [0..\*]

```
Set{ViewpointSatisfyRequirementUsageReferenceSubsetting_Mapping.getMapped(from),
EmptySubjectMembership_Factory.create(),
ReturnParameterFeatureMembership_Factory.create()}
```

### 7.8.6.3.61 ViewpointSatisfyRequirementUsageReferenceSubsetting\_Mapping

**SYSML2-200:** Description of Subsetting mapping classes is not correct

#### Description

Creates a subsetting relationship.

#### General Mappings

GenericToReferenceSubsetting\_Mapping

#### Mapping Source

Class

#### Mapping Target

ReferenceSubsetting

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

```
ViewpointViewpointUsage_Mapping.getMapped(from)
```

### 7.8.6.3.62 ViewpointViewpointUsage\_Mapping

#### Description

The mapping class creates the embedded viewpoint usage for the SysML::ModelElements::Viewpoint mapping.

#### General Mappings

GenericToUsage\_Mapping

#### Mapping Source

Class

#### Mapping Target

## ViewpointUsage

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ViewpointUsage::ownedRelationship () : Relationship [0..\*]

```
Helper.getTagValueAsElementColl(
 from, 'SysML::ModelElements::Viewpoint', 'concernList')
->collect(e | ViewpointFramedConcernMembership_Mapping.getMapped(e))
->including(ViewpointMetadataOwningMembership_Mapping.getMapped(from))
->including(EmptySubjectMembership_Factory.create())
->including(ViewpointRequirementConstraintMembership_Mapping.getMapped(from))
```

- ViewpointUsage::declaredName () : String [0..1]

```
from.name.substring(1,1).toLowerCase() + from.name.substring(2, from.name.size())
```

## 7.8.6.3.63 ViewpointViewpointUsageFeatureMembership\_Mapping

### Description

Creates a feature membership relationship for *ownedMemberFeature()*.

### General Mappings

GenericToFeatureMembership\_Mapping

### Mapping Source

Class

### Mapping Target

FeatureMembership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
ViewpointViewpointUsage\_Mapping.getMapped (from)

## 7.8.7 PortsAndFlows

This chapter lists all mapping specifications of SysML::PortsAndFlows model elements.

### 7.8.7.1 Overview

**SYSML2-441:** Change the table header of the overview tables in the mapping class specification chapters

**SYSML2-139:** Transformation does not cover SysMLv1::~InterfaceBlock

**Table 31. List of all mappings**

| SysML v1 Abstract Syntax/Stereotype      | SysML v2 Abstract Syntax |
|------------------------------------------|--------------------------|
| AcceptChangeStructuralFeatureEventAction | AcceptActionUsage        |
| AddFlowPropertyValueOnNestedPortAction   |                          |
| ChangeStructuralFeatureEvent             |                          |
| DirectedFeature                          | PerformActionUsage       |
| FlowProperty                             |                          |
| FullPort                                 | PartUsage                |
| InterfaceBlock                           | PortDefinition           |
| InvocationOnNestedPortAction             |                          |
| ItemFlow                                 |                          |
| ProxyPort                                |                          |
| TriggerOnNestedPort                      |                          |
| ~InterfaceBlock                          | PortDefinition           |

The following table gives an overview of which SysML v2 elements the SysML::Ports&Flows elements are transformed with which mapping class. The mapping details are in [7.8.7.3](#).

The justifications for the elements without mapping are given in [7.8.7.2](#).

### 7.8.7.2 SysML::Ports&Flows elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566:** Section containing tables about elements not mapped should get an introductory text

**SYSML2-139:** Transformation does not cover SysMLv1::~InterfaceBlock

**Table 32. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept                       | Rationale                     |
|----------------------------------------|-------------------------------|
| AddFlowPropertyValueOnNestedPortAction | Mapping is not specified yet. |
| ChangeStructuralFeatureEvent           | Mapping is not specified yet. |
| FlowProperty                           | Mapping is not specified yet. |
| InvocationOnNestedPortAction           | Mapping is not specified yet. |
| TriggerOnNestedPort                    | Mapping is not specified yet. |

### 7.8.7.3 Mapping Specifications

**[SYSML2-180: Mapping of UML4SysML::InformationFlow between definition elements is not supported](#)**

#### 7.8.7.3.1 AcceptChangeStructuralFeatureEventAction\_Mapping

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**

##### Description

The SysML::PortsAndFlows::AcceptChangeStructuralFeatureEventAction element is mapped to SysML v2 AcceptActionUsage. The details of the mapping are not defined yet.

##### General Mappings

AcceptEventAction\_Mapping

##### Mapping Source

AcceptEventAction

##### Mapping Target

AcceptActionUsage

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src,
'SysML::Ports&Flows::AcceptChangeStructuralFeatureEventAction')
```

##### Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

#### 7.8.7.3.2 CommonFullPort\_Mapping

## Description

The abstract mapping class is the base class of the mapping classes for the SysML::Ports&Flows::FullPort mappings.

## General Mappings

PropertyCommon\_Mapping

### Mapping Source

Port

### Mapping Target

PartUsage

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PartUsage::ownedRelationship () : Relationship [0..\*]

```
let typings: Set(KerML::FeatureTyping) = if from.type.oclIsUndefined() then
 Set{}
else
 Set{StructuralFeatureToFeatureTyping_Mapping.getMapped(from)}
endif in
let subsettings: Set(KerML::Subsetting) = from.subsettedProperty
 ->collect(p | PropertySubsetting_Mapping.getMapped(from, p))->asSet() in
let defaultValue: Set(KerML::OwningMembership) =
if from.defaultValue.oclIsUndefined() then
 Set{}
else
 Set{DefaultValue_Mapping.getMapped(from)}
endif in
 typings->union(subettings)->union(defaultValue)
->including(MultiplicityMembership_Mapping.getMapped(from))->asSet()
->including(FullPortMetadataOwningMembership_Mapping.getMapped(from))
```

### 7.8.7.3.3 FeatureDirectionKind

### 7.8.7.3.4 FlowDirectionKind

### **7.8.7.3.5 FullPort\_Mapping**

**SYSML2-443: Property\_Mapping should map to ItemUsage and the class name is misleading  
SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### **Description**

A SysML::Ports&Flows::FullPort element is mapped to a part usage in SysML v2 with metadata that marks the part usage as a full port. The metadata is defined in the SysML v1 library for SysML v2.

The mapping class FullPortUntyped\_Mapping does the same for full ports that have no type.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part sysMLv1FullPort : SysMLv1Block {SysMLv1Library::PortData {isFullPort = true;}}
```

#### **General Mappings**

Port\_Mapping  
CommonFullPort\_Mapping

#### **Mapping Source**

Port

#### **Mapping Target**

PartUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
(not src.type.oclIsUndefined()) and
Helper.hasStereotypeApplied(src, 'SysML::Ports&Flows::FullPort')
```

#### **Mapping rules**

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### **7.8.7.3.6 FullPortMetadata\_Mapping**

#### **Description**

Create the metadata usage element to annotate a port with the information that its SysML v1 mapping source element is a SysML v1 full port element.

#### **General Mappings**

GenericToMetadataUsage\_Mapping

**Mapping Source**

Port

**Mapping Target**

MetadataUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]  

```
Set<FullPortMetadataFeatureTyping_Mapping.getMapped(from),
FullPortMetadataFeatureMembership_Mapping.getMapped(from) }
```

### 7.8.7.3.7 FullPortMetadataFeatureMembership\_Mapping

**Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

**General Mappings**

GenericToFeatureMembership\_Mapping

**Mapping Source**

Port

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
FullPortMetadataReferenceUsage\_Mapping.getMapped (from)

### 7.8.7.3.8 FullPortMetadataFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

Port

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  
SYSML2::MetadataDefinition.allInstances()  
->any (m | m.qualifiedName = 'SysMLv1Library::PortData')

### 7.8.7.3.9 FullPortMetadataOwningMembership\_Mapping

#### Description

Creates a owning membership relationship for *ownedMemberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Port

**Mapping Target**

OwningMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
FullPortMetadata_Mapping.getMapped(from)
```

**7.8.7.3.10 FullPortMetadataReferenceUsage\_Mapping****Description**

Creates a reference usage.

**General Mappings**

GenericToReferenceUsage\_Mapping

**Mapping Source**

Port

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{FullPortMetadataReferenceUsageRedefinition_Mapping.getMapped(from),
FullPortMetadataReferenceUsageFeatureValue_Mapping.getMapped(from)}
```

### **7.8.7.3.11 FullPortMetadataReferenceUsageFeatureValue\_Mapping**

#### **Description**

Creates a feature value relationship.

#### **General Mappings**

GenericToFeatureValue\_Mapping

#### **Mapping Source**

Port

#### **Mapping Target**

FeatureValue

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
    LiteralBoolean\_Factory.create(true)

### **7.8.7.3.12 FullPortMetadataReferenceUsageRedefinition\_Mapping**

#### **Description**

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

#### **General Mappings**

GenericToRedefinition\_Mapping

#### **Mapping Source**

Port

#### **Mapping Target**

Redefinition

#### **Owned Mappings**

(none)

## Applicable filters

(none)

## Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::PortData::isFullPort')
```

### 7.8.7.3.13 FullPortUntyped\_Mapping

#### **SYSML2-7: Pin\_Mapping::filter: property src should be from**

##### Description

A SysML::Ports&Flows::FullPort element is mapped to a part usage in SysML v2 with metadata that marks the part usage as a full port. The metadata is defined in the SysML v1 library for SysML v2.

The mapping class FullPort\_Mapping does the same for full ports with a type.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
part sysMLv1FullPort {SysMLv1Library::PortData {isFullPort = true;}}
```

## General Mappings

PortUntyped\_Mapping

CommonFullPort\_Mapping

## Mapping Source

Port

## Mapping Target

PartUsage

## Owned Mappings

(none)

## Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
src.type.oclIsUndefined() and
Helper.hasStereotypeApplied(src, 'SysML::Ports&Flows::FullPort')
```

## Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.8.7.3.14 InterfaceBlock\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

A SysML::Ports&Flows::InterfaceBlock element is mapped to a SysML v2 PortDefinition.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
port def SysMLv1InterfaceBlock;
```

#### General Mappings

Block\_Mapping

#### Mapping Source

Class

#### Mapping Target

PortDefinition

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Ports&Flows::InterfaceBlock')
```

## Mapping rules

The mapping class only has inherited rules. See the mapping classes in the general mapping section for details.

### 7.8.7.3.15 InterfaceBlockConjugated\_Mapping

**SYSML2-569: Rename ~InterfaceBlock\_Mapping**

**SYSML2-139: Transformation does not cover SysMLv1::~InterfaceBlock**

#### Description

A SysML::Ports&Flows::~InterfaceBlock element is mapped to a SysML v2 PortDefinition. The SysML v1 constraints ensure that the port definition is compatible with the appropriate port definition, which is the target of the mapping of the original interface block. Instead of the special tilde symbol, the port definition name gets a "c"

symbol as a prefix. The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
port def cSysMLv1InterfaceBlock;
```

### General Mappings

InterfaceBlock\_Mapping

#### Mapping Source

Class

#### Mapping Target

PortDefinition

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Ports&Flows::~InterfaceBlock')
```

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PortDefinition::declaredName () : String [0..1]  

```
'c' + from.name.substring(2,from.name.size())
```

### 7.8.7.3.16 OperationDirectedFeature\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

#### Description

The mapping class sets the direction of the perform action usage if the SysML v1 mapping source operation has the stereotype SysML::Ports&Flows::DirectedFeature applied.

### General Mappings

Operation\_Mapping

#### Mapping Source

Operation

#### Mapping Target

PerformActionUsage

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Ports&Flows::DirectedFeature')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- PerformActionUsage::direction () : FeatureDirectionKind [0..1]

```
 Helper.getKerMLFeatureDirectionKind(
 Helper.getTagValueAsElement(
 from, 'SysML::Ports&Flows::DirectedFeature', 'featureDirection'
))
```

## 7.8.8 Requirements

This chapter lists all mapping specifications of SysML::Requirements model elements.

### 7.8.8.1 Overview

**SYSML2-441: Change the table header of the overview tables in the mapping class specification chapters**

Table 33. List of all mappings

| SysML v1 Abstract Syntax/Stereotype | SysML v2 Abstract Syntax          |
|-------------------------------------|-----------------------------------|
| Copy                                |                                   |
| DeriveReqt                          | ConnectionUsage                   |
| Refine                              | Dependency                        |
| Requirement                         | RequirementUsage                  |
| Satisfy                             | SatisfyRequirementUsage           |
| TestCase                            | VerificationCaseDefinition        |
| Trace                               | Dependency                        |
| Verify                              | RequirementVerificationMembership |

The following table gives an overview of which SysML v2 elements the SysML::Requirements elements are transformed with which mapping class. The mapping details are in [7.8.8.3](#).

The justifications for the elements without mapping are given in [7.8.8.2](#).

### 7.8.8.2 SysML::Requirements elements not mapped

In this section, missing transformation rules of SysML v1 elements to SysML v2 are justified for each individual element in the following table.

**SYSML2-566: Section containing tables about elements not mapped should get an introductory text**

**Table 34. List of SysML v1 elements not mapped of this section**

| SysML v1 Concept | Rationale                                         |
|------------------|---------------------------------------------------|
| Copy             | The copy relationship is not covered by SysML v2. |

### 7.8.8.3 Mapping Specifications

#### 7.8.8.3.1 DeriveReqt\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A SysML::Requirements::DeriveReqt relationship is mapped to a SysML v2 DerivationConnections::Derivation model library element.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
requirement <'id1'> SysMLv1Requirement {
 doc /*
 * requirement text
 */
}
requirement <'id2'> SysMLv1RequirementDerived {
 doc /*
 * requirement text
 */
}
connection : DerivationConnections::Derivation
 connect SysMLv1RequirementDerived to SysMLv1Requirement;
```

#### General Mappings

Abstraction\_Mapping  
GenericToConnectionUsage\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

ConnectionUsage

#### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Requirements::DeriveReqt')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `ConnectionUsage::ownedRelationship () : Relationship [0..*]`

```
Set{DeriveReqtFeatureTyping_Mapping.getMapped(from),
DeriveReqtSourceEndFeatureMembership_Mapping.getMapped(from),
DeriveReqtTargetEndFeatureMembership_Mapping.getMapped(from)}
->union(self.oclAsType(ElementMain_Mapping).ownedRelationship())
```

## 7.8.8.3.2 DeriveReqtFeatureTyping\_Mapping

### Description

Creates a feature typing relationship owned by the element `typedFeature()`.

### General Mappings

`GenericToFeatureTyping_Mapping`

### Mapping Source

Dependency

### Mapping Target

FeatureTyping

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- `FeatureTyping::type () : Type [1]`

```
SYSML2::ConnectionDefinition.allInstances()
->any(m | m.qualifiedName = 'DerivationConnections::Derivation')
```

### **7.8.8.3.3 DeriveReqtSourceEndFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToEndFeatureMembership\_Mapping

#### **Mapping Source**

Dependency

#### **Mapping Target**

EndFeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]  

```
DeriveReqtSourceFeature_Mapping.getMapped(from)
```

### **7.8.8.3.4 DeriveReqtSourceFeature\_Mapping**

#### **Description**

The mapping class creates the source feature of the ConnectionUsage relationship for the mapping of the SysML v1 deriveReqt relationship.

#### **General Mappings**

GenericToFeature\_Mapping

#### **Mapping Source**

Dependency

#### **Mapping Target**

Feature

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{DeriveReqtSourceFeatureReferenceSubsetting_Mapping.getMapped(from) }
```

### **7.8.8.3.5 DeriveReqtSourceFeatureReferenceSubsetting\_Mapping**

**SYSML2-200: Description of Subsetting mapping classes is not correct**

#### **Description**

Creates a subsetting relationship.

#### **General Mappings**

GenericToReferenceSubsetting\_Mapping

#### **Mapping Source**

Dependency

#### **Mapping Target**

ReferenceSubsetting

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

```
from.client->any(c | true)
```

### **7.8.8.3.6 DeriveReqtTargetEndFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

### **General Mappings**

GenericToEndFeatureMembership\_Mapping

### **Mapping Source**

Dependency

### **Mapping Target**

EndFeatureMembership

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- EndFeatureMembership::ownedMemberFeature () : Feature [1]

DeriveReqtTargetFeature\_Mapping.getMapped(from)

## **7.8.8.3.7 DeriveReqtTargetFeature\_Mapping**

### **Description**

The mapping class creates the target feature of the ConnectionUsage relationship for the mapping of the SysML v1 deriveReqt relationship.

### **General Mappings**

GenericToFeature\_Mapping

### **Mapping Source**

Dependency

### **Mapping Target**

Feature

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{DeriveReqtTargetFeatureReferenceSubsetting_Mapping.getMapped(from)}
```

### 7.8.8.3.8 DeriveReqtTargetFeatureReferenceSubsetting\_Mapping

**SYSML2-200: Description of Subsetting mapping classes is not correct**

#### Description

Creates a subsetting relationship.

#### General Mappings

GenericToReferenceSubsetting\_Mapping

#### Mapping Source

Dependency

#### Mapping Target

ReferenceSubsetting

#### Owned Mappings

(none)

#### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

```
from.supplier->any(c | true)
```

### 7.8.8.3.9 Refine\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from**

**SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

#### Description

A SysML::Requirements::Refine relationship is mapped to a SysML v2 Dependency relationship annotated with a metadata usage tagging it as a former SysML v1 refine relationship.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
requirement <'id1'> SysMLv1Requirement {
 doc /*
 * requirement text
 */
}
use case def SysMLv1UseCase;

dependency from SysMLv1UseCase to SysMLv1Requirement {
 @SysMLv1Library::RefineData {isRefine = true;}
}
```

## General Mappings

Abstraction\_Mapping

### Mapping Source

Abstraction

### Mapping Target

Dependency

### Owned Mappings

(none)

### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Requirements::Refine')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Dependency::ownedRelationship () : Relationship [0..\*]  

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
 ->including(RefineAnnotation_Mapping.getMapped(from))
```

## 7.8.8.3.10 RefineAnnotation\_Mapping

### Description

The mapping class creates the annotation relationship for the SysML::Requirements::Refine mapping.

## **General Mappings**

GenericToAnnotation\_Mapping

### **Mapping Source**

Abstraction

### **Mapping Target**

Annotation

## **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Annotation::annotatingElement () : AnnotatingElement [1]

RefineMetadataUsage\_Mapping.getMapped(from)

## **7.8.8.3.11 RefineMetadataFeatureMembership\_Mapping**

### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

## **General Mappings**

GenericToFeatureMembership\_Mapping

### **Mapping Source**

Abstraction

### **Mapping Target**

FeatureMembership

## **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
RefineMetadataReferenceUsage\_Mapping.getMapped(from)

### 7.8.8.3.12 RefineMetadataReferenceUsage\_Mapping

#### Description

Creates a reference usage.

#### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

ReferenceUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]  
Set{RefineMetadataReferenceUsageRedefinition\_Mapping.getMapped(from),  
RefineMetadataReferenceUsageFeatureValue\_Mapping.getMapped(from)}

### 7.8.8.3.13 RefineMetadataReferenceUsageFeatureValue\_Mapping

#### Description

Creates a feature value relationship.

#### General Mappings

GenericToFeatureValue\_Mapping

#### Mapping Source

Abstraction

**Mapping Target**

FeatureValue

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]  
    LiteralBoolean\_Factory.create(true)

**7.8.8.3.14 RefineMetadataReferenceUsageRedefinition\_Mapping****Description**

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

**General Mappings**

GenericToRedefinition\_Mapping

**Mapping Source**

Abstraction

**Mapping Target**

Redefinition

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  
    SYSML2::AttributeUsage.allInstances()  
    ->any(m | m.qualifiedName = 'SysMLv1Library::RefineData::isRefine')

### **7.8.8.3.15 RefineMetadataUsage\_Mapping**

#### **Description**

Create the metadata usage element to annotate a dependency relationship with the information that its SysML v1 mapping source element is a SysML v1 refine relationship.

#### **General Mappings**

GenericToMetadataUsage\_Mapping

#### **Mapping Source**

Abstraction

#### **Mapping Target**

MetadataUsage

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{RefineMetadataUsageFeatureTyping_Mapping.getMapped(from),
RefineMetadataFeatureMembership_Mapping.getMapped(from)}
```

### **7.8.8.3.16 RefineMetadataUsageFeatureTyping\_Mapping**

#### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

#### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

Abstraction

#### **Mapping Target**

FeatureTyping

#### **Owned Mappings**

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYSML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::RefineData')
```

### 7.8.8.3.17 Requirement\_Mapping

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**  
**[SYSML2-280: ElementMain\\_Mapping::ownedRelationship is wrong](#)**

#### Description

A SysML::Requirement is mapped to a SysML v2 RequirementUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
requirement <'id1'> SysMLv1Requirement {
 doc /*
 * requirement text
 */

 requirement <'id2'> SysMLv1NestedRequirement {
 doc /*
 * requirement text
 */
 }
}
```

#### General Mappings

NamedElementMain\_Mapping  
GenericToRequirementUsage\_Mapping

#### Mapping Source

Class

#### Mapping Target

RequirementUsage

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.isRequirement(src)
```

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- RequirementUsage::ownedRelationship () : Relationship [0..\*]  

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->including(CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from))
->including(RequirementDocumentationMembership_Mapping.getMapped(from))
->including(RequirementSubjectMembership_Mapping.getMapped(from))
```
- RequirementUsage::reqId () : String [1]  

```
let stereotype: UML::Stereotype = Helper.getRequirementStereotype(from) in
Helper.getTagValueAsString(from, stereotype.qualifiedName, 'id')
```

### 7.8.8.3.18 RequirementDocumentation\_Mapping

#### Description

The mapping class creates a Comment contained in a Requirement which contains the SysML::Requirements::AbstractRequirement::text property.

#### General Mappings

GenericToDocumentation\_Mapping

#### Mapping Source

Class

#### Mapping Target

Documentation

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Documentation::body () : String [1]

```
let stereotype: UML::Stereotype = Helper.getRequirementStereotype(from) in
Helper.getTagValueAsString(from, stereotype.qualifiedName, 'text')
```

### 7.8.8.3.19 RequirementDocumentationMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Class

#### Mapping Target

OwningMembership

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]

```
RequirementDocumentation_Mapping.getMapped(from)
```

### 7.8.8.3.20 RequirementSubject\_Mapping

#### Description

The mapping class creates the subject reference usage element of the requirement. It is not used since the concept does not exist SysML v1.

#### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

Class

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]

KerML::FeatureDirectionKind::'\_in'

**7.8.8.3.21 RequirementSubjectMembership\_Mapping**

**Description**

The subject is not used, because it is not a SysML v1 concept, but must be created for a SysML v2 requirement.

**General Mappings**

GenericToParameterMembership\_Mapping

**Mapping Source**

Class

**Mapping Target**

SubjectMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SubjectMembership::ownedMemberParameter () : Feature [0..1]

```
RequirementSubject_Mapping.getMapped(from)
```

### 7.8.8.3.22 Satisfy\_Mapping

**[SYSML2-7: Pin\\_Mapping::filter: property src should be from](#)**  
**[SYSML2-280: ElementMain\\_Mapping::ownedRelationship is wrong](#)**

#### Description

A SysML::Requirements::Satisfy relationship is mapped to a SysML v2 SatisfyRequirementUsage.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
// satisfy relationship from a block
part def SysMLv1Block {
 part sysMLv1PartProperty;
}
requirement <'ReqId1'> SysMLv1Requirement { doc /* requirement text */ }

ref :SysMLv1Block = all SysMLv1Block {
 satisfy requirement SysMLv1Requirement by self;
}

// satisfy relationship from a part property
satisfy SysMLv1Requirement by sysMLv1BlockUsage.sysMLv1PartProperty {
 sysMLv1BlockUsage : SysMLv1Block;
}
```

#### General Mappings

GenericToOccurrenceUsage\_Mapping  
Abstraction\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

SatisfyRequirementUsage

#### Owned Mappings

(none)

#### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
let satisfy: UML::Abstraction = src.oclAsType(UML::Abstraction) in
 if satisfy.oclIsUndefined() then
 false
 else
```

```

 Helper.hasStereotypeApplied(satisfy, 'SysML::Requirements::Satisfy')
endif

```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SatisfyRequirementUsage::ownedRelationship () : Relationship [0..\*]

```

let relationships : Set(KerML::Relationship) =
 self.oclaSType(ElementMain_Mapping).ownedRelationship()
->including(SatisfyFeatureTyping_Mapping.getMapped(from))
->including(SatisfySubjectSubjectMembership_Mapping.getMapped(from))
->including(CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from)) in
if from.client->any(c | true).oclIsKindOf(UML::Property) then
 relationships
->including(SatisfyReferenceUsageFeatureMembership_Mapping.getMapped(from))
else
 relationships
endif

```

### 7.8.8.3.23 SatisfyReferenceUsage\_Mapping

#### Description

Creates a reference usage.

#### General Mappings

GenericToReferenceUsage\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

ReferenceUsage

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set{SatisfyReferenceUsageFeatureTyping_Mapping.getMapped(from)}
```

- ReferenceUsage::declaredName () : String [0..1]

```

from.client
->any(c | true).owner.name.substring(1,1).toLowerCase()
+ from.client
->any(c | true).owner.name.
substring(2,from.client->any(c | true).owner.name.size())
+ 'SatisfyClientUsage'
```

### **7.8.8.3.24 SatisfyReferenceUsageFeatureMembership\_Mapping**

#### **Description**

Creates a feature membership relationship for *ownedMemberFeature()*.

#### **General Mappings**

GenericToFeatureMembership\_Mapping

#### **Mapping Source**

Abstraction

#### **Mapping Target**

FeatureMembership

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]

```
SatisfyReferenceUsage_Mapping.getMapped(from)
```

### **7.8.8.3.25 SatisfySubjectReferenceUsage\_Mapping**

#### **Description**

Creates a reference usage.

#### **General Mappings**

GenericToReferenceUsage\_Mapping

#### **Mapping Source**

Abstraction

### **Mapping Target**

ReferenceUsage

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::direction () : FeatureDirectionKind [0..1]  
    KerML::FeatureDirectionKind::\_'in'
- ReferenceUsage::ownedRelationship () : Relationship [0..\*]  
    Set{SatisfySubjectReferenceUsageFeatureValue\_Mapping.getMapped(from) }

## **7.8.8.3.26 SatisfySubjectReferenceUsageValue\_Mapping**

### **Description**

The mapping class create the feature reference expression for the subject of the SatisfyRequirementUsage element.

### **General Mappings**

GenericToFeatureReferenceExpression\_Mapping

### **Mapping Source**

Abstraction

### **Mapping Target**

FeatureReferenceExpression

### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureReferenceExpression::ownedRelationship () : Relationship [0..\*]

```
Set{SatisfySubjectReferenceUsageValueOwningMembership_Mapping.getMapped(from),
ReturnParameterFeatureMembership_Factory.create()}
```

### 7.8.8.3.27 SatisfySubjectReferenceUsageValueFeature\_Mapping

#### Description

The mapping class creates the feature element for the feature reference expression of the subject of the SatisfyRequirementUsage element.

#### General Mappings

GenericToFeature\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

Feature

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Feature::ownedRelationship () : Relationship [0..\*]

```
Set{SatisfySubjectReferenceUsageFeatureChaining_Mapping.getMapped(from),
SatisfySubjectReferenceUsageValueFeatureChainingProperty_Mapping.getMapped(from)}
```

### 7.8.8.3.28 SatisfySubjectReferenceUsageFeatureChaining\_Mapping

#### Description

The mapping class creates the feature chaining element from SysML v2 SatisfyRequirementUsage's reference usage element.

#### General Mappings

GenericToFeatureChaining\_Mapping

**Mapping Source**

Abstraction

**Mapping Target**

FeatureChaining

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]  
SatisfyReferenceUsage\_Mapping.getMapped(from)

**7.8.8.3.29 SatisfySubjectReferenceUsageValueFeatureChainingProperty\_Mapping****Description**

The mapping class creates the feature chaining element from the source element of the SysML v1 satisfy relationship.

**General Mappings**

GenericToFeatureChaining\_Mapping

**Mapping Source**

Abstraction

**Mapping Target**

FeatureChaining

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureChaining::chainingFeature () : Feature [1]

```
from.client->any(c | true)
```

### 7.8.8.3.30 SatisfySubjectReferenceUsageFeatureValue\_Mapping

#### Description

Creates a feature value relationship.

#### General Mappings

GenericToFeatureValue\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

FeatureValue

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
SatisfySubjectReferenceUsageValue_Mapping.getMapped(from)
```

### 7.8.8.3.31 SatisfySubjectReferenceUsageValueOwningMembership\_Mapping

#### Description

Creates a owning membership relationship for *ownedMemberElement()*.

#### General Mappings

GenericToOwningMembership\_Mapping

#### Mapping Source

Abstraction

**Mapping Target**

OwningMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- OwningMembership::ownedMemberElement () : Element [1]  
SatisfySubjectReferenceUsageValueFeature\_Mapping.getMapped(from)

### **7.8.8.3.32 SatisfySubjectSubjectMembership\_Mapping**

**Description**

Creates a membership relationship for *memberElement()*.

**General Mappings**

GenericToSubjectMembership\_Mapping

**Mapping Source**

Abstraction

**Mapping Target**

SubjectMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- SubjectMembership::ownedMemberParameter () : Feature [1]  
SatisfySubjectReferenceUsage\_Mapping.getMapped(from)

### **7.8.8.3.33 SatisfyFeatureTyping\_Mapping**

#### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

#### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

Abstraction

#### **Mapping Target**

FeatureTyping

#### **Owned Mappings**

(none)

#### **Applicable filters**

(none)

#### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  

```
from.supplier->any(s | true)
```

### **7.8.8.3.34 SatisfyReferenceUsageFeatureTyping\_Mapping**

#### **Description**

Creates a feature typing relationship owned by the element *typedFeature()*.

#### **General Mappings**

GenericToFeatureTyping\_Mapping

#### **Mapping Source**

Abstraction

#### **Mapping Target**

FeatureTyping

#### **Owned Mappings**

(none)

### **Applicable filters**

(none)

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]  
from.client->any(c | true).owner

### **7.8.8.3.35 TestCaseActivity\_Mapping**

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-240: TestCaseActivity\_Mapping uses non-existing mapping classes**

#### **Description**

A SysML::Requirements::TestCase applied to an activity is mapped to a SysML v2 VerificationCaseDefinition element.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
verification def SysMLv1ActivityTestCase {
 return verdict : VerificationCases::VerdictKind;
}
```

#### **General Mappings**

ActivityAsDefinition\_Mapping

#### **Mapping Source**

Activity

#### **Mapping Target**

VerificationCaseDefinition

#### **Owned Mappings**

(none)

### **Applicable filters**

This mapping applies only if the following (OCL) condition implemented by the operation *filter(src : Element) : Boolean* is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Requirements::TestCase')
```

### **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- VerificationCaseDefinition::ownedRelationship () : Relationship [0..\*]

```
let relationships : Set(KerML::Relationship) =
 Helper.activityOwnedRelationship(from) in
let verdictParameter : Set(UML::Parameter) =
 from.ownedElement->select(e | e.oclIsKindOf(UML::Parameter) and
 (e.oclaType(UML::Parameter).type.name = 'VerdictKind')) in
let parameters : Set(UML::Paramter) =
 ((from.ownedElement->select(e | e.oclIsKindOf(UML::Parameter))) -
 verdictParameter) in
let verifyRelationships : Set(UML::Abstraction) =
 from.clientDependency
 ->select(v |
 Helper.hasStereotypeApplied(v, 'SysML::Requirements::Verify')) in
relationships
->union(parameters->collect(p | ParameterMembership_Mapping.getMapped(p)))
->union(verdictParameter
 ->collect(vp |
 TestCaseActivityReturnParameterMembership_Mapping.getMapped(vp)))
->including(EmptySubjectMembership_Factory.create())
->including(EmptyObjectiveMembership_Factory.create())
->union(verifyRelationships->collect(v | Verify_Mapping.getMapped(v)))
```

### 7.8.8.3.36 TestCaseActivityReturnParameterMembership\_Mapping

#### Description

Creates a membership relationship for *memberElement()*.

#### General Mappings

ParameterMembership\_Mapping

#### Mapping Source

Parameter

#### Mapping Target

ReturnParameterMembership

#### Owned Mappings

(none)

### 7.8.8.3.37 TestCaseVerifyObjectiveMembership\_Mapping

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ownedMemberFeature () : Feature [1]

```
TestCaseVerifyObjectiveRequirementUsage_Mapping.getMapped(from)
```

### 7.8.8.3.38 TestCaseVerifyObjectiveRequirementUsage\_Mapping

#### Description

The mapping class creates the objective requirements usage of the SysML v2 test case.

#### General Mappings

No general mappings.

#### Mapping Source

Abstraction

#### Mapping Target

No target element.

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ownedRelationship () : Relationship [0..\*]

```
Set{Verify_Mapping.getMapped(from)}
```

### 7.8.8.3.39 TestCaseVerifyRequirementUsageReferenceSubsetting\_Mapping

#### **SYSML2-200: Description of Subsetting mapping classes is not correct**

#### Description

Creates a subsetting relationship.

#### General Mappings

GenericToSubsetting\_Mapping

#### Mapping Source

Abstraction

**Mapping Target**

ReferenceSubsetting

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceSubsetting::referencedFeature () : Feature [1]

```
from.supplier->get(0)
```

**7.8.8.3.40 TestCaseVerifyRequirementUsage\_Mapping**

**SYSML2-459: Resolution of approved issue SYSML2-241 is not considered by merged issue SYSML2-240**

**Description**

The mapping class creates the requirements usage of the SysML v2 test case for the verify relationship.

**General Mappings**

GenericToUsage\_Mapping

**Mapping Source**

Abstraction

**Mapping Target**

RequirementUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- RequirementUsage::ownedRelationship () : Relationship [0..\*]

```
Set{TestCaseVerifyRequirementUsageReferenceSubsetting_Mapping.getMapped(from),
EmptySubjectMembership_Factory.create(),
CommonReturnParameterReferenceUsageMembership_Mapping.getMapped(from)}
```

#### 7.8.8.3.41 Trace\_Mapping

**SYSML2-7: Pin\_Mapping::filter: property src should be from  
SYSML2-280: ElementMain\_Mapping::ownedRelationship is wrong**

##### Description

A SysML::Requirements::Trace relationship is mapped to a SysML v2 Dependency relationship annotated with a metadata usage tagging it as a former SysML v1 trace relationship.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
requirement <'id1'> SysMLv1Requirement1 {
 doc /*
 * requirement text
 */
}
requirement <'id2'> SysMLv1Requirement2 {
 doc /*
 * requirement text
 */
}
dependency from SysMLv1Requirement1 to SysMLv1Requirement2 {
 @SysMLv1Library::TraceData {isTrace = true;}
}
```

#### General Mappings

Abstraction\_Mapping

##### Mapping Source

Abstraction

##### Mapping Target

Dependency

##### Owned Mappings

(none)

##### Applicable filters

This mapping applies only if the following (OCL) condition implemented by the operation `filter(src : Element) : Boolean` is verified:

```
Helper.hasStereotypeApplied(src, 'SysML::Requirements::Trace')
```

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Dependency::ownedRelationship () : Relationship [0..\*]

```
self.oclAsType(ElementMain_Mapping).ownedRelationship()
->including(TraceAnnotation_Mapping.getMapped(from))
```

## 7.8.8.3.42 TraceAnnotation\_Mapping

### Description

The mapping class creates the annotation relationship for the SysML::Requirements::Trace mapping.

### General Mappings

GenericToAnnotation\_Mapping

### Mapping Source

Abstraction

### Mapping Target

Annotation

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Annotation::annotatingElement () : AnnotatingElement [1]

```
TraceMetadataUsage_Mapping.getMapped(from)
```

## 7.8.8.3.43 TraceMetadataFeatureMembership\_Mapping

### Description

Creates a feature membership relationship for `ownedMemberFeature()`.

### General Mappings

GenericToFeatureMembership\_Mapping

**Mapping Source**

Abstraction

**Mapping Target**

FeatureMembership

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureMembership::ownedMemberFeature () : Feature [1]  
    TraceMetadataReferenceUsage\_Mapping.getMapped(from)

### 7.8.8.3.44 TraceMetadataReferenceUsage\_Mapping

**Description**

Creates a reference usage.

**General Mappings**

GenericToReferenceUsage\_Mapping

**Mapping Source**

Abstraction

**Mapping Target**

ReferenceUsage

**Owned Mappings**

(none)

**Applicable filters**

(none)

**Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- ReferenceUsage::ownedRelationship () : Relationship [0..\*]

```
Set<TraceMetadataReferenceUsageRedefinition_Mapping.getMapped(from),
TraceMetadataReferenceUsageFeatureValue_Mapping.getMapped(from) }
```

#### 7.8.8.3.45 TraceMetadataReferenceUsageFeatureValue\_Mapping

##### Description

Creates a feature value relationship.

##### General Mappings

GenericToFeatureValue\_Mapping

##### Mapping Source

Abstraction

##### Mapping Target

FeatureValue

##### Owned Mappings

(none)

##### Applicable filters

(none)

##### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureValue::value () : Expression [1]

```
LiteralBoolean_Factory.create(true)
```

#### 7.8.8.3.46 TraceMetadataReferenceUsageRedefinition\_Mapping

##### Description

Creates a redefinition relationship for the *redefiningFeature()* and the *redefinedFeature()*.

##### General Mappings

GenericToRedefinition\_Mapping

##### Mapping Source

Abstraction

## **Mapping Target**

Redefinition

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- Redefinition::redefinedFeature () : Feature [1]  

```
SYSML2::AttributeUsage.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::TraceData::isTrace')
```

## **7.8.8.3.47 TraceMetadataUsage\_Mapping**

### **Description**

Create the metadata usage element to annotate a dependency relationship with the information that its SysML v1 mapping source element is a SysML v1 trace relationship.

## **General Mappings**

GenericToMetadataUsage\_Mapping

## **Mapping Source**

Abstraction

## **Mapping Target**

MetadataUsage

## **Owned Mappings**

(none)

## **Applicable filters**

(none)

## **Mapping rules**

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- MetadataUsage::ownedRelationship () : Relationship [0..\*]

```
Set{TraceMetadataUsageFeatureTyping_Mapping.getMapped(from),
TraceMetadataFeatureMembership_Mapping.getMapped(from)}
```

### 7.8.8.3.48 TraceMetadataUsageFeatureTyping\_Mapping

#### Description

Creates a feature typing relationship owned by the element *typedFeature()*.

#### General Mappings

GenericToFeatureTyping\_Mapping

#### Mapping Source

Abstraction

#### Mapping Target

FeatureTyping

#### Owned Mappings

(none)

#### Applicable filters

(none)

#### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- FeatureTyping::type () : Type [1]

```
SYSML2::MetadataDefinition.allInstances()
->any(m | m.qualifiedName = 'SysMLv1Library::TraceData')
```

### 7.8.8.3.49 Verify\_Mapping

#### Description

A SysML::Requirements::Verify relationship is mapped to a SysML v2 RequirementVerificationMembership relationship.

The following shows an example of what the textual SysML v2 syntax of the result of the transformation may look like.

```
requirement <'id1'> SysMLv1Requirement {
 doc /*
 * requirement text
 */
}
verification def SysMLv1TestCase {
 objective objective_SysMLv1TestCase {
 verify SysMLv1Requirement;
```

```
 }
 return verdict : VerificationCases::VerdictKind;
 }
```

## General Mappings

GenericToRelationship\_Mapping

### Mapping Source

Abstraction

### Mapping Target

RequirementVerificationMembership

### Owned Mappings

(none)

### Applicable filters

(none)

### Mapping rules

In addition to the inherited rules, the following lists the mapping class specific mapping rules for the target element properties.

- RequirementVerificationMembership::ownedRelatedElement () : Element [0..\*]  
Set{TestCaseVerifyRequirementUsage\_Mapping.getMapped(from)}

## 7.8.8.3.50 Model Libraries

### 7.8.8.3.50.1 Verdicts

#### 7.8.8.3.50.1.1 VerdictKind

The enumeration VerdictKind is mapped to the SysML v2 VerificationCases::VerdictKind model library element.