Contract number: ITEA2 – 10039

Safe Automotive soFtware architEcture (SAFE)

ITEA Roadmap application domains:

Major: Services, Systems & Software Creation

Minor: Society

ITEA Roadmap technology categories:

Major: Systems Engineering & Software Engineering

Minor 1: Engineering Process Support

WP3 Deliverable D3.5.c:

Final release of System, SW, HW reference meta-model definition

Due date of deliverable: 28/02/13

Actual submission date: 28/02/13

Start date of the project: 01/07/2011 Duration: 42 months

Project coordinator name: Stefan Voget

Organization name of lead contractor for this deliverable: ITEMIS

Contributors: Aquintos, BMW, Conti-F, Conti-G, Dassault Systèmes, fortiss, FZI, Itemis, LaBRi, OFFIS, PS, TTTech, VEEM.

Revision chart and history log

Version	Date	Reason
0.1	2013-05-21	Initialization of document
0.2	2013-05-23	Conti-F review, change in section 2 and 5
1.0	2014-02-28	Conti-G, Finalization

1 Table of contents

2 Introduction

This document describes the language specification of SAFE.

The meta-model is based on the integration of the results of the conceptual work of all work tasks 3.x. It defines a domain specific language for safety modeling and safety analysis on M2 level. The meta-model provides extensions to EAST-ADL and AUTOSAR which are linked via references.

This document describes the meta-modeling rules, technical principles and the language specification based on an Enterprise Architect file.

The meta-model

The language specification is the core of the methods of the SAFE project. Its main objective is to enhance existing techniques to be able to reach the ISO26262 process in the context of model based development of E/E-architectures in vehicles or sub-systems of vehicles.

The target is to use model based techniques to represent the required entities capable to support safety case evaluation and documentation of vehicle architectures (and/or the subsystems). Results of safety assessments are recorded and documented including hazard identification and scenario goal description, in relation to safety requirements, their decomposition in the functional and technical architecture.

It delivers an open automotive meta-model for description of all design artefacts supporting the engineering process. It is based on the experience from the automotive domain and other domains and on existing techniques such as EAST-ADL for functional abstraction, AUTOSAR for software component, and a hardware description. The hardware part is improved to support electrical distribution systems and new hardware component descriptions either from the EAST-ADL hardware design extension and AUTOSAR ECU resource template with inspiration/alignment to the IP_XACT standard. This meta-model covers specific modelling techniques to support analysis methods, to compare and manage architecture variants during the architecture design phase and as final product variant.

For the safety evaluation, the following objectives are considered:

- Failure error modelling and propagation to be able to perform safety and cut-set analysis thanks to model based technique, aligned with extending analysis methods such as FTA, FMEA with improved quantified value based on system execution context or new methods
- Hardware and software COTS evaluation methods for safety test conformity and integration in safety systems
- Clarification of needs via explicit elicitation of safety requirements and especially the decision tracing the safety measures
- Specification of criteria and methods for architecture safety evaluation and comparison to support design decisions regarding functional safety
- Generation of safety case documentation with respect to safety activities and related model based work-product

3 Modeling Rules

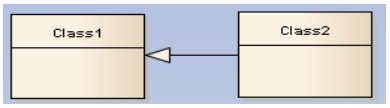
This chapter collects some basic principles used in the definition of the SAFE meta-model.

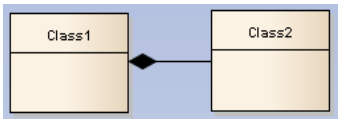
To define a homogeneous meta-model and enable automatic code generation, a set of rules have been defined.

Use of Enterprise Architect

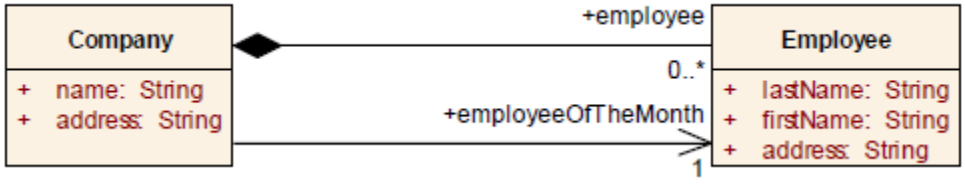
- **Rule 0:** Work on a replica
  - Rationale: Enables the merge function in EA, required to export a unique MM
- **Rule 1:** Provide documentation on
    - Diagrams
    - Packages
    - Classes
    - Attributes
    - Associations.
  - Rationale: Export from EA to deliverables.
    - It helps partners to understand your intentions
    - It can be used as javadoc in generated code
    - It can be included in generated documention

Class relations

- **Rule 2:** Only 3 class relation types are allowed:
  - Generalization 

  - Composition (i.e. contain) 

  - Direct Association



- **Rule 2.1:** For Compose and Association relations, multiplicity shall be explicitly defined
  - Rationale: otherwise generator take assumption.
- **Rule 2.2:** For Compose and Association have to be named at the navigable end, not the connector itself.

  - Rationale: otherwise relation have no mean for code generation
- **Rule 2.3:** Don't use Compose or Association relation with data-type or enumeration
  - Rationale: data-type and enumeration aren't class. (i.e. issue in code generation)
- **Rule 2.4:** always start creating the composition relation from the class to be aggregated
  - Rationale: Ecore generation process assumes that the source always aggregates the target.
- **Rule 2.5:** All associations starting at one class have to have different names at the navigable end.
  - Rationale: in Ecore the „features" have to be unique

Naming convention

- **Rule 3:** Only use java compliant names.
  - Rationale: Names used in the model will be used in generated code
  - Java compliant names (see http://java.sun.com/)
  - E.g.: no reserved keyword
  - Advice 1: Similar concept, different meta-model
    - Rationale: simplify understanding
    - Similar name. Ex: ARPackage <-> SAFEPackage
- **Rule 3.1:** Spaces in entries of enumerations are not allowed

Basic Data types

- **Rule 4:** Only use basic datatypes from the CommonStructure package
  - Rationale: what if a float is not a float ?
- **Rule 4.1 :** These types have the stereotype <<primitive>> or <<enumeration>>
- **Rule 4.2:** All attributes shall have a type
- **Rule 4.3:** Default value shall have the same type

Main structural principle of meta-model

SAFE uses SAFEElement as a root of the meta-model

- **Rule 5.1:** All classes inherit directly or indirectly from SAFEElement

SAFE uses SafetyExtension for structuring the meta-model along abstraction levels (the abstraction levels are taken from EAST-ADL and AUTOSAR).

- **Rule 5.2:** All classes are directly or indirectly contained in one of the subclasses of SafetyExtension

References

Elements from foreign meta-models (e.g. Chromosome, EAST-ADL, AUTOSAR, …) can be referenced by using proxy elements. These elements are located in the package CommonStructure / References

- **Rule 6.1:** reference elements are used by containment relationship only
  - Rationale: if one links them via an association, the referenced element can not be created in a tool.
- **Rule 6.2:** the reference element has the same name as the name in the original meta-model.
- **Rule 6.3:** The meta-model is chosen by the package in which the element is located in

Abstract Classes

- Rule 7: Abstract classes are indicated to be abstract by setting the flag “Abstract” in the properties of the element. The flag is at the tab “Details”.

Note: sometimes the stereotype <<abstract>> is used to make the status more visible. But this setting is for diagram illustration only and does not have a meta-model meaning. Especially, the tool platform generator uses the flag above.

4 Information Model Structure

The SAFE meta-model is structured in the following packages.

CommonStructure	Technical package defining basic structures
	• AUTOSARInstanceRefs
	  ○ Specific structure for instance refs to enable a referencing of AUTOSAR InstanceRefs from the SAFE meta-model.
	• DataTypes
	  ○ Collects all used primitive data types
	• FormulaExpression
	  ○ Enables the definition of formula language
	• References
	  ○ Enables the link and usage of external meta-models (e.g. EAST-ADL, AUTOSAR, CHROMOSOME, …)
	• SafetyExtensions
	  ○ Basic structuring principle based on abstraction levels of SAFE meta-model
	• TopLevel
	  ○ 2nd structuring principle used in SAFE meta-model, based on hierarchies

Configuration	Link to variant management
ErrorModel	Basic component failure description as well as result of safety analysis
Hardware	Safety extensions on hardware level.
Hazards	Definition of hazard, risk, event, controllability
Requirements	Provides links to a requirements perspective and extends safety elements enabling the requirements traceability necessary to fulfill a safety process.
Software	Safety extensions on software level
System	Safety extensions on system level

Model Documentation

Model Detail

This document provides a complete overview of all element details. For simpler and more focused reports, simply copy this initial template and turn off the sections not required.

Package Model

Package Notes:

Package ClassModel

Package Notes:

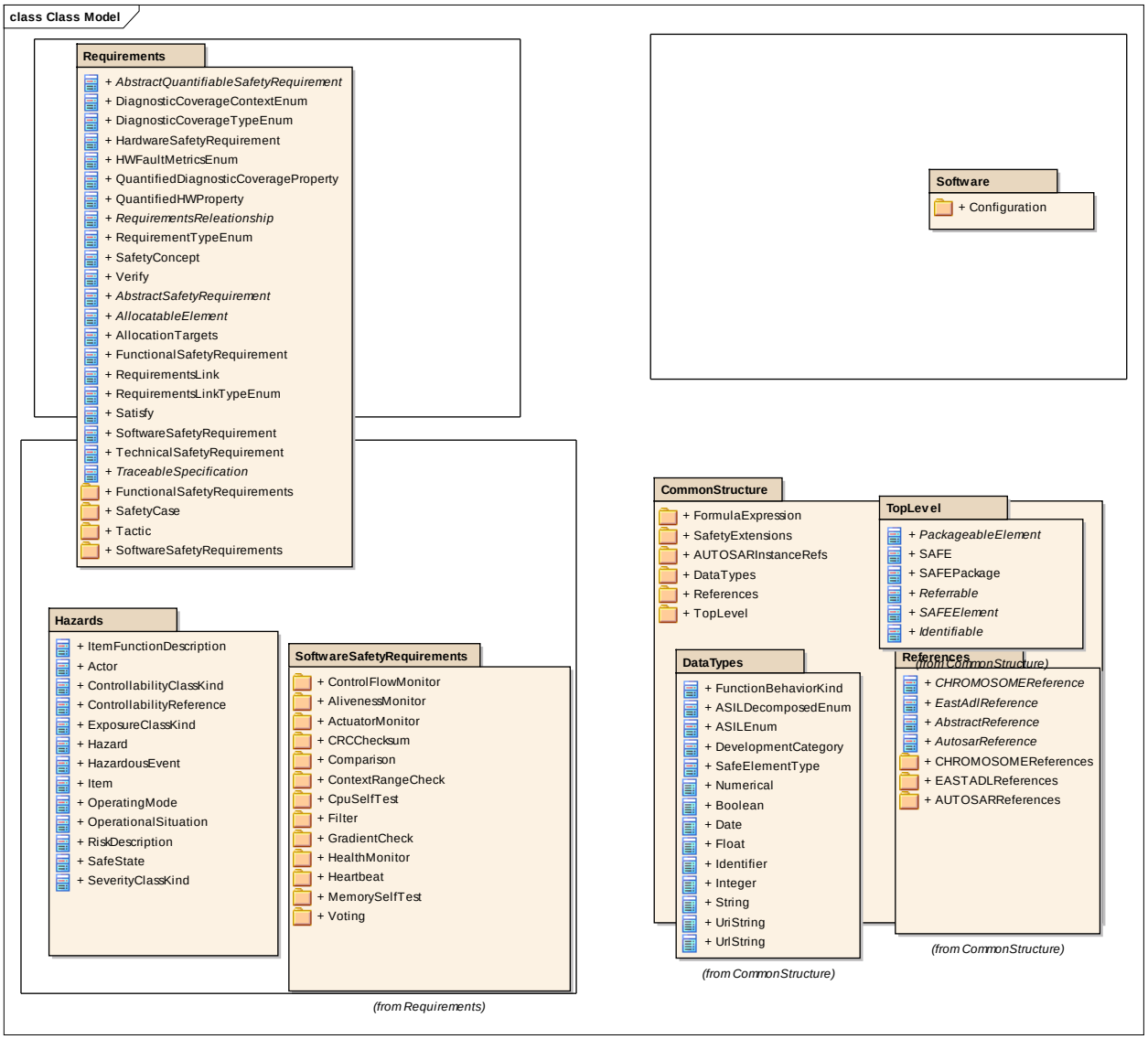


Figure 1: **Class Model** - *(Class diagram)*

Diagram Notes:

Package CommonStructure

Package Notes:

Package AUTOSARInstanceRefs

Package Notes:

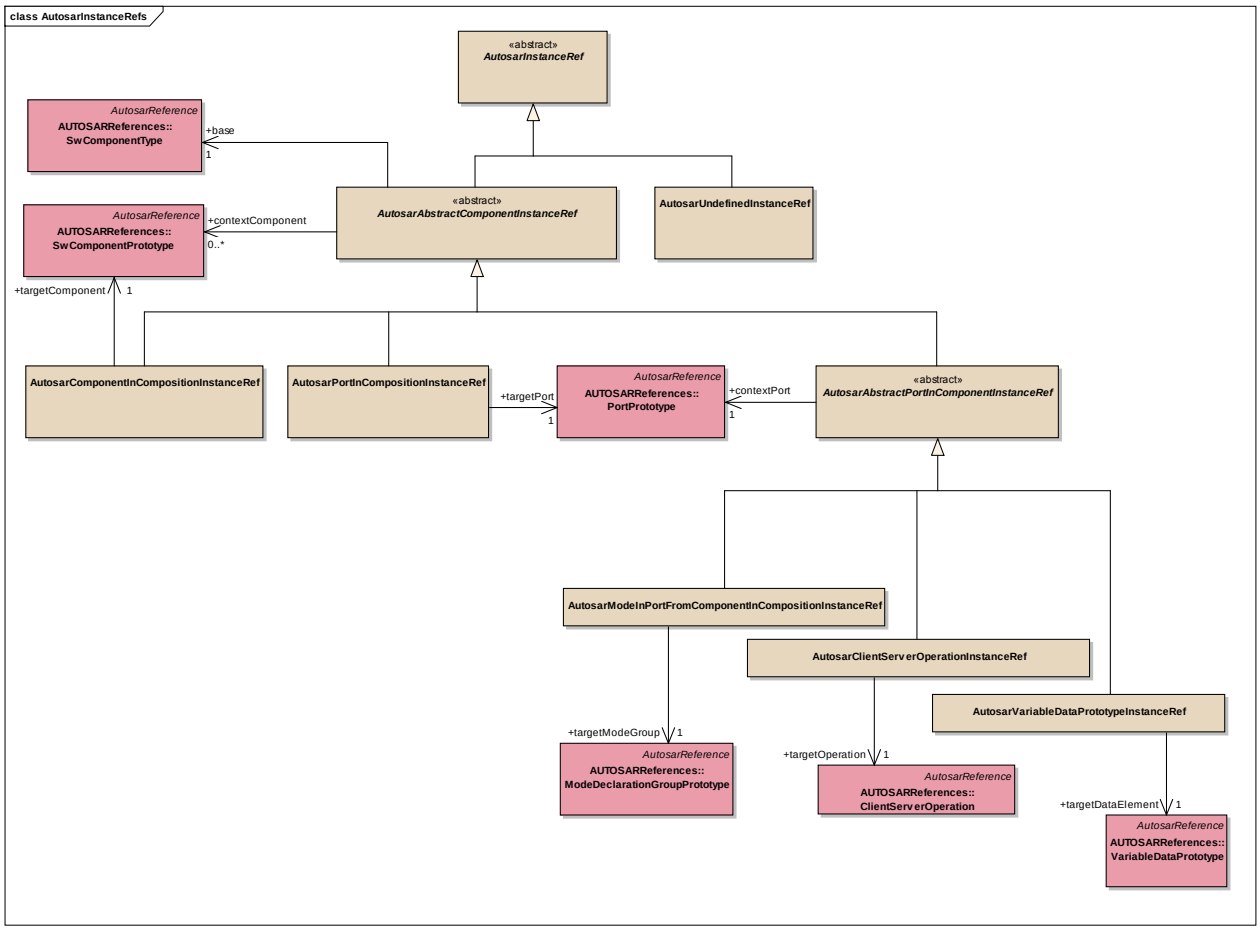


Figure 2: **AutosarInstanceRefs** - *(Class diagram)*

Diagram Notes:

Abstract AutosarAbstractComponentInstanceRef

Element Base Classes: **AutosarInstanceRef**

Element Notes:

Connections

Connector	Source	Target

Connector	Source	Target
Generalization Source -> Destination	AutosarAbstractComponentInstanceRef	AutosarInstanceRef
Generalization Source -> Destination	AutosarPortInCompositionInstanceRef	AutosarAbstractComponentInstanceRef
Generalization Source -> Destination	AutosarComponentInCompositionInstanceRef	AutosarAbstractComponentInstanceRef
Generalization Source -> Destination	AutosarAbstractPortInComponentInstanceRef	AutosarAbstractComponentInstanceRef
Association Source -> Destination	AutosarAbstractComponentInstanceRef	SwComponentType
Association Source -> Destination	AutosarAbstractComponentInstanceRef	SwComponentPrototype

Abstract AutosarAbstractPortInComponentInstanceRef

Element Base Classes: **AutosarAbstractComponentInstanceRef**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarAbstractPortInComponentInstanceRef
Generalization Source -> Destination	AutosarModeInPortFromComponentInCompositionInstanceRef	AutosarAbstractPortInComponentInstanceRef

Connector	Source	Target
Association Source -> Destination	AutosarAbstractPortInComponentInstanceRef	PortPrototype
Generalization Source -> Destination	AutosarClientServerOperationInstanceRef	AutosarAbstractPortInComponentInstanceRef
Generalization Source -> Destination	AutosarAbstractPortInComponentInstanceRef	AutosarAbstractComponentInstanceRef

Class AutosarClientServerOperationInstanceRef

Element Base Classes: **AutosarAbstractPortInComponentInstanceRef**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	AutosarClientServerOperationInstanceRef	ClientServerOperation
Aggregation Source -> Destination	AutosarClientServerOperationInstanceRef	MFPOperation
Generalization Source -> Destination	AutosarClientServerOperationInstanceRef	AutosarAbstractPortInComponentInstanceRef

Class AutosarComponentInCompositionInstanceRef

Element Base Classes: **AutosarAbstractComponentInstanceRef**

Element Notes:

A reference to an AUTOSAR software component within a software component composition instance which finds itself in another model / file.

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	EMPSwComponent
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	ControlFlowMonitor
Generalization Source -> Destination	AutosarComponentInCompositionInstanceRef	AutosarAbstractComponentInstanceRef
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	AutosarCheckPoint
Association Source -> Destination	AutosarComponentInCompositionInstanceRef	SwComponentPrototype
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	AutosarActuatorMonitor

Abstract AutosarInstanceRef

Element Base Classes:

Element Notes:

A reference to an instance of an AUTOSAR element which finds itself in another model / file.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarAbstractComponentInstanceRef	AutosarInstanceRef

Connector	Source	Target
Generalization Source -> Destination	AutosarUndefinedInstanceRef	AutosarInstanceRef

Class AutosarModeInPortFromComponentInCompositionInstanceRef

Element Base Classes: **AutosarAbstractPortInComponentInstanceRef**

Element Notes:

A reference to an AUTOSAR mode for a specific port of a specific software component within a software component composition defined in another model / file.

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarModeInPortFromComponentInCompositionInstanceRef	AutosarContext
Generalization Source -> Destination	AutosarModeInPortFromComponentInCompositionInstanceRef	AutosarAbstractPortInComponentInstanceRef
Association Source -> Destination	AutosarModeInPortFromComponentInCompositionInstanceRef	ModeDeclarationGroupPrototype

Class AutosarPortInCompositionInstanceRef

Element Base Classes: **AutosarAbstractComponentInstanceRef**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarPortInCompositionInstanceRef	OperationBlock
Generalization Source -> Destination	AutosarPortInCompositionInstanceRef	AutosarAbstractComponentInstanceRef
Association Source -> Destination	AutosarPortInCompositionInstanceRef	PortPrototype

Class AutosarUndefinedInstanceRef

Element Base Classes: **AutosarInstanceRef**

Element Notes:

Class for representing AUTOSAR references which are not understood by the SAFE meta model.

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarUndefinedInstanceRef	AUTOSARSequenceConstraint
Generalization Source -> Destination	AutosarUndefinedInstanceRef	AutosarInstanceRef

Class AutosarVariableDataPrototypeInstanceRef

Element Base Classes: **AutosarAbstractPortInComponentInstanceRef**

Element Notes:

A reference to an AUTOSAR variable data prototype of a specific software component within a software component composition which is defined in another model / file.

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarEvaluationFunction
Generalization Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarAbstractPortInComponentInstanceRef
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	MFPVariable
Association Source -> Destination	AutosarVariableDataPrototypeInstanceRef	VariableDataPrototype
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarContextRangeCheck
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarComparisonParam
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarGradientCheck
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	ComponentPrototypeCRC

Package DataTypes

Package Notes:

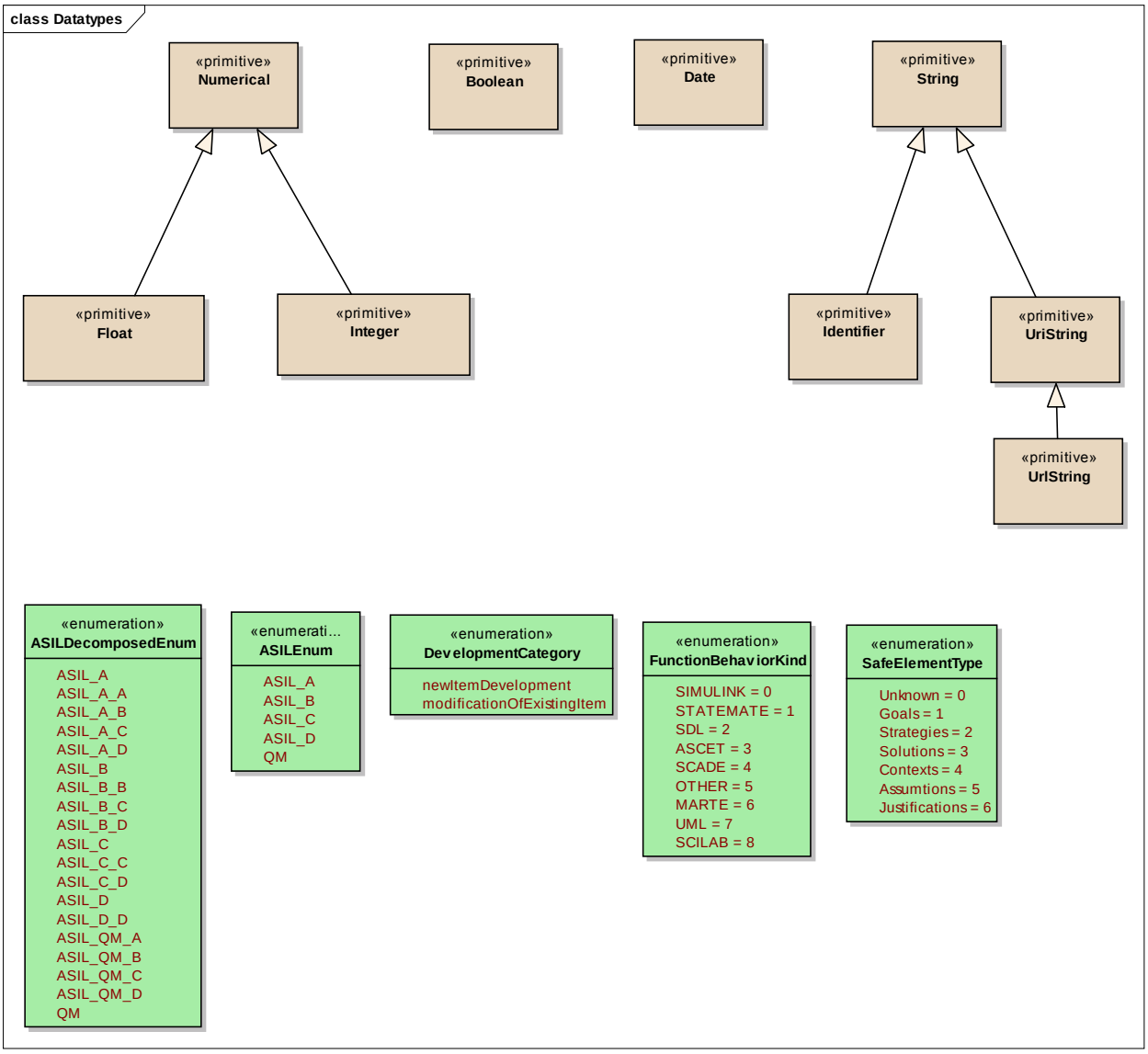


Figure 3: **Datatypes** - *(Class diagram)*

Diagram Notes:

Enumeration ASILDecomposedEnum

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
ASIL_A		
ASIL_A_A		
ASIL_A_B		
ASIL_A_C		
ASIL_A_D		
ASIL_B		
ASIL_B_B		
ASIL_B_C		
ASIL_B_D		
ASIL_C		
ASIL_C_C		
ASIL_C_D		
ASIL_D		
ASIL_D_D		
ASIL_QM_A		
ASIL_QM_B		

Attribute	Notes	Default
ASIL_QM_C		
ASIL_QM_D		
QM		

Enumeration ASILEnum

Element Base Classes:

Element Notes:

An ASIL shall be determined by using the parameters

- severity
- probability of exposure
- controllability

ISO 26262 Reference: Part 3 Chapter 7.4.4

Attributes

Attribute	Notes	Default
ASIL_A		
ASIL_B		
ASIL_C		
ASIL_D		
QM		

Enumeration DevelopmentCategory

Element Base Classes:

Element Notes:

This element is an enumeration for the development kind of an item.

Values:

- new

- modification

ISO 262626 Reference: Part 3 Chapter 6.4.1

Attributes

Attribute	Notes	Default
newItemDevelopment		
modificationOfExistingItem		

Enumeration FunctionBehaviorKind

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
SIMULINK		0
STATEMATE		1

Attribute	Notes	Default
SDL		2
ASCET		3
SCADE		4
OTHER		5
MARTE		6
UML		7
SCILAB		8

Enumeration SafeElementType

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
Unknown		0
Goals		1
Strategies		2

Attribute	Notes	Default
Solutions		3
Contexts		4
Assumtions		5
Justifications		6

DataType Boolean

Element Base Classes:

Element Notes:

DataType Date

Element Base Classes:

Element Notes:

DataType Float

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	Float	Numerical

DataType Identifier

Element Base Classes:

Element Notes:

An Identifier is a string with a number of constraints on its appearance, satisfying the requirements typical programming languages define for their Identifiers.

It needs to start with a letter, may consist of letters, digits and underscore. It must not have two consecutive underscores (to support subsequent name mangling based on "__").

Connections

Connector	Source	Target
Generalization Source -> Destination	Identifier	String

DataType Integer

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	Integer	Numerical

DataType Numerical

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	Float	Numerical
Generalization Source -> Destination	Integer	Numerical

DataType String

Element Base Classes:

Element Notes:

Any string.

This primitive type is redefined here to solve an issue with entreprise architect ecore generation.

Connections

Connector	Source	Target
Generalization Source -> Destination	UriString	String
Generalization Source -> Destination	Identifier	String

DataType UriString

Element Base Classes:

Element Notes:

A Uniform Resource Identifier (URI), is a compact string of characters used to identify or name a resource.

Connections

Connector	Source	Target
Generalization Source -> Destination	UrlString	UriString
Generalization Source -> Destination	UriString	String

DataType UrlString

Element Base Classes:

Element Notes:

A Uniform Resource Location

Connections

Connector	Source	Target
Generalization Source -> Destination	UrlString	UriString

Package FormulaExpression

Package Notes:

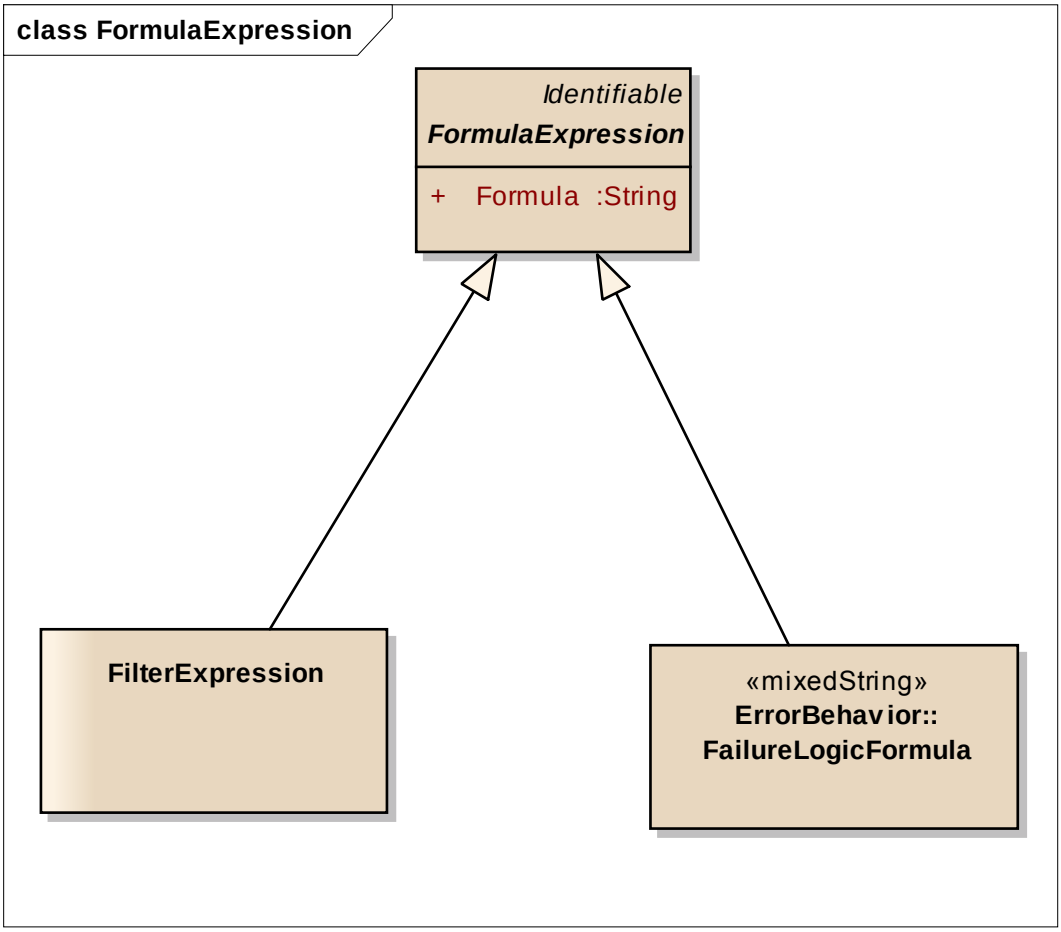


Figure 4: **FormulaExpression** - *(Class diagram)*

Diagram Notes:

Class FilterExpression

Element Base Classes: **FormulaExpression**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FilterExpression	FormulaExpression

Connector	Source	Target
Aggregation Source -> Destination	FilterExpression	Filter

Class FormulaExpression

Element Base Classes: **Identifiable**

Element Notes:

This class represents the syntax of the formula language. The class is modeled as an abstract class in order to be specialized into particular use cases. For each use case the referable objects might be specified in the specialization.

The stereotype "mixedString" of this class and the specialized sub-classes indicates the usage of mixed content model with intermixed text. The grammar of the mixed content model is described in the description field of the dedicated class.

The term "mixed content" is used as defined in XML (http://www.w3schools.com/schema/schema_complex_mixed.asp).

Connections

Connector	Source	Target
Generalization Source -> Destination	FormulaExpression	Identifiable
Generalization Source -> Destination	FailureLogicFormula	FormulaExpression
Generalization Source -> Destination	FilterExpression	FormulaExpression

Attributes

Attribute	Notes	Default
Formula String		

Package References

Package Notes:

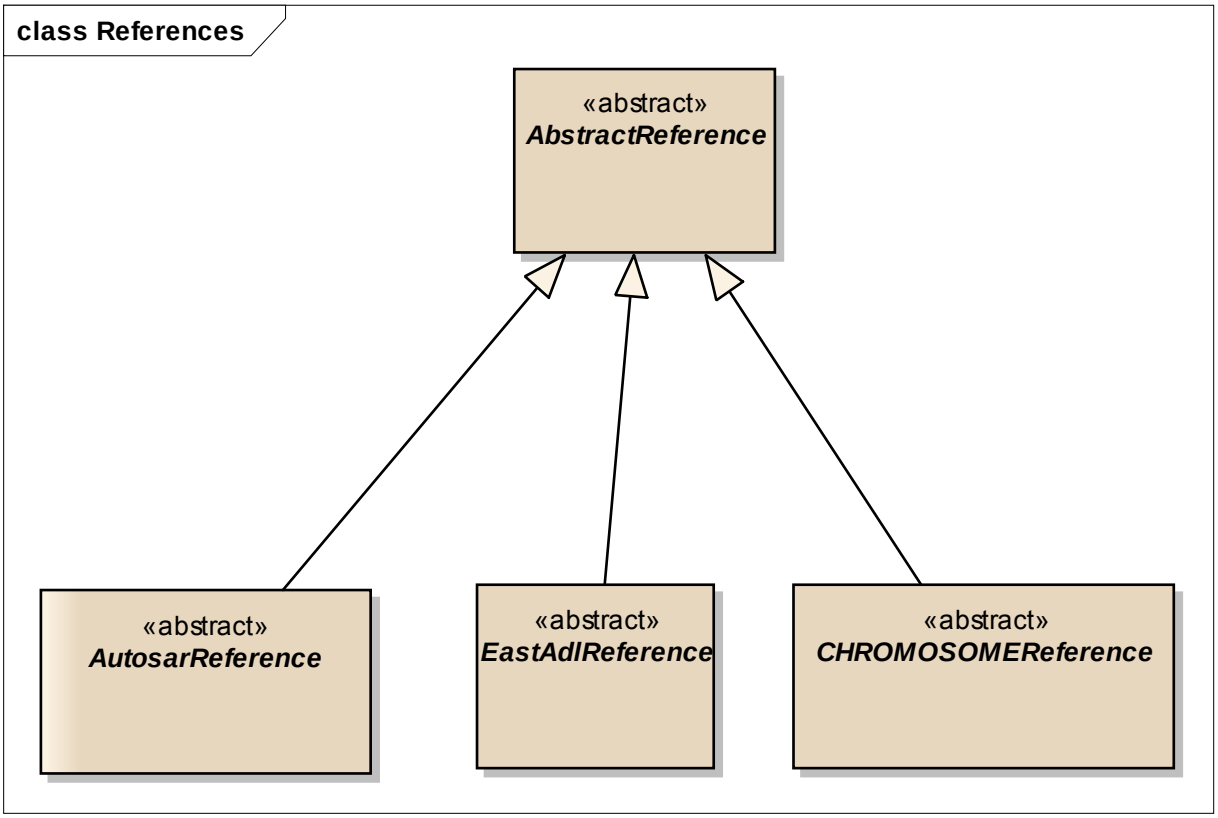


Figure 5: **References** - *(Class diagram)*

Diagram Notes:

Abstract AbstractReference

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	CHROMOSOMEReference	AbstractReference

Connector	Source	Target
Generalization Source -> Destination	EastAdlReference	AbstractReference
Generalization Source -> Destination	AutosarReference	AbstractReference

Abstract AutosarReference

Element Base Classes: **AbstractReference**

Element Notes:

The base reference element which allows references to AUTOSAR elements defined outside SAFE models.

Connections

Connector	Source	Target
Generalization Source -> Destination	VariableDataPrototype	AutosarReference
Generalization Source -> Destination	HwElementPrototype	AutosarReference
Generalization Source -> Destination	SwComponentType	AutosarReference
Generalization Source -> Destination	SwComponentPrototype	AutosarReference
Generalization Source -> Destination	ModeDeclaration	AutosarReference
Generalization Source -> Destination	HwElementType	AutosarReference
Generalization Source -> Destination	ModeDeclarationGroupPrototype	AutosarReference
Generalization Source -> Destination	PortPrototype	AutosarReference
Generalization Source -> Destination	ExecutableEntity	AutosarReference

Connector	Source	Target
Generalization Source -> Destination	CompositionSwComponentType	AutosarReference
Generalization Source -> Destination	BswModuleEntry	AutosarReference
Generalization Source -> Destination	System	AutosarReference
Generalization Source -> Destination	EcuInstance	AutosarReference
Generalization Source -> Destination	EcuPartition	AutosarReference
Generalization Source -> Destination	AutosarReference	AbstractReference
Generalization Source -> Destination	BswModuleDescription	AutosarReference
Generalization Source -> Destination	ClientServerOperation	AutosarReference

Abstract CHROMOSOMEReference

Element Base Classes: **AbstractReference**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	Node	CHROMOSOMEReference
Generalization Source -> Destination	System	CHROMOSOMEReference
Generalization Source -> Destination	CHROMOSOMEReference	AbstractReference

Connector	Source	Target
Generalization Source -> Destination	Application	CHROMOSOMEReference
Generalization Source -> Destination	Component	CHROMOSOMEReference
Generalization Source -> Destination	ComponentModeInstanceRef	CHROMOSOMEReference
Generalization Source -> Destination	Topic	CHROMOSOMEReference

Abstract EastAdlReference

Element Base Classes: **AbstractReference**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FunctionPort	EastAdlReference
Generalization Source -> Destination	HardwarePin	EastAdlReference
Generalization Source -> Destination	FeatureModel	EastAdlReference
Generalization Source -> Destination	AnalysisFunctionType	EastAdlReference
Generalization Source -> Destination	FunctionConnector	EastAdlReference
Generalization Source -> Destination	FunctionType	EastAdlReference
Generalization Source -> Destination	EastAdlReference	AbstractReference

Connector	Source	Target
Generalization Source -> Destination	VehicleFeature	EastAdlReference
Generalization Source -> Destination	HardwareComponentPrototype	EastAdlReference
Generalization Source -> Destination	FunctionPrototype	EastAdlReference
Generalization Source -> Destination	HardwarePort	EastAdlReference
Generalization Source -> Destination	DesignFunctionPrototype	EastAdlReference
Generalization Source -> Destination	Environment	EastAdlReference
Generalization Source -> Destination	AnalysisFunctionPrototype	EastAdlReference
Generalization Source -> Destination	ErrorBehavior	EastAdlReference
Generalization Source -> Destination	Requirement	EastAdlReference
Generalization Source -> Destination	DesignLevel	EastAdlReference
Generalization Source -> Destination	HardwareComponentType	EastAdlReference
Generalization Source -> Destination	DesignFunctionType	EastAdlReference

Package CHROMOSOMEReferences

Package Notes:

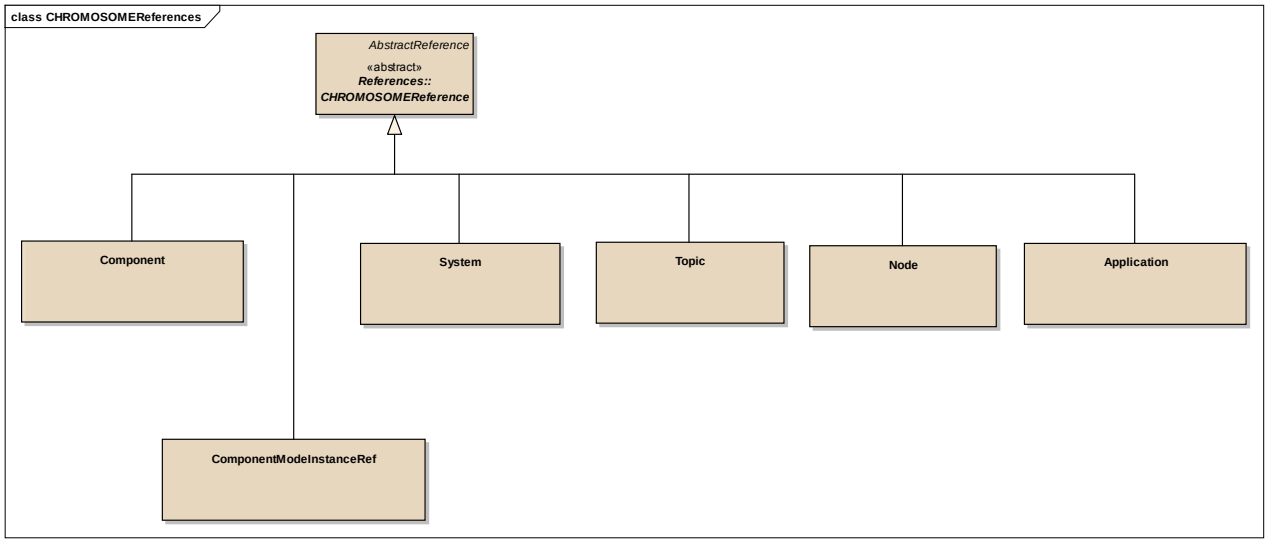


Figure 6: **CHROMOSOMEReferences** - *(Class diagram)*

Diagram Notes:

Class Application

Element Base Classes: **CHROMOSOMEReference**

Element Notes:

A reference to a specific CHROMOSOME application.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeApplicationHealthMonitor	Application
Generalization Source -> Destination	Application	CHROMOSOMEReference
Association Source -> Destination	ChromosomeApplicationSafetyExtension	Application

Class Component

Element Base Classes: **CHROMOSOMEReference**

Element Notes:

A reference to a specific CHROMOSOME component instance.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeHeartbeatSender	Component
Association Source -> Destination	ChromosomeHealthMonitor	Component
Association Source -> Destination	ChromosomeMemoryTest	Component
Association Source -> Destination	ChromosomeCpuSelfTest	Component
Generalization Source -> Destination	Component	CHROMOSOMEReference
Association Source -> Destination	ChromosomeHeartbeatReceiver	Component
Association Source -> Destination	ChromosomeHealthMonitor	Component
Association Source -> Destination	ChromosomeVoter	Component
Association Source -> Destination	ChromosomeComparison	Component

Class ComponentModeInstanceRef

Element Base Classes: **CHROMOSOMEReference**

Element Notes:

Enables referencing of mode of a specific CHROMOSOME component instance.

Connections

Connector	Source	Target
Generalization Source -> Destination	ComponentModeInstanceRef	CHROMOSOMEReference
Association Source -> Destination	ChromosomeComponentModeCondition	ComponentModeInstanceRef

Class Node

Element Base Classes: **CHROMOSOMEReference**

Element Notes:

A reference to a CHROMOSOME node. A node in CHROMOSOME is a hardware unit, which acts as a deployment target for software components (like ECU in Autosar).

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeMemoryTest	Node
Generalization Source -> Destination	Node	CHROMOSOMEReference
Association Source -> Destination	ChromosomeCpuSelfTest	Node
Association Source -> Destination	ChromosomeNodeHealthMonitor	Node

Class System

Element Base Classes: **CHROMOSOMEReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeSystemSafetyExtension	System
Generalization Source -> Destination	System	CHROMOSOMEReference

Class Topic

Element Base Classes: **CHROMOSOMEReference**

Element Notes:

A reference to a specific CHROMOSOME topic.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeVoterParameter	Topic
Association Source -> Destination	ChromosomeErrorEventCondition	Topic
Association Source -> Destination	CHROMOSOMEHealthMonitorNotification	Topic
Generalization Source -> Destination	Topic	CHROMOSOMEReference
Association Source -> Destination	ChromosomeComparisonParameter	Topic

Package EASTADLReferences

Package Notes:

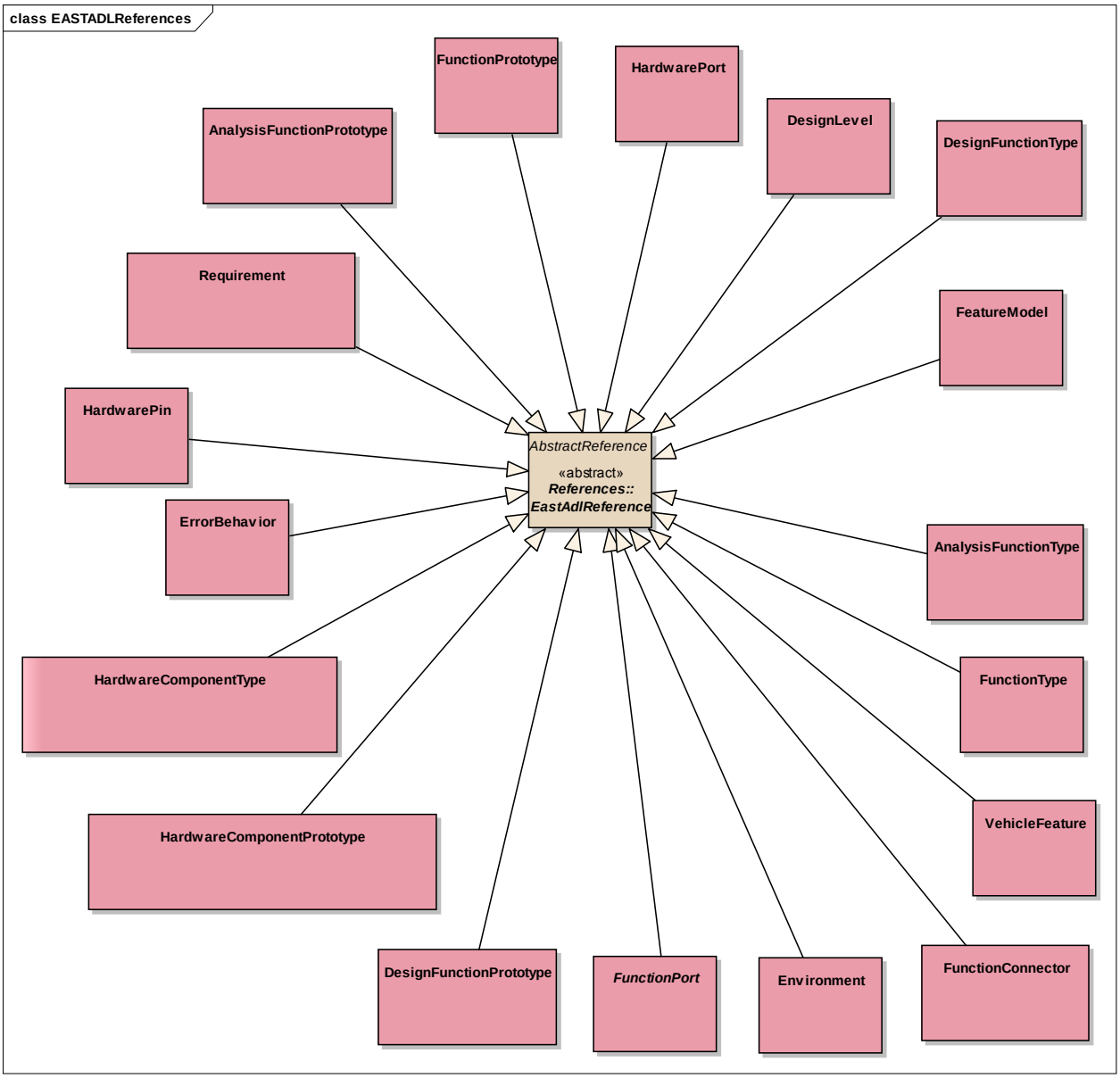


Figure 7: **EASTADLReferences** - *(Class diagram)*

Diagram Notes:

Class AnalysisFunctionPrototype

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	AllocationTargets	AnalysisFunctionPrototype
Dependency Source -> Destination	EMPAnalysis	AnalysisFunctionPrototype
Generalization Source -> Destination	AnalysisFunctionPrototype	EastAdlReference
Association instanceRef.context Source -> Destination	ErrorModelPrototype_analysisFunctionTarget	AnalysisFunctionPrototype
Association instanceRef.target Source -> Destination	ErrorModelPrototype_analysisFunctionTarget	AnalysisFunctionPrototype

Class AnalysisFunctionType

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	FunctionalSafetyExtension	AnalysisFunctionType
Generalization Source -> Destination	AnalysisFunctionType	EastAdlReference
Association Source -> Destination	EMTypeAnalysis	AnalysisFunctionType

Class DesignFunctionPrototype

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	ErrorModelPrototype_designFunctionTarget	DesignFunctionPrototype
Association Source -> Destination	ErrorModelPrototype_designFunctionTarget	DesignFunctionPrototype
Generalization Source -> Destination	DesignFunctionPrototype	EastAdlReference
Dependency Source -> Destination	EMPDesign	DesignFunctionPrototype
Association Source -> Destination	AllocationTargets	DesignFunctionPrototype

Class DesignFunctionType

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	SoftwareSafetyExtension	DesignFunctionType
Generalization Source -> Destination	DesignFunctionType	EastAdlReference
Association	EMTypeDesign	DesignFunctionType

Connector	Source	Target
Source -> Destination		

Class DesignLevel

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	DesignLevelSafetyExtension	DesignLevel
Generalization Source -> Destination	DesignLevel	EastAdlReference

Class Environment

Element Base Classes: **EastAdlReference**

Element Notes:

Class used to reference the EAST-ADL Environment Element

Environmental Elements describe elements that have the potential to influence the vehicle behavior during the analyzed operational situation.

(e.g. main road, trees next to the road, buildings next to the road, snow,...)

Connections

Connector	Source	Target
Association Source -> Destination	OperationalSituation	Environment

Connector	Source	Target
Generalization Source -> Destination	Environment	EastAdlReference

Class ErrorBehavior

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorBehavior	AbstractErrorBehavior
Generalization Source -> Destination	ErrorBehavior	EastAdlReference
Aggregation Source -> Destination	FailureLogicFormula	ErrorBehavior

Class FeatureModel

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	HazardandRiskSafetyExtension	FeatureModel
Generalization Source -> Destination	FeatureModel	EastAdlReference

Class FunctionConnector

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FunctionConnector	EastAdlReference
Association Source -> Destination	HardwareSoftwareInterfaceElement	FunctionConnector

Class FunctionPort

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Dependency Source -> Destination	MFPFunctionPort	FunctionPort
Generalization Source -> Destination	FunctionPort	EastAdlReference
Association Source -> Destination	HardwareSoftwareInterfaceElement	FunctionPort
Association instanceRef.target Source -> Destination	FaultFailurePort_functionTarget	FunctionPort

Class FunctionPrototype

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FunctionPrototype	EastAdlReference
Association instanceRef.context Source -> Destination	FaultFailurePort_functionTarget	FunctionPrototype

Class FunctionType

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FunctionType	EastAdlReference

Class HardwareComponentPrototype

Element Base Classes: **EastAdlReference**

Element Notes:

Class used to reference an EAST-ADL HardwareComponentPrototype element.

Connections

Connector	Source	Target
Generalization Source -> Destination	HardwareComponentPrototype	EastAdlReference
Dependency Source -> Destination	EMPHwComponent	HardwareComponentPrototype
Association Source -> Destination	AllocationTargets	HardwareComponentPrototype
Association Source -> Destination	FaultFailurePort_hwPortTarget	HardwareComponentPrototype
Association Source -> Destination	HwComponentScopeInstanceRef	HardwareComponentPrototype
Dependency Source -> Destination	HWComponentPrototypeScope	HardwareComponentPrototype
Association Source -> Destination	HwComponentScopeInstanceRef	HardwareComponentPrototype
Association instanceRef.context Source -> Destination	ErrorModelPrototype_hwTarget	HardwareComponentPrototype
Association instanceRef.target Source -> Destination	ErrorModelPrototype_hwTarget	HardwareComponentPrototype

Class HardwareComponentType

Element Base Classes: **EastAdlReference**

Element Notes:

Class used to reference an EAST-ADL HardwareComponentType element.

Connections

Connector	Source	Target
Association Source -> Destination	HardwareSafetyExtension	HardwareComponentType
Association Source -> Destination	HwComponentScopeInstanceRef	HardwareComponentType
Association Source -> Destination	EMTypeHwComponent	HardwareComponentType
Generalization Source -> Destination	HardwareComponentType	EastAdlReference

Class HardwarePin

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	HardwareSoftwareInterfaceElement	HardwarePin
Generalization Source -> Destination	HardwarePin	EastAdlReference
Dependency Source -> Destination	MFPHardwarePin	HardwarePin
Association instandeRef.target Source -> Destination	FaultFailurePort_hwPinTarget	HardwarePin

Class HardwarePort

Element Base Classes: **EastAdlReference**

Element Notes:

Connections

Connector	Source	Target
Dependency Source -> Destination	MFPHardwarePort	HardwarePort
Generalization Source -> Destination	HardwarePort	EastAdlReference
Association Source -> Destination	FaultFailurePort_hwPortTarget	HardwarePort

Class Requirement

Element Base Classes: **EastAdlReference**

Element Notes:

Class used to reference an EAST-ADL Requirement element.

Connections

Connector	Source	Target
Association Source -> Destination	AbstractSafetyRequirement	Requirement
Association Source -> Destination	Item	Requirement
Generalization Source -> Destination	Requirement	EastAdlReference

Class VehicleFeature

Element Base Classes: **EastAdlReference**

Element Notes:

Class used to reference an EAST-ADL VehicleFeature element.

Connections

Connector	Source	Target
Generalization Source -> Destination	VehicleFeature	EastAdlReference
Association Source -> Destination	Item	VehicleFeature
Association Source -> Destination	AllocationTargets	VehicleFeature

Package AUTOSARReferences

Package Notes:

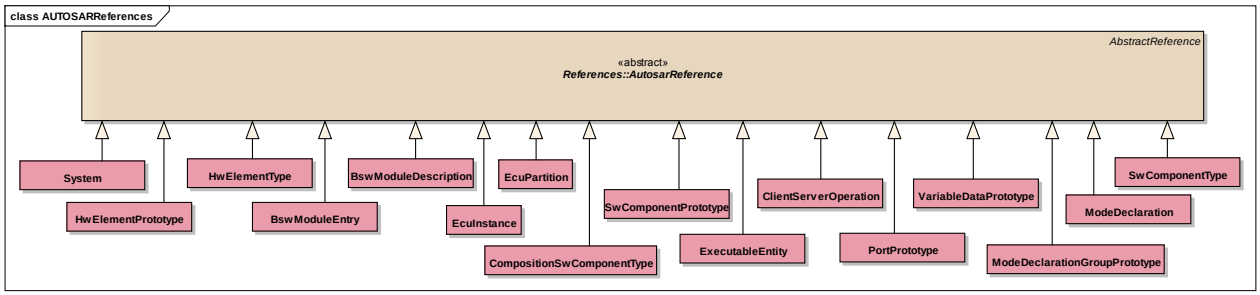


Figure 8: **AUTOSARReferences** - *(Class diagram)*

Diagram Notes:

Class BswModuleDescription

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	EMPBswModule	BswModuleDescription
Association Source -> Destination	EMTypeBswModule	BswModuleDescription
Generalization Source -> Destination	BswModuleDescription	AutosarReference

Class BswModuleEntry

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	MFPBswPort	BswModuleEntry
Association Source -> Destination	AllocationTargets	BswModuleEntry
Generalization Source -> Destination	BswModuleEntry	AutosarReference

Class ClientServerOperation

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target

Connector	Source	Target
Association Source -> Destination	AutosarClientServerOperationInstanceRef	ClientServerOperation
Generalization Source -> Destination	ClientServerOperation	AutosarReference

Class CompositionSwComponentType

Element Base Classes: **AutosarReference**

Element Notes:

Class used to reference instances of software component compositions in an AUTOSAR model.

Connections

Connector	Source	Target
Generalization Source -> Destination	CompositionSwComponentType	AutosarReference

Class EcuInstance

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	EMTypeApplicationEnvironment	EcuInstance
Generalization Source -> Destination	EcuInstance	AutosarReference

Class EcuPartition

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	EMTypeApplicationEnvironment	EcuPartition
Generalization Source -> Destination	EcuPartition	AutosarReference

Class ExecutableEntity

Element Base Classes: **AutosarReference**

Element Notes:

Class used to reference instances of AUTOSAR runnable entity elements.

Connections

Connector	Source	Target
Association Source -> Destination	AutosarCheckPoint	ExecutableEntity
Generalization Source -> Destination	ExecutableEntity	AutosarReference
Association Source -> Destination	RunnableEntitySequenceElement	ExecutableEntity

Class HwElementPrototype

Element Base Classes: **AutosarReference**

Element Notes:

class used to reference an AUTOSAR-Element used to model a safety hardware component

Connections

Connector	Source	Target
Dependency Source -> Destination	HWElementPrototypeScope	HwElementPrototype
Generalization Source -> Destination	HwElementPrototype	AutosarReference
Association Source -> Destination	HWElementScopeInstanceRef	HwElementPrototype
Association Source -> Destination	HWElementScopeInstanceRef	HwElementPrototype
Association Source -> Destination	AllocationTargets	HwElementPrototype

Class HwElementType

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	HWElementScopeInstanceRef	HwElementType
Generalization Source -> Destination	HwElementType	AutosarReference
Association Source -> Destination	AutosarHardwareSafetyExtension	HwElementType

Class ModeDeclaration

Element Base Classes: **AutosarReference**

Element Notes:

Class used to reference an AUTOSAR mode declaration.

Connections

Connector	Source	Target
Association Source -> Destination	AutosarContext	ModeDeclaration
Generalization Source -> Destination	ModeDeclaration	AutosarReference

Class ModeDeclarationGroupPrototype

Element Base Classes: **AutosarReference**

Element Notes:

A reference to an AUTOSAR mode for a specific port of a specific software component within a software component composition defined in another model / file.

Connections

Connector	Source	Target
Generalization Source -> Destination	ModeDeclarationGroupPrototype	AutosarReference
Association Source -> Destination	AutosarModeInPortFromComponentInCompositionInstanceRef	ModeDeclarationGroupPrototype

Class PortPrototype

Element Base Classes: **AutosarReference**

Element Notes:

Class used to reference AUTOSAR port prototype elements.

Connections

Connector	Source	Target
Association Source -> Destination	AutosarAbstractPortInComponentInstanceRef	PortPrototype
Generalization Source -> Destination	PortPrototype	AutosarReference
Association Source -> Destination	AutosarPortInCompositionInstanceRef	PortPrototype

Class SwComponentPrototype

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	AllocationTargets	SwComponentPrototype
Generalization Source -> Destination	SwComponentPrototype	AutosarReference
Association Source -> Destination	AutosarAbstractComponentInstanceRef	SwComponentPrototype
Association Source -> Destination	AutosarComponentInCompositionInstanceRef	SwComponentPrototype

Class SwComponentType

Element Base Classes: **AutosarReference**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	EMTypeSwComponent	SwComponentType
Generalization Source -> Destination	SwComponentType	AutosarReference
Association Source -> Destination	AutosarVfbSafetyExtension	SwComponentType
Association Source -> Destination	AutosarAbstractComponentInstanceRef	SwComponentType

Class System

Element Base Classes: **AutosarReference**

Element Notes:

Class used to reference an instance of an AUTOSAR System element.

Connections

Connector	Source	Target
Association Source -> Destination	AutosarSystemSafetyExtension	System
Generalization Source -> Destination	System	AutosarReference

Class VariableDataPrototype

Element Base Classes: **AutosarReference**

Element Notes:

A reference to an AUTOSAR variable data prototype of a specific software component within a software component composition which is defined in another model / file.

Connections

Connector	Source	Target
Generalization Source -> Destination	VariableDataPrototype	AutosarReference
Association Source -> Destination	AutosarVariableDataPrototypeInstanceRef	VariableDataPrototype
Association Source -> Destination	InterfaceCRC	VariableDataPrototype

Package SafetyExtensions

Package Notes:

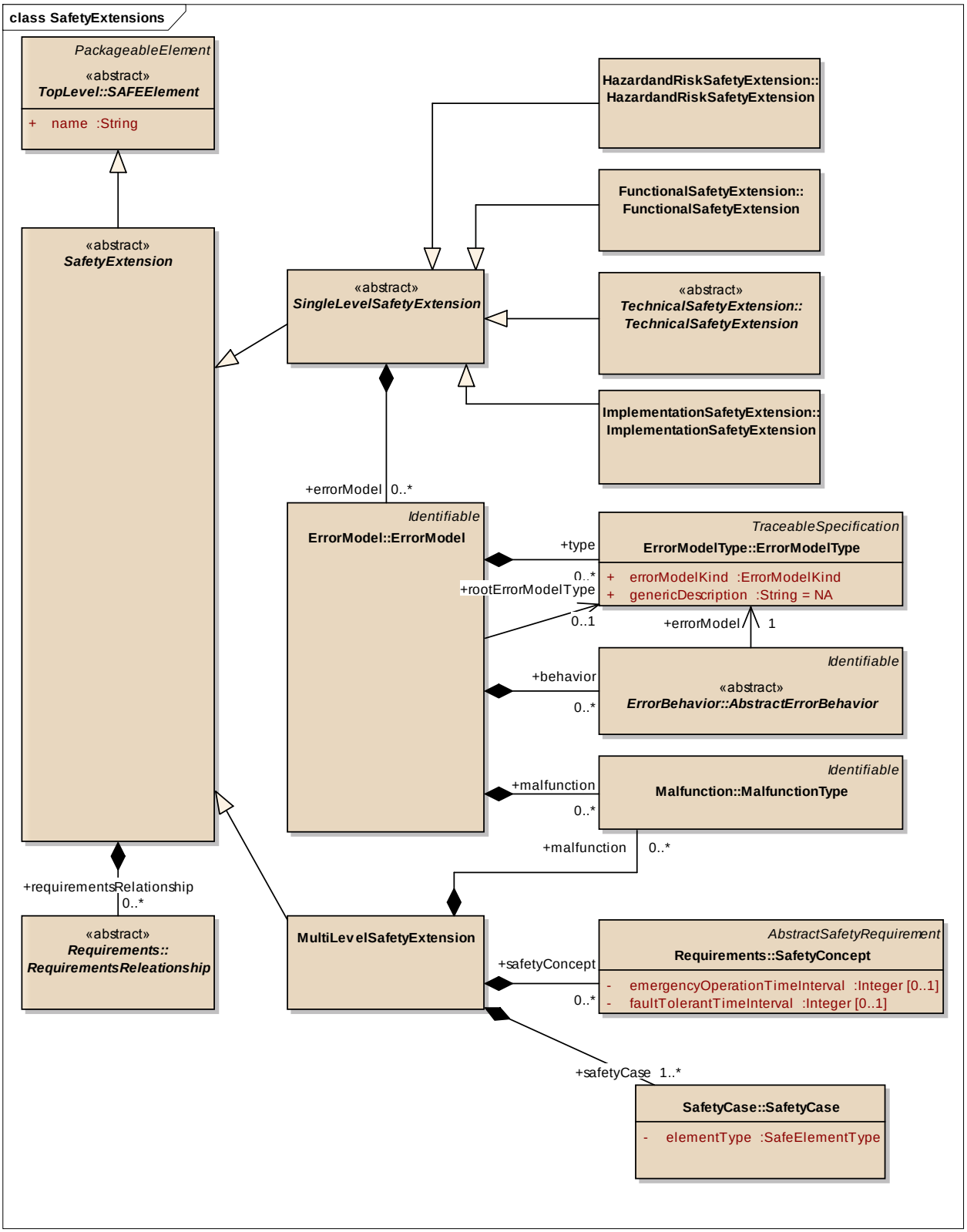


Figure 9: **SafetyExtensions** - *(Class diagram)*

Diagram Notes:

Class MultiLevelSafetyExtension

Element Base Classes: **SafetyExtension**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	MalfunctionType	MultiLevelSafetyExtension
Aggregation Source -> Destination	SafetyConcept	MultiLevelSafetyExtension
Aggregation Source -> Destination	SafetyCase	MultiLevelSafetyExtension
Generalization Source -> Destination	MultiLevelSafetyExtension	SafetyExtension

Abstract SafetyExtension

Element Base Classes: **SAFEElement**

Element Notes:

The abstract parent class of the different abstraction-level specific safety extensions.

Depending on the specific level of abstraction (HazardRiskModel, FunctionalSafetyExtension, TechnicalSafetyExtension, ...), the following restriction apply:

- the subtypes of AbstractSafetyRequirement (TechnicalSafetyRequirement, FunctionalSafetyRequirement, SafetyGoal, HW/SWSafetyRequirement) shall be used in the respective level of abstraction via the "requirements" relation of the safety extension

- the error model associated with the safety extension via the "erroModel" relation shall only allow to reference system model artifacts which are visibly at the level of abstraction of the safety extension (e.g. an AUTOSAR software component is not visible in the HazardandRiskModel)

Connections

Connector	Source	Target
Generalization Source -> Destination	SingleLevelSafetyExtension	SafetyExtension
Generalization Source -> Destination	SafetyExtension	SAFEElement
Aggregation Source -> Destination	RequirementsReleationship	SafetyExtension
Generalization Source -> Destination	MultiLevelSafetyExtension	SafetyExtension

Abstract SingleLevelSafetyExtension

Element Base Classes: **SafetyExtension**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModel	SingleLevelSafetyExtension
Generalization Source -> Destination	SingleLevelSafetyExtension	SafetyExtension
Generalization Source -> Destination	HazardandRiskSafetyExtension	SingleLevelSafetyExtension
Generalization Source -> Destination	ImplementationSafetyExtension	SingleLevelSafetyExtension
Generalization Source -> Destination	FunctionalSafetyExtension	SingleLevelSafetyExtension
Generalization Source -> Destination	TechnicalSafetyExtension	SingleLevelSafetyExtension

Package FunctionalSafetyExtension

Package Notes:

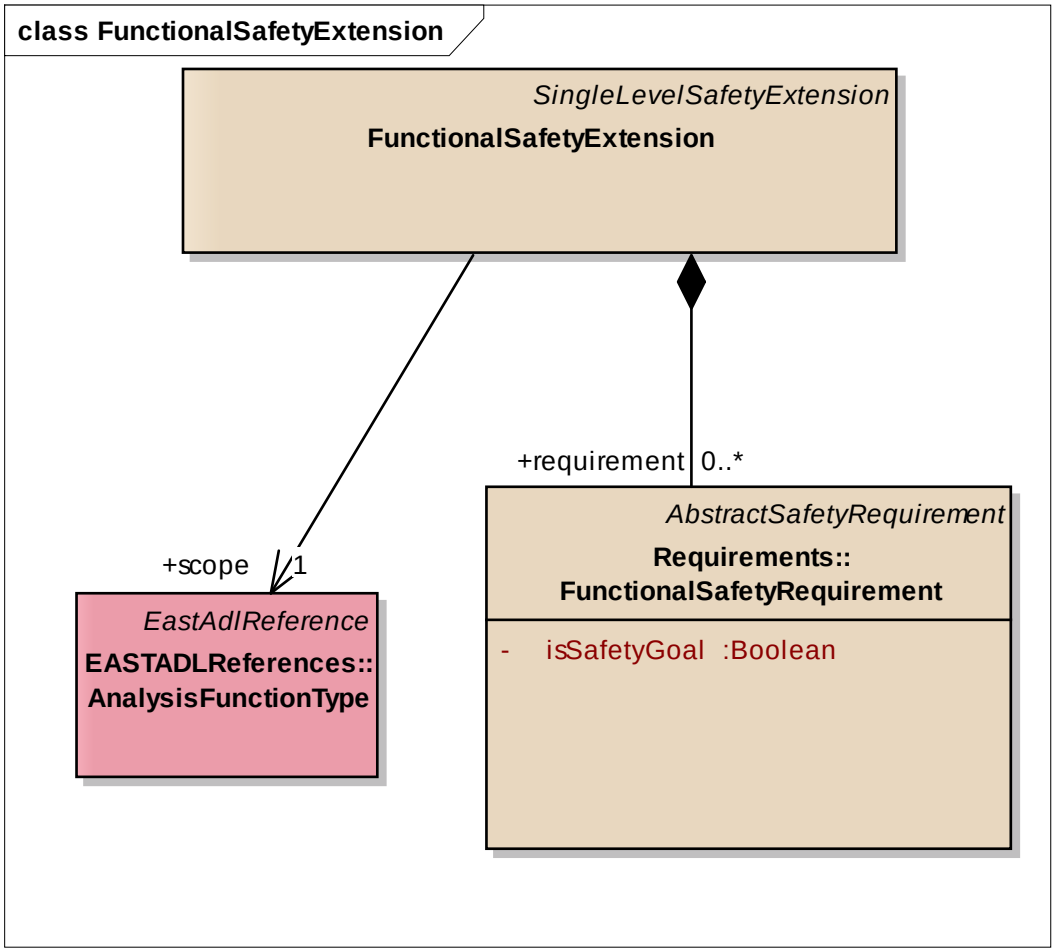


Figure 10: **FunctionalSafetyExtension** - *(Class diagram)*

Diagram Notes:

Class FunctionalSafetyExtension

Element Base Classes: **SingleLevelSafetyExtension**

Element Notes:

The FunctionalSafetyExtension is used as interface to the AnalysisLevel defined in EAST-ADL. This extension specifies the add-on needed to model the functional safety concept defined in the ISO 26262 part 3 chapter 8.

Connections

Connector	Source	Target
Association Source -> Destination	FunctionalSafetyExtension	AnalysisFunctionType
Generalization Source -> Destination	FunctionalSafetyExtension	SingleLevelSafetyExtension
Aggregation Source -> Destination	FunctionalSafetyRequirement	FunctionalSafetyExtension

Package HazardandRiskSafetyExtension

Package Notes:

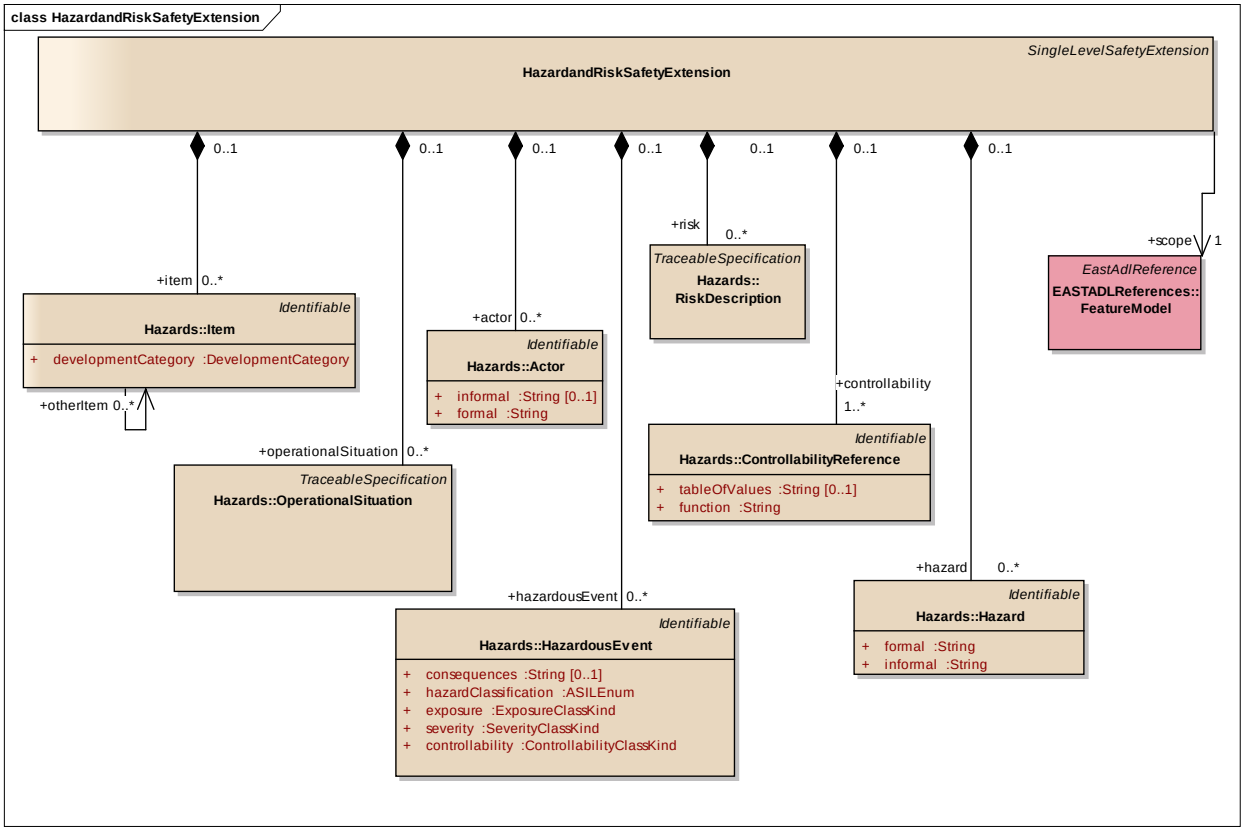


Figure 11: **HazardandRiskSafetyExtension** - *(Class diagram)*

Diagram Notes:

Class HazardandRiskSafetyExtension

Element Base Classes: **SingleLevelSafetyExtension**

Element Notes:

The HazardAndRiskSafetyExtension is used as interface to the VehicleFeature defined in EAST-ADL. This extension specifies the add-on needed to model the hazard analysis and risk assessment defined in ISO 26262 part 3 chapter 7. Further details according to modeling of hazards and safety goals are described in D3.1.1.b

Connections

Connector	Source	Target
Aggregation Source -> Destination	FunctionalSafetyRequirement	HazardandRiskSafetyExtension

Connector	Source	Target
Aggregation Source -> Destination	OperationalSituation	HazardandRiskSafetyExtension
Aggregation Source -> Destination	HazardousEvent	HazardandRiskSafetyExtension
Association Source -> Destination	HazardandRiskSafetyExtension	FeatureModel
Generalization Source -> Destination	HazardandRiskSafetyExtension	SingleLevelSafetyExtension
Aggregation Source -> Destination	Item	HazardandRiskSafetyExtension
Aggregation Source -> Destination	Hazard	HazardandRiskSafetyExtension
Aggregation Source -> Destination	ControllabilityReference	HazardandRiskSafetyExtension
Aggregation Source -> Destination	Actor	HazardandRiskSafetyExtension
Aggregation Source -> Destination	RiskDescription	HazardandRiskSafetyExtension

Package ImplementationSafetyExtension

Package Notes:

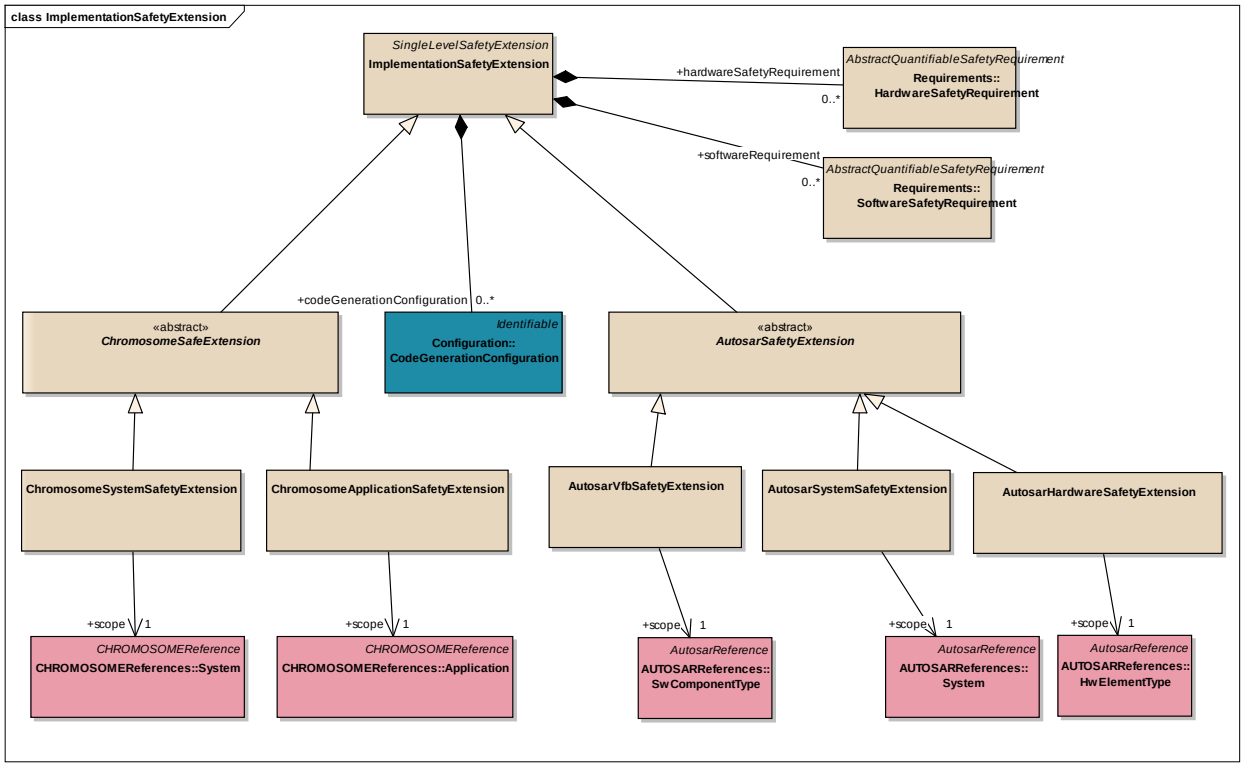


Figure 12: **ImplementationSafetyExtension** - *(Class diagram)*

Diagram Notes:

Class AutosarHardwareSafetyExtension

Element Base Classes: **AutosarSafetyExtension**

Element Notes:

This class represent the Safety Extension point for referenced element of Hardware Element as part of a component (respectively AUTOSAR HW Element Type) to allow the capture of hardware failure and summary failure quantified contribution to HWComponent.

Connections

Connector	Source	Target
Association Unspecified	AutosarHardwareSafetyExtension	HWPartFailureAnalysis
Association Unspecified	HWPartFailure	AutosarHardwareSafetyExtension

Connector	Source	Target
Association Source -> Destination	AutosarHardwareSafetyExtension	HwElementType
Generalization Source -> Destination	AutosarHardwareSafetyExtension	AutosarSafetyExtension

Abstract AutosarSafetyExtension

Element Base Classes: **ImplementationSafetyExtension**

Element Notes:

This element represents abstract AUTOSAR SAFE extensions. These are extensions to AUTOSAR which define software safety mechanisms related to AUTOSAR elements.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarVfbSafetyExtension	AutosarSafetyExtension
Generalization Source -> Destination	AutosarSafetyExtension	ImplementationSafetyExtension
Generalization Source -> Destination	AutosarSystemSafetyExtension	AutosarSafetyExtension
Generalization Source -> Destination	AutosarHardwareSafetyExtension	AutosarSafetyExtension

Class AutosarSystemSafetyExtension

Element Base Classes: **AutosarSafetyExtension**

Element Notes:

This meta-class defines SAFE extensions which are specified for the AUTOSAR system level.

Connections

Connector	Source	Target
Association Source -> Destination	AutosarSystemSafetyExtension	System
Generalization Source -> Destination	AutosarSystemSafetyExtension	AutosarSafetyExtension

Class AutosarVfbSafetyExtension

Element Base Classes: **AutosarSafetyExtension**

Element Notes:

This meta-class defines the element used for specifications of SAFE extensions at VFB level on AUTOSAR. This means elements which relate to a composition directly, not yet in a given system context.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarVfbSafetyExtension	AutosarSafetyExtension
Association Source -> Destination	AutosarVfbSafetyExtension	SwComponentType

Class ChromosomeApplicationSafetyExtension

Element Base Classes: **ChromosomeSafeExtension**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeApplicationSafetyExtension	Application

Connector	Source	Target
Generalization Source -> Destination	ChromosomeApplicationSafetyExtension	ChromosomeSafeExtension

Abstract ChromosomeSafeExtension

Element Base Classes: **ImplementationSafetyExtension**

Element Notes:

This element represents abstract Chromosome SAFE extensions.

Connections

Connector	Source	Target
Generalization Source -> Destination	ChromosomeSafeExtension	ImplementationSafetyExtension
Generalization Source -> Destination	ChromosomeSystemSafetyExtension	ChromosomeSafeExtension
Generalization Source -> Destination	ChromosomeApplicationSafetyExtension	ChromosomeSafeExtension

Class ChromosomeSystemSafetyExtension

Element Base Classes: **ChromosomeSafeExtension**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeSystemSafetyExtension	System

Connector	Source	Target
Generalization Source -> Destination	ChromosomeSystemSafetyExtension	ChromosomeSafeExtension

Class ImplementationSafetyExtension

Element Base Classes: **SingleLevelSafetyExtension**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ChromosomeSafeExtension	ImplementationSafetyExtension
Aggregation Source -> Destination	SoftwareSafetyRequirement	ImplementationSafetyExtension
Aggregation Source -> Destination	CodeGenerationConfiguration	ImplementationSafetyExtension
Generalization Source -> Destination	AutosarSafetyExtension	ImplementationSafetyExtension
Generalization Source -> Destination	ImplementationSafetyExtension	SingleLevelSafetyExtension
Aggregation Source -> Destination	HardwareSafetyRequirement	ImplementationSafetyExtension

Package TechnicalSafetyExtension

Package Notes:

This package describes the TechnicalSafetyExtension as defined in the SafetyExtension-Diagram.

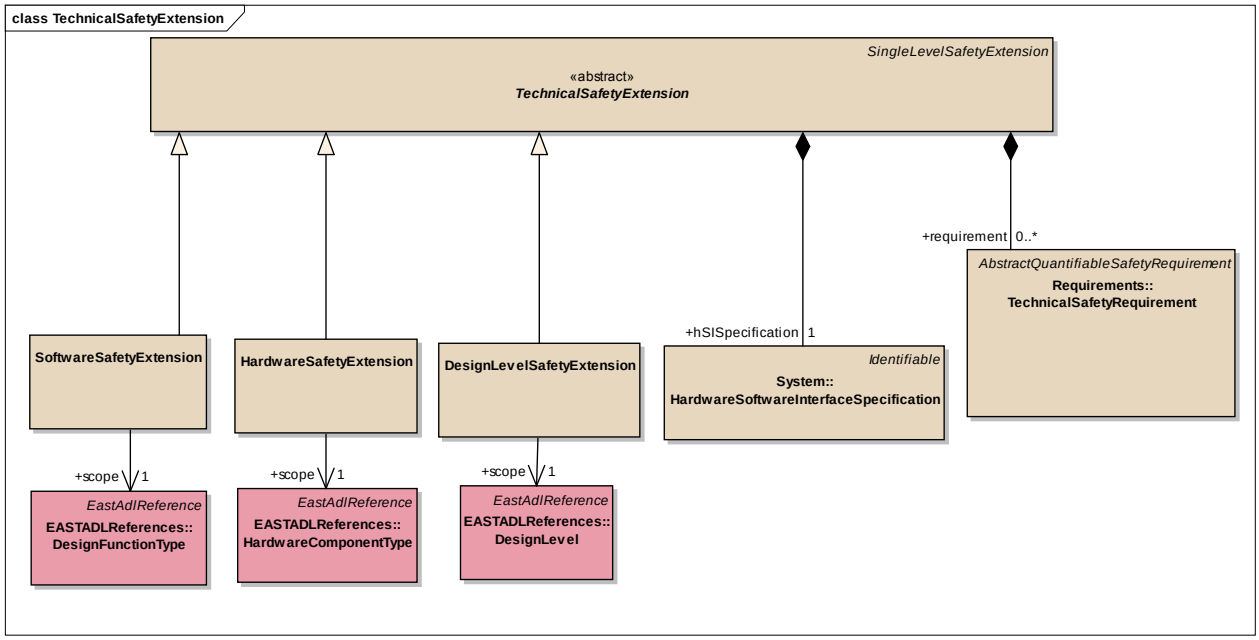


Figure 13: **TechnicalSafetyExtension** - *(Class diagram)*

Diagram Notes:

Class DesignLevelSafetyExtension

Element Base Classes: **TechnicalSafetyExtension**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	DesignLevelSafetyExtension	DesignLevel
Generalization Source -> Destination	DesignLevelSafetyExtension	TechnicalSafetyExtension

Class HardwareSafetyExtension

Element Base Classes: **TechnicalSafetyExtension**

Element Notes:

This class represent the Safety Extension point for referenced element of Hardware Component Type (respectively from EAST-ADL) to allow the capture of hardware failure and summary failure and analysis results.

Connections

Connector	Source	Target
Association Source -> Destination	HardwareSafetyExtension	HardwareComponentType
Aggregation Source -> Destination	HardwareComponentFailure	HardwareSafetyExtension
Aggregation Source -> Destination	HardwareFailureAnalysis	HardwareSafetyExtension
Generalization Source -> Destination	HardwareSafetyExtension	TechnicalSafetyExtension

Class SoftwareSafetyExtension

Element Base Classes: **TechnicalSafetyExtension**

Element Notes:

The SafetySoftwareExtension is used to specify the implementation of the safety relevant SoftwareDesignComponents that are allocated to the SafetySoftwareDesign.

Connections

Connector	Source	Target
Generalization Source -> Destination	SoftwareSafetyExtension	TechnicalSafetyExtension
Association Source -> Destination	SoftwareSafetyExtension	DesignFunctionType

Abstract TechnicalSafetyExtension

Element Base Classes: **SingleLevelSafetyExtension**

Element Notes:

The TechnicalSafetyExtension is used as interface to the DesignLevel defined in EAST-ADL. This extension specifies the add-on needed to model a specific technical solution that is derived based on the functional safety concept. It contains the

- technical safety concept (ISO 26262 part 4 chapter 7)
- hardware software interface specification (ISO 26262 part 4 chapter 7.4.6)

Connections

Connector	Source	Target
Aggregation Source -> Destination	TechnicalSafetyRequirement	TechnicalSafetyExtension
Generalization Source -> Destination	SoftwareSafetyExtension	TechnicalSafetyExtension
Aggregation Source -> Destination	HardwareSoftwareInterfaceSpecification	TechnicalSafetyExtension
Generalization Source -> Destination	DesignLevelSafetyExtension	TechnicalSafetyExtension
Generalization Source -> Destination	TechnicalSafetyExtension	SingleLevelSafetyExtension
Generalization Source -> Destination	HardwareSafetyExtension	TechnicalSafetyExtension

Package TopLevel

Package Notes:

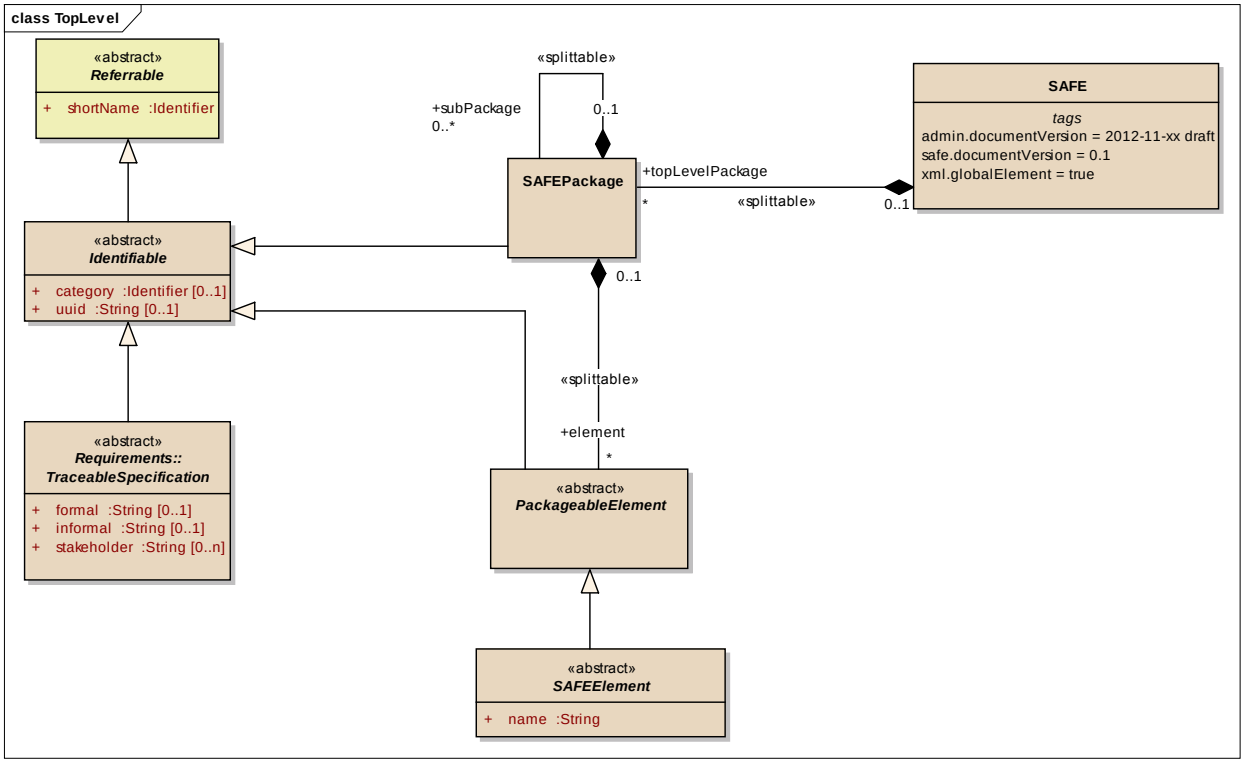


Figure 14: **TopLevel** - *(Class diagram)*

Diagram Notes:

Top-Level structure is built up similar to the TopLevelStructure defined in AUTOSAR.

Abstract PackageableElement

Element Base Classes: **Identifiable**

Element Notes:

This meta-class specifies the ability to be a member of a SAFE package.

Connections

Connector	Source	Target
Generalization Source -> Destination	Restrictable	PackageableElement
Aggregation Source -> Destination	PackageableElement	SAFEPackage
Generalization Source -> Destination	Variant	PackageableElement

Connector	Source	Target
Generalization Source -> Destination	SAFEElement	PackageableElement
Generalization Source -> Destination	PackageableElement	Identifiable

Abstract Referrable

Element Base Classes:

Element Notes:

Instances of this class can be referred to by their identifier (while adhering to namespace borders).

Connections

Connector	Source	Target
Generalization Source -> Destination	Identifiable	Referrable
Generalization Source -> Destination	HWFault	Referrable
Generalization Source -> Destination	HWFailureMode	Referrable
Generalization Source -> Destination	HWComponentQuantifiedFMFromPart	Referrable
Generalization Source -> Destination	HWFailureRate	Referrable

Attributes

Attribute	Notes	Default
shortName Identifier	This specifies an identifying shortName for the object. It needs to be unique within its context and is intended for humans but even more for technical reference.	

Class SAFE

Element Base Classes:

Element Notes:

The root element of a SAFE description, also the root element in corresponding XML documents.

Connections

Connector	Source	Target
Aggregation Source -> Destination	SAFEPackage	SAFE

Abstract SAFEElement

Element Base Classes: **PackageableElement**

Element Notes:

This class serves as a base class for all SAFE class that represent something (i.e. not technical class).

Connections

Connector	Source	Target
Generalization Source -> Destination	SafetyExtension	SAFEElement
Generalization Source -> Destination	SAFEElement	PackageableElement

Attributes

Attribute	Notes	Default
name String	Optional descriptive name of the SAFEElement, this name does not have the length restrictions as found for the AUTOSAR Identfiable shortName.	

Class SAFEPackage

Element Base Classes: **Identifiable**

Element Notes:

Used for organization of the packageable elements in the model.

Semantics:

SAFEPackages can be organized hierarchically, where each level may contain a number of SAFEPackageableElements.

Connections

Connector	Source	Target
Generalization Source -> Destination	SAFEPackage	Identifiable
Aggregation Source -> Destination	PackageableElement	SAFEPackage
Aggregation Source -> Destination	SAFEPackage	SAFE
Aggregation Source -> Destination	SAFEPackage	SAFEPackage

Abstract Identifiable

Element Base Classes: **Referrable**

Element Notes:

Instances of this class can be referred to by their identifier (within the namespace borders). In addition to this, Identifiables are objects which contribute significantly to the overall structure of an AUTOSAR description. In particular, Identifiables might contain Identifiables.

Connections

Connector	Source	Target
Generalization Source -> Destination	MemorySelfTestConfig	Identifiable
Generalization Source -> Destination	Actor	Identifiable
Generalization Source -> Destination	QuantitativeErrorModel	Identifiable

Connector	Source	Target
Generalization Source -> Destination	MTCompositeElementPrototype	Identifiable
Generalization Source -> Destination	VoterConfig	Identifiable
Generalization Source -> Destination	ErrorModelPrototype	Identifiable
Generalization Source -> Destination	Identifiable	Referrable
Generalization Source -> Destination	FormulaExpression	Identifiable
Generalization Source -> Destination	MalfunctionType	Identifiable
Generalization Source -> Destination	MalfunctionPrototype	Identifiable
Generalization Source -> Destination	HazardousEvent	Identifiable
Generalization Source -> Destination	HWComponentQuantifiedFMFromPart	Identifiable
Generalization Source -> Destination	ConfigParameter	Identifiable
Generalization Source -> Destination	SAFEPackage	Identifiable
Generalization Source -> Destination	HardwareSoftwareInterfaceSpecification	Identifiable
Generalization Source -> Destination	HWPartFailureAnalysis	Identifiable
Association Source -> Destination	Variant	Identifiable
Generalization Source -> Destination	MalfunctionMapping	Identifiable
Generalization Source -> Destination	CpuSelfTestConfig	Identifiable

Connector	Source	Target
Association Source -> Destination	Variant	Identifiable
Generalization Source -> Destination	RunnableEntitySequenceElement	Identifiable
Generalization Source -> Destination	EvaluationFunction	Identifiable
Generalization Source -> Destination	Tactic	Identifiable
Association Source -> Destination	Restrictable	Identifiable
Generalization Source -> Destination	Item	Identifiable
Generalization Source -> Destination	HWQuantitativeFailureAnalysis	Identifiable
Generalization Source -> Destination	AllocationTargets	Identifiable
Generalization Source -> Destination	HardwareSoftwareInterfaceElement	Identifiable
Generalization Source -> Destination	HardwareComponentFailure	Identifiable
Generalization Source -> Destination	ItemFunctionDescription	Identifiable
Generalization Source -> Destination	ComparisonParameter	Identifiable
Generalization Source -> Destination	FilterParameter	Identifiable
Generalization Source -> Destination	TraceableSpecification	Identifiable
Generalization Source -> Destination	FormulaDocumentation	Identifiable
Generalization Source -> Destination	OperatingMode	Identifiable

Connector	Source	Target
Generalization Source -> Destination	HWFault	Identifiable
Generalization Source -> Destination	Situation	Identifiable
Generalization Source -> Destination	HWPartFailure	Identifiable
Generalization Source -> Destination	CodeGenerationConfiguration	Identifiable
Generalization Source -> Destination	ErrorBehaviorMapping	Identifiable
Generalization Source -> Destination	HWPMHF	Identifiable
Generalization Source -> Destination	ErrorModelMapping	Identifiable
Generalization Source -> Destination	ErrorModel	Identifiable
Generalization Source -> Destination	SafeState	Identifiable
Generalization Source -> Destination	ContextParameter	Identifiable
Generalization Source -> Destination	RangeCheckContext	Identifiable
Generalization Source -> Destination	AbstractErrorBehavior	Identifiable
Generalization Source -> Destination	PackageableElement	Identifiable
Generalization Source -> Destination	ControllabilityReference	Identifiable
Generalization Source -> Destination	HardwareFailureAnalysis	Identifiable
Generalization Source -> Destination	HWElementFailureRateClass	Identifiable

Connector	Source	Target
Generalization Source -> Destination	VoterParameter	Identifiable
Generalization Source -> Destination	HWFailureClassContainer	Identifiable
Generalization Source -> Destination	ControlBlock	Identifiable
Generalization Source -> Destination	MemoryRange	Identifiable
Generalization Source -> Destination	Hazard	Identifiable
Generalization Source -> Destination	CheckPoint	Identifiable
Generalization Source -> Destination	FaultFailurePropagationLink	Identifiable
Generalization Source -> Destination	HeartbeatConfig	Identifiable

Attributes

Attribute	Notes	Default
category Identifier	This element assigns a category to the parent element. The category is intended to specialize the usage and/or the content identifiable object. Such a specialization may also impose particular semantic constraints on the entire substructure (not only the identifiable itself).	
uuid String	The purpose of this attribute is to provide a globally unique identifier for an instance of a metaclass. The values of this attribute should be globally unique strings prefixed by the type of identifier. For example, to include a DCE UUID as defined by The Open Group, the UUID would be preceded by "DCE:". The values of this attribute may be used to support merging of different AUTOSAR models. The form of the UUID (Universally Unique Identifier) is taken from a standard defined by the Open Group (was Open Software Foundation). This standard is widely used, including by Microsoft for COM (GUIDs) and by many companies for DCE, which is based on CORBA. The method for generating these 128-bit IDs is published in the standard and the effectiveness and uniqueness of the IDs is	

Attribute	Notes	Default
	not in practice disputed. If the id namespace is omitted, DCE is assumed. An example is "DCE:2fac1234-31f8-11b4-a222-08002b34c003".	

Package Configuration

Package Notes:

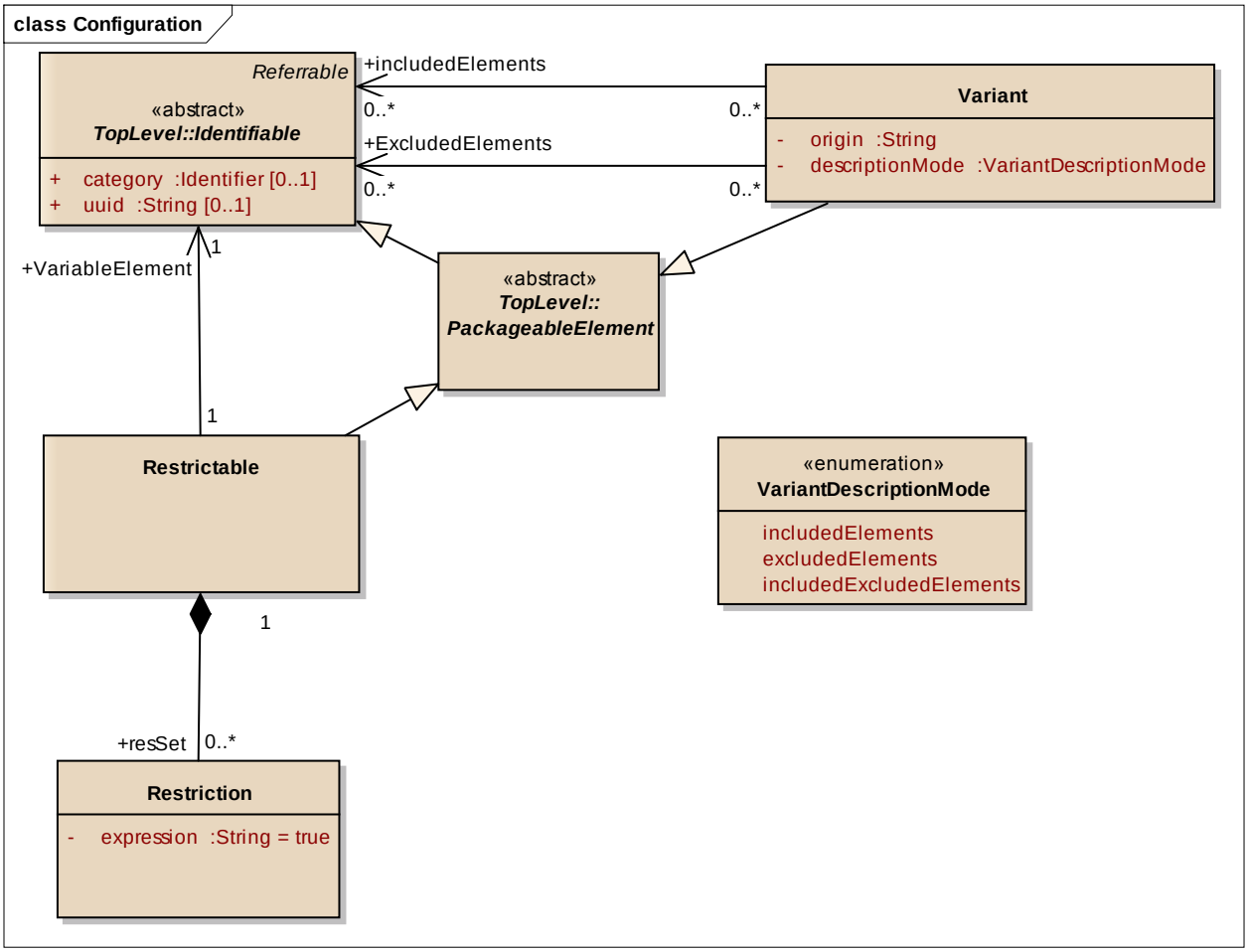


Figure 15: **Configuration** - *(Class diagram)*

Diagram Notes:

Class Restrictable

Element Base Classes: **PackageableElement**

Element Notes:

A RestrictableElement allows connecting SAFE model elements with variant information.

The identified element, pointed to by VariableElement of the class RestrictableElement, is part of a variant if at least one of the defined restrictions evaluate to true.

Connections

Connector	Source	Target
Generalization Source -> Destination	Restrictable	PackageableElement
Association Source -> Destination	Restrictable	Identifiable
Aggregation Source -> Destination	Restriction	Restrictable

Class Restriction

Element Base Classes:

Element Notes:

A Restriction defines an expression that can be evaluated to true or false.

Connections

Connector	Source	Target
Aggregation Source -> Destination	Restriction	Restrictable

Attributes

Attribute	Notes	Default
expression String		true

Class Variant

Element Base Classes: **PackageableElement**

Element Notes:

Represents the "actual" variant.

Connections

Connector	Source	Target
Association Source -> Destination	Variant	Identifiable
Association Source -> Destination	Variant	Identifiable
Generalization Source -> Destination	Variant	PackageableElement

Attributes

Attribute	Notes	Default
origin String		
descriptionMode VariantDescriptionMode		

Enumeration VariantDescriptionMode

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default

Attribute	Notes	Default
includedElements		
excludedElements		
includedExcludedElements		

Package ErrorModel

Package Notes:

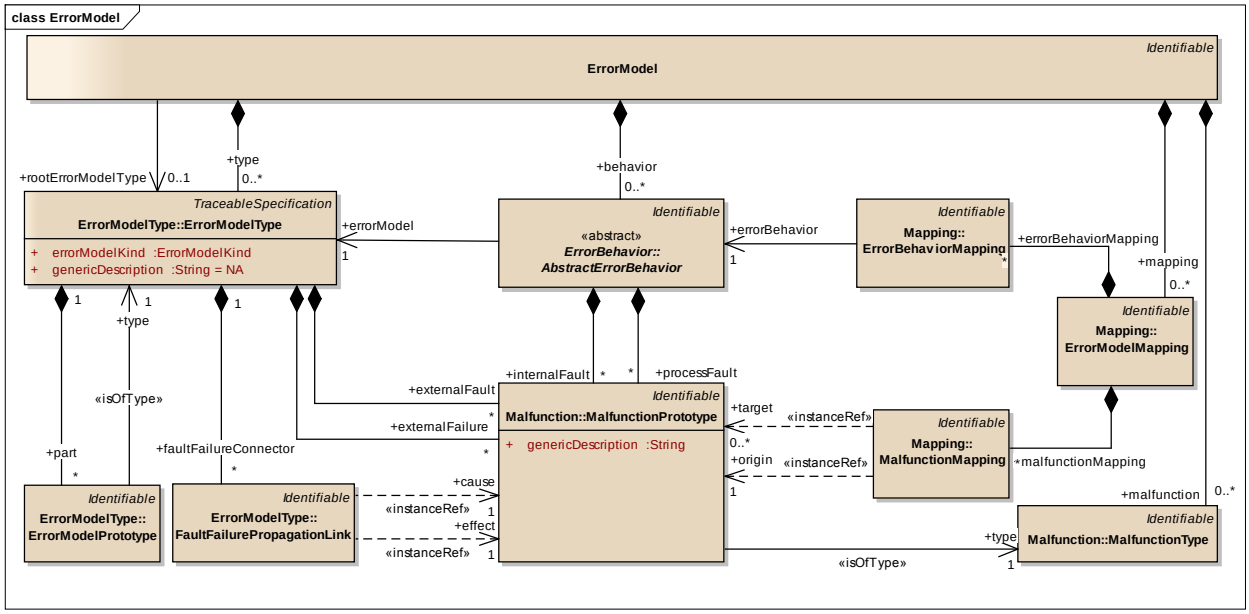


Figure 16: **ErrorModel** - *(Class diagram)*

Diagram Notes:

Class ErrorModel

Element Base Classes: **Identifiable**

Element Notes:

The error model is a container for all artifacts, which are needed to describe the error model of an architectural element: malfunctions, error types and error behaviors

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModel	SingleLevelSafetyExtension
Association Source -> Destination	ErrorModel	ErrorModelType
Aggregation Source -> Destination	ErrorModelMapping	ErrorModel
Aggregation Source -> Destination	AbstractErrorBehavior	ErrorModel
Generalization Source -> Destination	ErrorModel	Identifiable
Aggregation Source -> Destination	ErrorModelType	ErrorModel
Aggregation Source -> Destination	MalfunctionType	ErrorModel

Package ErrorBehavior

Package Notes:

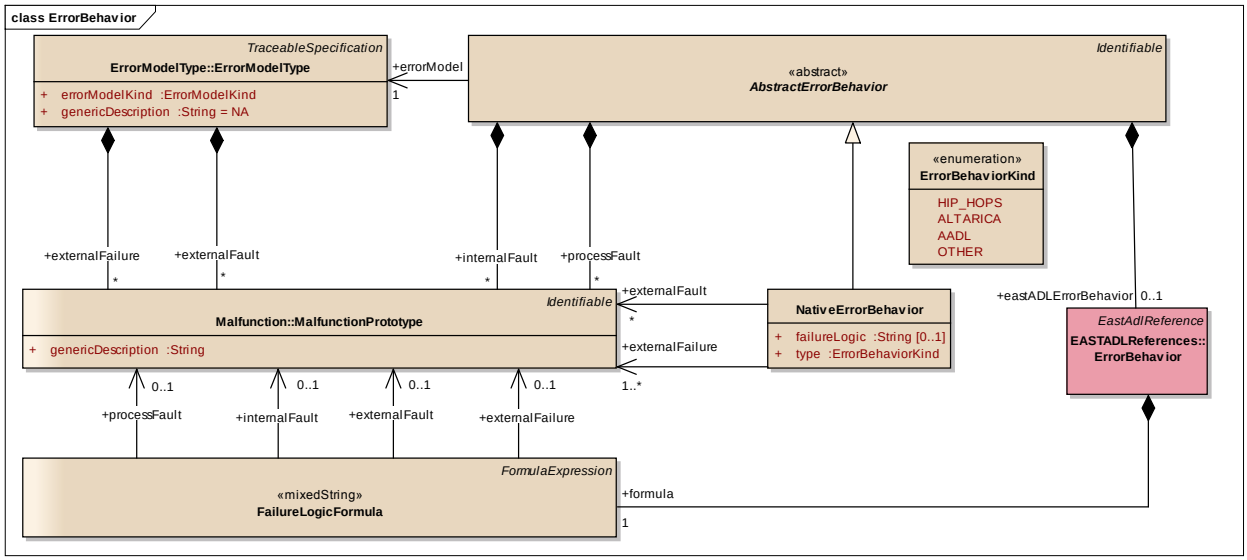


Figure 17: **ErrorBehavior** - *(Class diagram)*

Diagram Notes:

Diagram for ErrorBehavior.

Abstract AbstractErrorBehavior

Element Base Classes: **Identifiable**

Element Notes:

This class contains information about the error behavior independent of concrete behavior descriptions.

The AbstractErrorBehavior contains internalFaults, representing faults that are either propagated to externalFailures of the ErrorModelType or masked, according to the definition of its fault propagation.

A processFault represents a flaw introduced during design, and may lead to any of the failures represented by the ErrorModelType. A processFault therefore has a direct propagation to all externalFailures and cannot be masked.

Each error behavior description relates the occurrences of internal faults and incoming external faults to external failures. The faults and failures that the error behavior propagates to and from the target element are declared through the malfunction prototypes of the error model.

Semantics:

An error behavior describes the error propagation logic of its containing ErrorModelType.

The ErrorBehavior description represents the error propagation from internal faults or external faults to external failures. Faults are identified by the internalFault externalFault associations. The propagated external failures are identified by the externalFailure association.

Connections

Connector	Source	Target
Generalization Source -> Destination	EastADLErrorBehavior	AbstractErrorBehavior
Aggregation Source -> Destination	MalfunctionPrototype	AbstractErrorBehavior
Generalization Source -> Destination	NativeErrorBehavior	AbstractErrorBehavior
Association Source -> Destination	ErrorBehaviorMapping	AbstractErrorBehavior
Association Source -> Destination	AbstractErrorBehavior	ErrorModelType
Aggregation Source -> Destination	AbstractErrorBehavior	ErrorModel
Aggregation Source -> Destination	ErrorBehavior	AbstractErrorBehavior
Aggregation Source -> Destination	MalfunctionPrototype	AbstractErrorBehavior
Generalization Source -> Destination	AbstractErrorBehavior	Identifiable

Class EastADLErrorBehavior

Element Base Classes: **AbstractErrorBehavior**

Element Notes:

EASTADLErrorBehavior specifies a concrete failure logic description language, which describes the error propagation through the architectural element referenced by the containing ErrorModelType (e.g. function, hw component, sw component).

The failure logic is defined via a formula language called FailureLogicFormula (see "formula" association)

Connections

Connector	Source	Target
Generalization Source -> Destination	EastADLErrorBehavior	AbstractErrorBehavior

Enumeration ErrorBehaviorKind

Element Base Classes:

Element Notes:

The ErrorBehaviorKind metaclass represents an enumeration of literals describing various types of formalisms used for specifying error behavior.

Semantics:

ErrorBehaviorKind represents different formalisms for ErrorBehavior. The semantics is defined at each enumeration literal.

Extension:

Enumeration, no extension.

Attributes

Attribute	Notes	Default
HIP_HOPS	A specification of error behavior according to the external formalism HiP-HOPS.	
ALTARICA	A specification of error behavior according to the external formalism ALTARICA.	
AADL	A specification of error behavior according to the external formalism AADL.	
OTHER	A specification of error behavior according to other user defined formalism.	

Attribute	Notes	Default

Class FailureLogicFormula

Element Base Classes: **FormulaExpression**

Element Notes:

FailureLogicFormula is used to describe the error propagation through the architectural element associated with the containing ErrorModelType. The grammer of the FailureLogicFormula is defined in the respective specification document.

The failure logic formula language supports at least

- AND operator
- OR operator
- NOT operator
- local symbol or variable
- Typed for Boolean expression
- Allows stratified negation (failure itself shall not be used in its negated form), e.g. failure1 = fault2 or fault3 and not(failure1) expression is forbidden
- Permits to n:k gates. The error occurs if k out of n input faults occur.

Connections

Connector	Source	Target
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype
Generalization Source -> Destination	FailureLogicFormula	FormulaExpression
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype
Aggregation Source -> Destination	FailureLogicFormula	ErrorBehavior

Class NativeErrorBehavior

Element Base Classes: **AbstractErrorBehavior**

Element Notes:

NativeErrorBehavior represents the descriptions of failure logics or semantics that the architectural element associated by the ErrorModelType exhibits.

Semantics:

The NativeErrorBehavior is defined in the failureLogic string, either directly or as a url referencing an external specification.

The failureLogic can be based on different formalisms, depending on the analysis techniques and tools available. This is indicated by its type:ErrorBehaviorKind attribute. The failureLogic attribute contains the actual failure propagation logic.

Extension:

UML:Behavior

Connections

Connector	Source	Target
Generalization Source -> Destination	NativeErrorBehavior	AbstractErrorBehavior
Association Source -> Destination	NativeErrorBehavior	MalfunctionPrototype
Association Source -> Destination	NativeErrorBehavior	MalfunctionPrototype

Attributes

Attribute	Notes	Default
failureLogic String	The specification of error behavior based on an external formalism or the path to the file containing the external specification.	
type ErrorBehaviorKind	The type of formalism applied for the error behavior description.	

Package ErrorModelType

Package Notes:

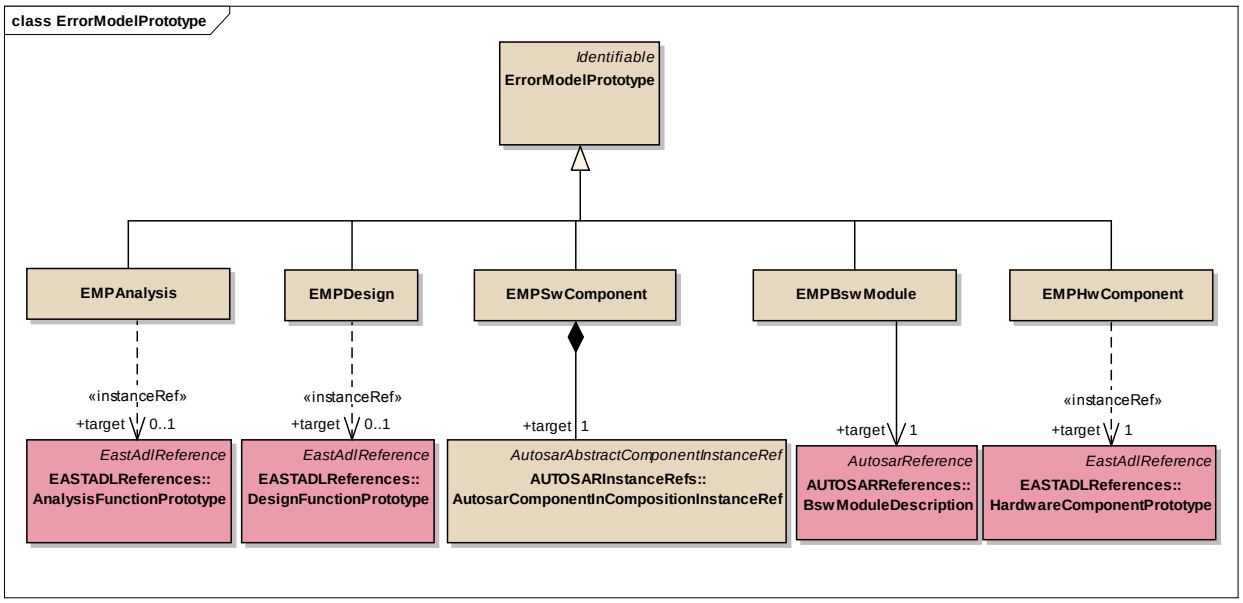


Figure 18: **ErrorModelPrototype** - *(Class diagram)*

Diagram Notes:

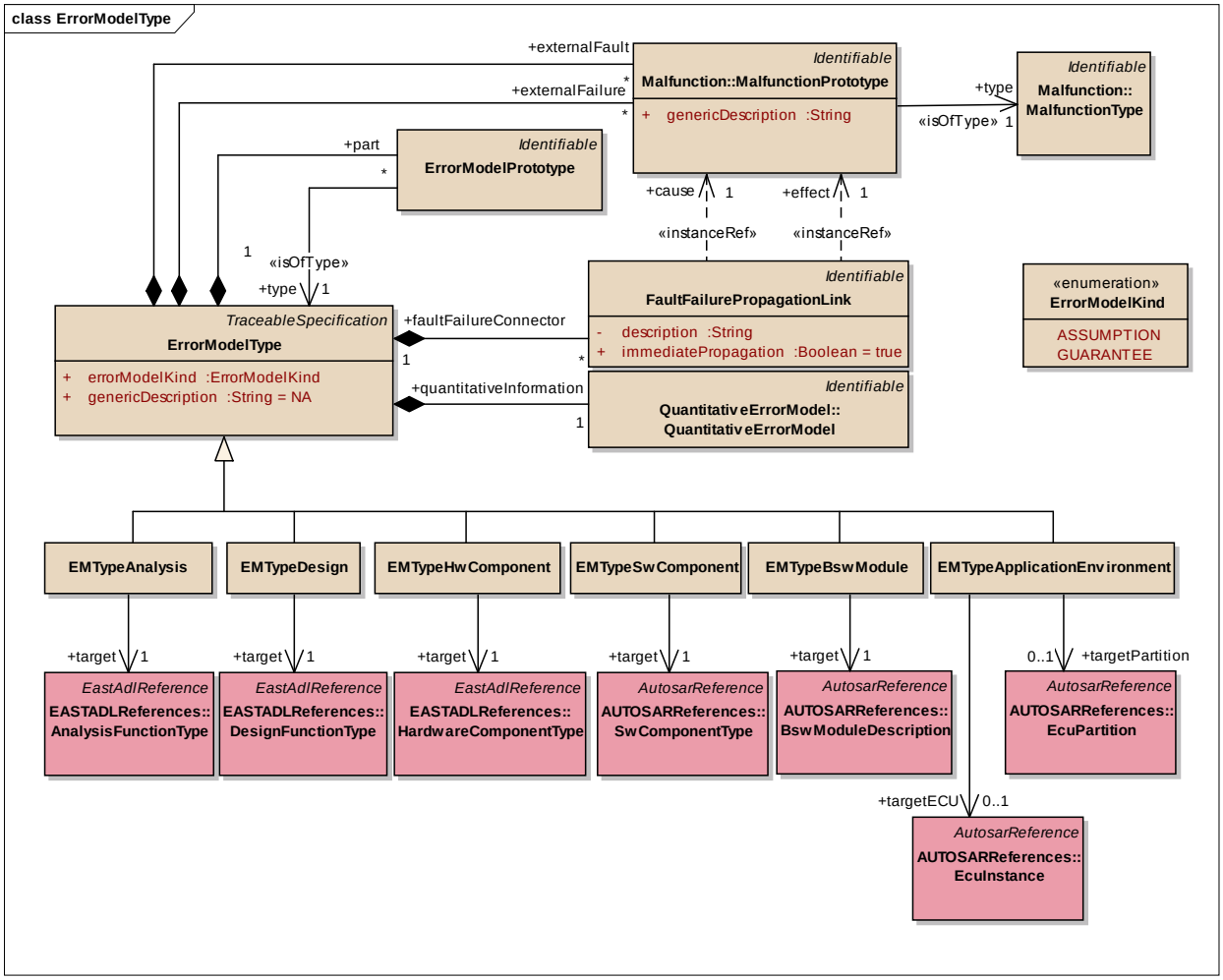


Figure 19: **ErrorModelType** - *(Class diagram)*

Diagram Notes:

The EAST-ADL metaclasses for defining the error model structure.

Class EMPAnalysis

Element Base Classes: **ErrorModelPrototype**

Element Notes:

Error model prototype specified for a concrete analysis function instance.

Connections

Connector	Source	Target
Aggregation	ErrorModelPrototype_analysisFunctionTarget	EMPAnalysis

Connector	Source	Target
Source -> Destination		
Dependency Source -> Destination	EMPAnalysis	AnalysisFunctionPrototype
Generalization Source -> Destination	EMPAnalysis	ErrorModelPrototype

Class EMPBswModule

Element Base Classes: **ErrorModelPrototype**

Element Notes:

Error model prototype specified for a concrete bsw software module.

Connections

Connector	Source	Target
Generalization Source -> Destination	EMPBswModule	ErrorModelPrototype
Association Source -> Destination	EMPBswModule	BswModuleDescription

Class EMPDesign

Element Base Classes: **ErrorModelPrototype**

Element Notes:

Error model prototype specified for a concrete design function instance.

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModelPrototype_designFunctionTarget	EMPDesign

Connector	Source	Target
Generalization Source -> Destination	EMPDesign	ErrorModelPrototype
Dependency Source -> Destination	EMPDesign	DesignFunctionPrototype

Class EMPHwComponent

Element Base Classes: **ErrorModelPrototype**

Element Notes:

Error model prototype specified for a concrete hardware component instance.

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModelPrototype_hwTarget	EMPHwComponent
Dependency Source -> Destination	EMPHwComponent	HardwareComponentPrototype
Generalization Source -> Destination	EMPHwComponent	ErrorModelPrototype

Class EMPReference

Element Base Classes:

Element Notes:

Class EMPSwComponent

Element Base Classes: **ErrorModelPrototype**

Element Notes:

Error model prototype specified for a concrete software component instance.

Connections

Connector	Source	Target
Generalization Source -> Destination	EMPSwComponent	ErrorModelPrototype
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	EMPSwComponent

Class EMTypeAnalysis

Element Base Classes: **ErrorModelType**

Element Notes:

Error model type specified for a concrete analysis function.

Connections

Connector	Source	Target
Generalization Source -> Destination	EMTypeAnalysis	ErrorModelType
Association Source -> Destination	EMTypeAnalysis	AnalysisFunctionType

Class EMTypeApplicationEnvironment

Element Base Classes: **ErrorModelType**

Element Notes:

The EMTypeApplicationEnvironment allows describing the error model for the runtime environment of application software on AUTOSAR implementation level. The application environment can be virtual, meaning that the modeled error model is an assumption of the environment the application software is

running on. In a real system, EMTypeApplicationEnvironment is associated with a concrete ECUInstance in the system and (optional) with a ECU partition within this ECU. In this case, the error model represents the description of the unintended behavior originated by this runtime environment.

Connections

Connector	Source	Target
Generalization Source -> Destination	EMTypeApplicationEnvironment	ErrorModelType
Association Source -> Destination	EMTypeApplicationEnvironment	EcuPartition
Association Source -> Destination	EMTypeApplicationEnvironment	EcuInstance

Class EMTypeBswModule

Element Base Classes: **ErrorModelType**

Element Notes:

Error model type specified for a concrete basic software module.

Connections

Connector	Source	Target
Generalization Source -> Destination	EMTypeBswModule	ErrorModelType
Association Source -> Destination	EMTypeBswModule	BswModuleDescription

Class EMTypeDesign

Element Base Classes: **ErrorModelType**

Element Notes:

Error model type specified for a concrete design function.

Connections

Connector	Source	Target
Generalization Source -> Destination	EMTypeDesign	ErrorModelType
Association Source -> Destination	EMTypeDesign	DesignFunctionType

Class EMTypeHwComponent

Element Base Classes: **ErrorModelType**

Element Notes:

Error model type specified for a concrete hardware component

Connections

Connector	Source	Target
Generalization Source -> Destination	EMTypeHwComponent	ErrorModelType
Association Source -> Destination	EMTypeHwComponent	HardwareComponentType

Class EMTypeSwComponent

Element Base Classes: **ErrorModelType**

Element Notes:

Error model type specified for a concrete software component

Connections

Connector	Source	Target
Association Source -> Destination	EMTypeSwComponent	SwComponentType

Connector	Source	Target
Generalization Source -> Destination	EMTypeSwComponent	ErrorModelType

Enumeration ErrorModelKind

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
ASSUMPTION		
GUARANTEE		

Class ErrorModelPrototype

Element Base Classes: **Identifiable**

Element Notes:

The ErrorModelPrototype is used to define hierarchical error models allowing additional detail or structure to the error model of a particular target. A hierarchal structure can also be defined when several ErrorModels are integrated to a larger ErrorModel representing a system integrated from several targets.

There are diffent subtypes of ErrorModelPrototype specified, allowing to add additional information describe the context of the ErrorModelProtoype.

Semantics:

An ErrorModelPrototype represents an occurrence of the ErrorModelType that types it.

Extension:

(See ADLFunctionPrototype)

Connections

Connector	Source	Target
Generalization Source -> Destination	ErrorModelPrototype	Identifiable
Association Source -> Destination	ErrorModelPrototype	ErrorModelType
Generalization Source -> Destination	EMPSwComponent	ErrorModelPrototype
Association Source -> Destination	ErrorModelInstanceRef	ErrorModelPrototype
Generalization Source -> Destination	EMPBswModule	ErrorModelPrototype
Association Source -> Destination	ErrorModelInstanceRef	ErrorModelPrototype
Generalization Source -> Destination	EMPHwComponent	ErrorModelPrototype
Generalization Source -> Destination	EMPDesign	ErrorModelPrototype
Association Source -> Destination	MalfunctionInstanceRef	ErrorModelPrototype
Aggregation Source -> Destination	ErrorModelType	ErrorModelPrototype
Generalization Source -> Destination	EMPAnalysis	ErrorModelPrototype

Class ErrorModelType

Element Base Classes: **TraceableSpecification**

Element Notes:

ErrorModelType and ErrorModelPrototype support the hierarchical composition of error models based on the type-prototype pattern also adopted for the nominal architecture composition. The purpose of the error models is to represent information relating to the anomalies of a nominal model element.

Independent of the different subtypes of ErrorModelType, this class describes the external faults affecting the element, external failures caused by the element and fault propagations within the nominal element.

ErrorModelType inherits the abstract metaclass TraceableSpecification, allowing the ErrorModelType to be referenced from its design context in a similar way as requirements, test cases and other specifications.

Constraints:

For An ErrorModelType without part, a respective error behavior shall be defined in the safety model.

Semantics:

The ErrorModelType represents a specification of the faults and fault propagations of its target element.

Both types and prototypes may be targets, and the following cases are relevant:

- One nominal type:

The ErrorModelType represents the identified nominal type wherever this nominal type is instantiated.

- Several nominal types:

The ErrorModelType represents the identified nominal types individually, i.e. the same error model applies to all nominal types and is reused.

- One nominal prototype:

The ErrorModelType represents the identified nominal prototype whenever its context, i.e. its top-level composition is instantiated.

- Several nominal prototypes with instanceref:

The ErrorModelType represents the identified set of nominal prototypes (together) whenever their context, i.e. their top-level composition, is instantiated.

The fault propagation of an errorModelType is defined by its contained parts, the ErrorModelPrototypes and their connections. In case an error behavior is defined for this error model type, the fault propagation information, the error behavior and the parts of the error model shall be consistent.

FaultFailurePropagationLinks define valid propagation paths in the ErrorModelType. In case the contained external faults and external failures reference nominal ports, the connectivity of the nominal model may serve as a pattern for connecting malfunction prototypes in the ErrorModelType.

Extension:

(see ADLTraceableSpecfication)

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModelType	FaultFailurePropagationLink
Generalization Source -> Destination	EMTypeApplicationEnvironment	ErrorModelType
Association Source -> Destination	ErrorModelPrototype	ErrorModelType
Generalization Source -> Destination	EMTypeBswModule	ErrorModelType
Generalization Source -> Destination	EMTypeAnalysis	ErrorModelType
Association Source -> Destination	ErrorModel	ErrorModelType
Association Source -> Destination	AbstractErrorBehavior	ErrorModelType
Generalization Source -> Destination	EMTypeHwComponent	ErrorModelType
Aggregation Source -> Destination	MalfunctionPrototype	ErrorModelType
Generalization Source -> Destination	ErrorModelType	TraceableSpecification
Aggregation Source -> Destination	MalfunctionPrototype	ErrorModelType
Aggregation Source -> Destination	QuantitativeErrorModel	ErrorModelType

Connector	Source	Target
Generalization Source -> Destination	EMTypeSwComponent	ErrorModelType
Aggregation Source -> Destination	ErrorModelType	ErrorModel
Aggregation Source -> Destination	ErrorModelType	ErrorModelPrototype
Generalization Source -> Destination	EMTypeDesign	ErrorModelType

Attributes

Attribute	Notes	Default
errorModelKind ErrorModelKind		
genericDescription String		NA

Class FaultFailurePropagationLink

Element Base Classes: **Identifiable**

Element Notes:

The FaultFailurePropagationLink metaclass represents the links for the propagations of faults/failures across system elements. In particular, it defines that one error model provides the faults/failures that another error model receives.

A fault/failure link can only be applied to compatible ports, either for fault/failure delegation within an error model or for fault/failure transmission across two error models. A FaultFailurePropagationLink can only connect fault/failures that have compatible types.

Constraints:

[1] Only compatible cause-effect pairs may be connected.

[2] Two fault/failure are compatible if the MalfunctionType of the cause represents a subset of the MalfunctionType set represented by the MalfunctionType of the effect.

Semantics:

The FaultFailurePropagationLink defines a Failure propagation path, from the cause on one error model to the effect of another error model.

Extension:

UML::Connector

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModelType	FaultFailurePropagationLink
Aggregation Source -> Destination	MalfunctionInstanceRef	FaultFailurePropagationLink
Aggregation Source -> Destination	FaultFailurePropagationLink	MalfunctionInstanceRef
Dependency Source -> Destination	FaultFailurePropagationLink	MalfunctionPrototype
Dependency Source -> Destination	FaultFailurePropagationLink	MalfunctionPrototype
Generalization Source -> Destination	FaultFailurePropagationLink	Identifiable

Attributes

Attribute	Notes	Default
description String		
immediatePropagation Boolean		true

Package QuantitativeErrorModel

Package Notes:

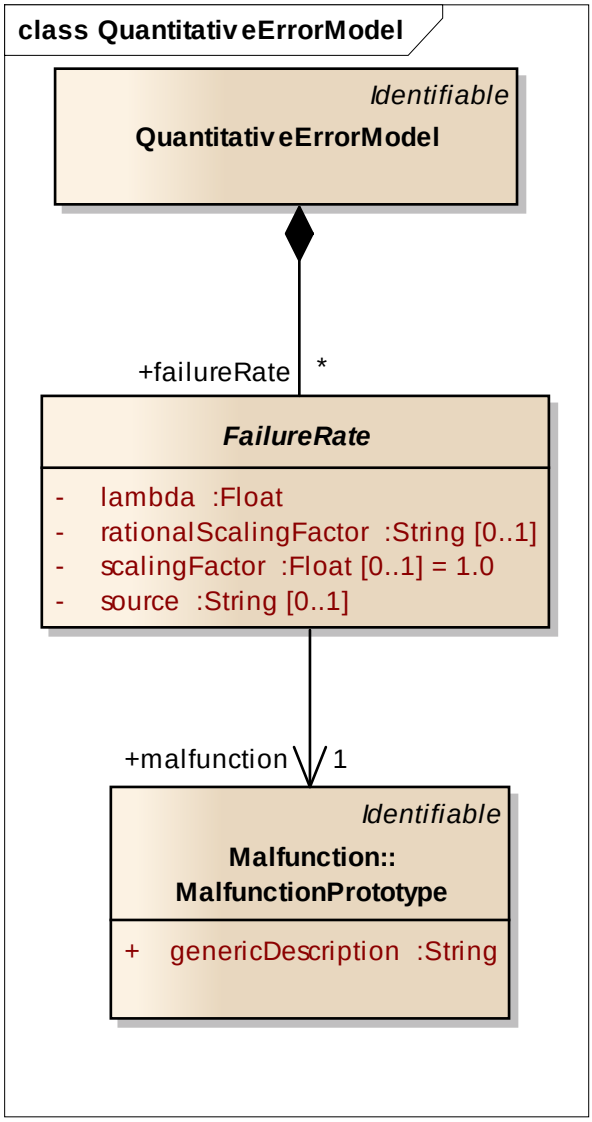


Figure 20: **QuantitativeErrorModel** - *(Class diagram)*

Diagram Notes:

Class FailureRate

Element Base Classes:

Element Notes:

This super class provides generic information required by all the concrete distributions gaussian, gumbel, frechet, exponential and uniform.

Connections

Connector	Source	Target
Aggregation Source -> Destination	FailureRate	QuantitativeErrorModel
Association Source -> Destination	FailureRate	MalfunctionPrototype

Attributes

Attribute	Notes	Default
lambda Float		
rationalScalingFactor String	The rationaleScalingFactor shall provide a rationale, if a scaling factor different to 1.0 is applied.	
scalingFactor Float	The scalingFactor allows potential scaling between different sources of failure rates as described in ISO Part 5 Annex F.	1.0
source String	FIT rate source shall documented according to possible source as described in ISO 26262 Part 5 8.4.3.	

Class QuantitativeErrorModel

Element Base Classes: **Identifiable**

Element Notes:

This class is the container for all quantitative information relevant for the error model associated with this class.

Connections

Connector	Source	Target
Generalization Source -> Destination	QuantitativeErrorModel	Identifiable
Aggregation Source -> Destination	FailureRate	QuantitativeErrorModel
Aggregation Source -> Destination	QuantitativeErrorModel	ErrorModelType

Package Malfunction

Package Notes:

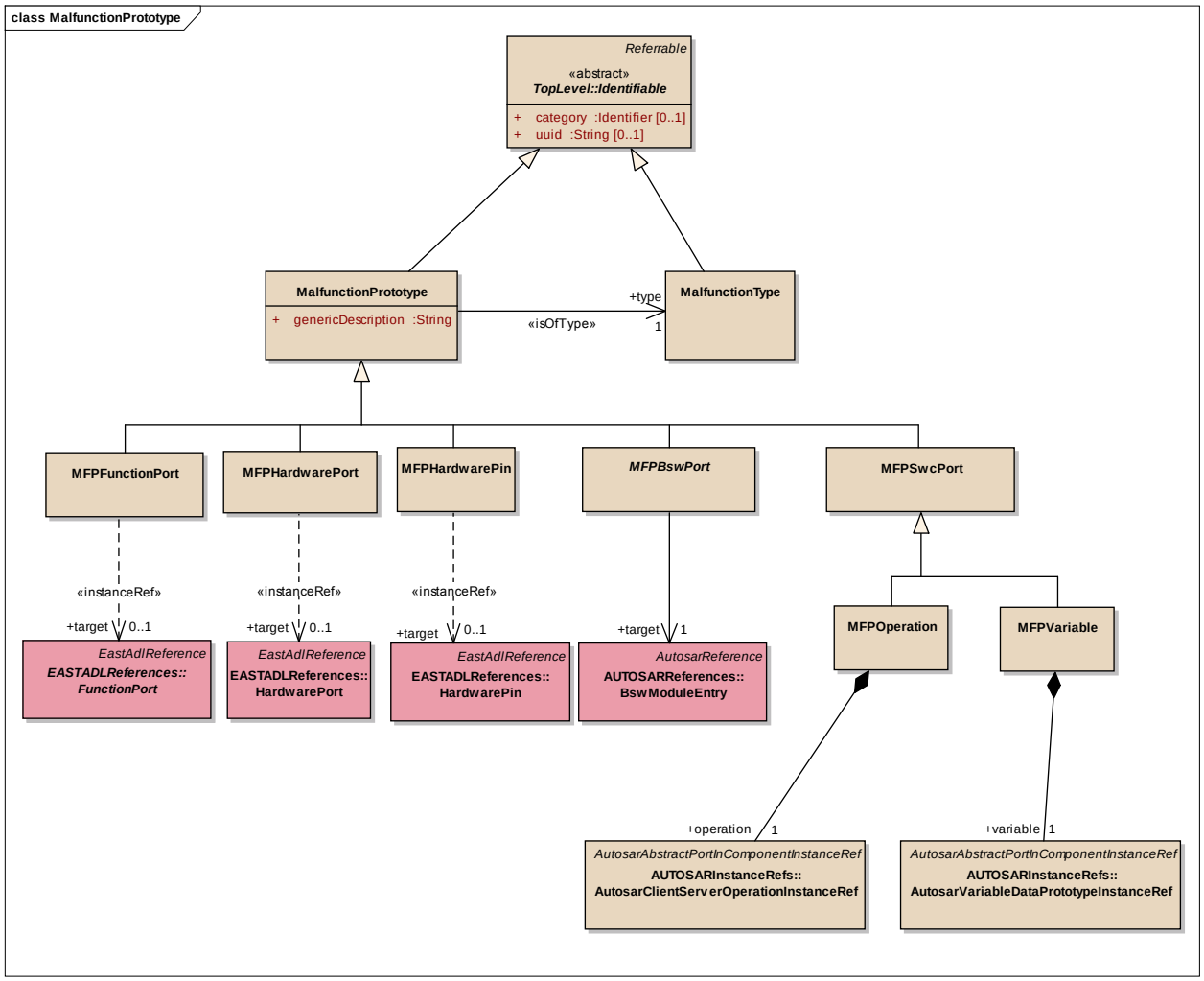


Figure 21: **MalfunctionPrototype** - *(Class diagram)*

Diagram Notes:

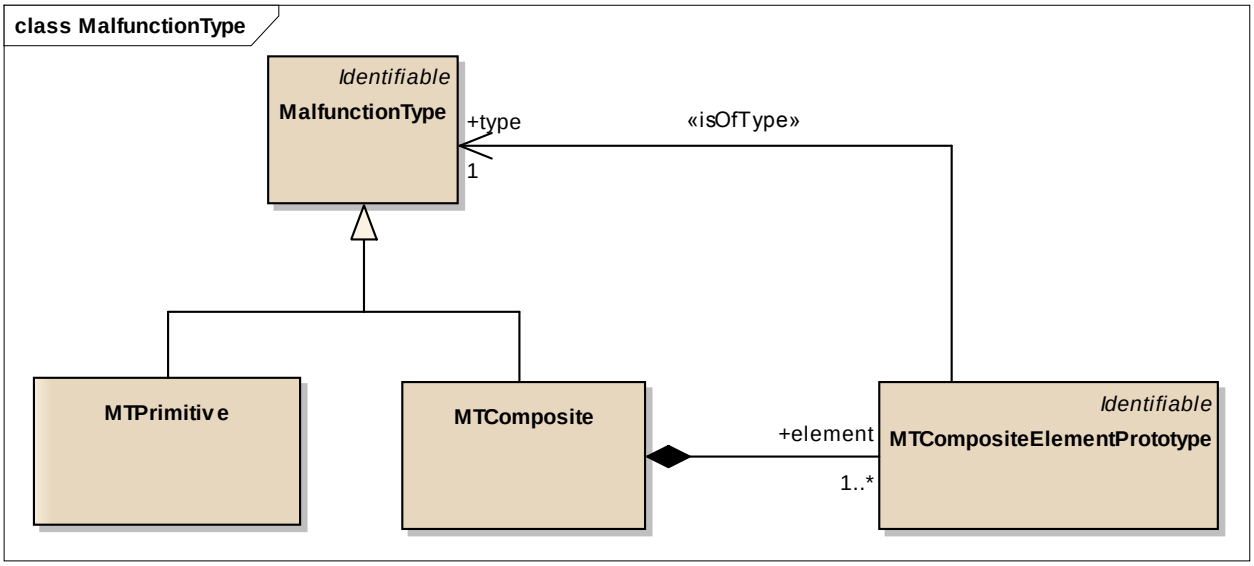


Figure 22: **MalfunctionType** - *(Class diagram)*

Diagram Notes:

Class MFPBswPort

Element Base Classes: **MalfunctionPrototype**

Element Notes:

The MalfunctionPrototype pointing to a basic software module entry

Connections

Connector	Source	Target
Generalization Source -> Destination	MFPBswPort	MalfunctionPrototype
Association Source -> Destination	MFPBswPort	BswModuleEntry

Class MFPFunctionPort

Element Base Classes: **MalfunctionPrototype**

Element Notes:

The MalfunctionPrototype pointing to a function port instance

Connections

Connector	Source	Target
Dependency Source -> Destination	MFPFunctionPort	FunctionPort
Generalization Source -> Destination	MFPFunctionPort	MalfunctionPrototype
Aggregation Source -> Destination	FaultFailurePort_functionTarget	MFPFunctionPort

Class MFPHardwarePin

Element Base Classes: **MalfunctionPrototype**

Element Notes:

The MalfunctionPrototype pointing to a HardwarPin instance

Connections

Connector	Source	Target
Aggregation Source -> Destination	FaultFailurePort_hwPinTarget	MFPHardwarePin
Generalization Source -> Destination	MFPHardwarePin	MalfunctionPrototype
Dependency Source -> Destination	MFPHardwarePin	HardwarePin

Class MFPHardwarePort

Element Base Classes: **MalfunctionPrototype**

Element Notes:

The MalfunctionPrototype pointing to a HardwarPort instance

Connections

Connector	Source	Target
Dependency Source -> Destination	MFPHardwarePort	HardwarePort
Aggregation Source -> Destination	FaultFailurePort_hwPortTarget	MFPHardwarePort
Generalization Source -> Destination	MFPHardwarePort	MalfunctionPrototype

Class MFPOperation

Element Base Classes: **MFPSwcPort**

Element Notes:

The MalfunctionPrototype pointing to an AUTOSAR operation instance

Connections

Connector	Source	Target
Generalization Source -> Destination	MFPOperation	MFPSwcPort
Aggregation Source -> Destination	AutosarClientServerOperationInstanceRef	MFPOperation

Class MFPSwcPort

Element Base Classes: **MalfunctionPrototype**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	MFPSwcPort	MalfunctionPrototype
Generalization Source -> Destination	MFPOperation	MFPSwcPort
Generalization Source -> Destination	MFPVariable	MFPSwcPort

Class MFPVariable

Element Base Classes: **MFPSwcPort**

Element Notes:

The MalfunctionPrototype pointing to an AUTOSAR variable instance

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	MFPVariable
Generalization Source -> Destination	MFPVariable	MFPSwcPort

Class MTComposite

Element Base Classes: **MalfunctionType**

Element Notes:

This composite malfunction type allows to define the different ways, how the malfunction becomes visible. As a typical example, a composite malfunction type could have the elements "commission" and "omission".

BrakeMalfunctionType (type: MTComposite):

- BrakePressureTooLow (type: MTPrimitive):

Semantics="brake pressure is below 20% of requested value"

- Omission (type: MTPrimitive):

Semantics="brake pressure is below 10% of maximal brake pressure"

- Commission (type: MTPrimitive):

Semantics="brake pressure exceeds requested value with more than 10% of maximal brake pressure".

Connections

Connector	Source	Target
Generalization Source -> Destination	MTComposite	MalfunctionType
Aggregation Source -> Destination	MTCompositeElementPrototype	MTComposite

Class MTCompositeElementPrototype

Element Base Classes: **Identifiable**

Element Notes:

This class is used the apply the composite pattern for the definition of MalfunctionTypes. It is contained by the composite MalfunctionTtype and refers to the type of the elements within the composite MalfunctionType.

Connections

Connector	Source	Target
Generalization Source -> Destination	MTCompositeElementPrototype	Identifiable
Aggregation Source -> Destination	MTCompositeElementPrototype	MTComposite

Connector	Source	Target
Association Source -> Destination	MTCompositeElementPrototype	MalfunctionType

Class MTPrimitive

Element Base Classes: **MalfunctionType**

Element Notes:

This class is used for the atomic definition of a malfunction type. The description field of the derived Identifiable class shall be used to describe the malfunction type.

Connections

Connector	Source	Target
Generalization Source -> Destination	MTPrimitive	MalfunctionType

Class MalfunctionPrototype

Element Base Classes: **Identifiable**

Element Notes:

A malfunction is a failure or unintended behavior of the item or element of the item that has the potential to propogate. The MalfunctionPrototype metaclass represents an error that may occur internally in an ErrorModel or be propagated to it, or a failure that is propagated out of an Error Model. The MalfunctionPrototype may represent different errors depending on its type (enumeration of generic description).

Semantics:

An malfunction prototype refers to a condition that deviates from expectations based on requirements specifications, design documents, user documents, standards, etc., or from someone's perceptions or experiences (ISO26262). The set of available faults or failures represented by the MalfunctionPrototype is defined by its type, typically an enumeration type like {omission, commission}. It is an abstract class further specialized with metaclasses for different types of fault/failure.

Extension:

(UML::Part)

Connections

Connector	Source	Target
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype
Aggregation Source -> Destination	MalfunctionPrototype	AbstractErrorBehavior
Dependency Source -> Destination	MalfunctionMapping	MalfunctionPrototype
Generalization Source -> Destination	MalfunctionPrototype	Identifiable
Association Source -> Destination	RiskDescription	MalfunctionPrototype
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype
Generalization Source -> Destination	MFPFunctionPort	MalfunctionPrototype
Generalization Source -> Destination	MFPSwcPort	MalfunctionPrototype
Generalization Source -> Destination	MFPHardwarePin	MalfunctionPrototype
Aggregation Source -> Destination	MalfunctionPrototype	Item
Aggregation Source -> Destination	MalfunctionPrototype	ErrorModelType
Dependency Source -> Destination	HWPartFailureAnalysis	MalfunctionPrototype
Generalization Source -> Destination	MFPBswPort	MalfunctionPrototype
Aggregation Source -> Destination	MalfunctionPrototype	ErrorModelType
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype

Connector	Source	Target
Association Source -> Destination	MalfunctionInstanceRef	MalfunctionPrototype
Aggregation Source -> Destination	MalfunctionPrototype	AbstractErrorBehavior
Association Source -> Destination	NativeErrorBehavior	MalfunctionPrototype
Association Source -> Destination	MalfunctionPrototype	MalfunctionType
Association Source -> Destination	FailureRate	MalfunctionPrototype
Dependency Source -> Destination	HardwareFailureAnalysis	MalfunctionPrototype
Association Source -> Destination	NativeErrorBehavior	MalfunctionPrototype
Association Source -> Destination	FailureLogicFormula	MalfunctionPrototype
Association Source -> Destination	HazardousEvent	MalfunctionPrototype
Dependency Source -> Destination	FaultFailurePropagationLink	MalfunctionPrototype
Dependency Source -> Destination	FaultFailurePropagationLink	MalfunctionPrototype
Dependency Source -> Destination	MalfunctionMapping	MalfunctionPrototype
Generalization Source -> Destination	MFPHardwarePort	MalfunctionPrototype

Attributes

Attribute	Notes	Default
genericDescription String	A description of the MalfunctionPrototype	

Class MalfunctionType

Element Base Classes: **Identifiable**

Element Notes:

A MalfunctionType describes how a malfunction becomes visible. Currently, it can either be a primitive description of a malfunction or defines as hierarchical composition of different "appearance" possibilities.

Connections

Connector	Source	Target
Generalization Source -> Destination	MalfunctionType	Identifiable
Aggregation Source -> Destination	MalfunctionType	MultiLevelSafetyExtension
Generalization Source -> Destination	MTComposite	MalfunctionType
Generalization Source -> Destination	MTPrimitive	MalfunctionType
Association Source -> Destination	MalfunctionPrototype	MalfunctionType
Association Source -> Destination	MTCompositeElementPrototype	MalfunctionType
Generalization Source -> Destination	HWFailureMode	MalfunctionType
Aggregation Source -> Destination	MalfunctionType	ErrorModel

Package Mapping

Package Notes:

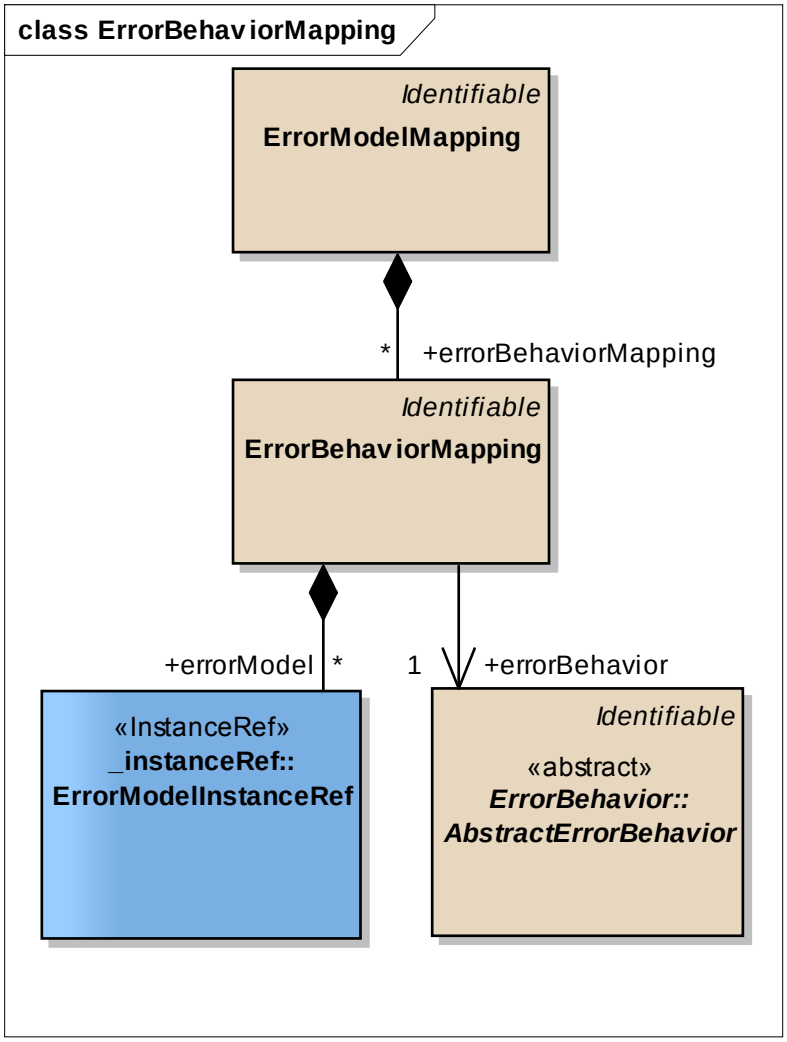


Figure 23: **ErrorBehaviorMapping** - *(Class diagram)*

Diagram Notes:

Class ErrorBehaviorMapping

Element Base Classes: **Identifiable**

Element Notes:

The class ErrorBehaviorMapping allows to map an error behavior to a concrete instance of an ErrorModelType within the ErrorModel. For each instance reference referred by the association named "errorModel", the following rule must be considered:

- The ErrorModelType of the first context element in the instanceRef of type ErrorModelInstanceRef must match with the attribute "rootErrorModelType" of the ErrorModel, where this ErrorBehaviorMapping is contained

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorBehaviorMapping	ErrorModelMapping
Association Source -> Destination	ErrorBehaviorMapping	AbstractErrorBehavior
Generalization Source -> Destination	ErrorBehaviorMapping	Identifiable
Aggregation Source -> Destination	ErrorModelInstanceRef	ErrorBehaviorMapping

Class ErrorModelMapping

Element Base Classes: **Identifiable**

Element Notes:

This class contains mapping information for error model.

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorBehaviorMapping	ErrorModelMapping
Aggregation Source -> Destination	ErrorModelMapping	ErrorModel
Generalization Source -> Destination	ErrorModelMapping	Identifiable
Aggregation Source -> Destination	MalfunctionMapping	ErrorModelMapping

Class MalfunctionMapping

Element Base Classes: **Identifiable**

Element Notes:

Via the class MalfunctionMapping it is possible to map malfunctions of one abstraction level (e.g. EAST ADL implementation level) to another level of abstraction (e.g. EAST ADL analysis level). This way the correlation between different levels of abstraction can be made explicit.

The MalfunctionMapping shall be attached to the "lower" level of abstraction. Example:

- a malfunction defined for a software component (implementation level) shall be related with a malfunction defined on design level
- in this case, (at least) two instances of SafetyExtensions (TechnicalSafetyExtension, AutosarSystemSafetyExtension) exist. In each of them an ErrorModelType is defined containing the before mentioned malfunctions
- in the AutosarSystemSafetyExtension, we define an MalfunctionMapping and relate the software malfunction (in the role of "origin") to the malfunction defined within the TechnicalSafetyExtension (in the role "target")

Thus, the following rules shall be applied:

- the MalfunctionPrototype referenced via the "origin" relationship shall be defined within the same SafetyExtension as an instance of this class
- the MalfunctionPrototype references via the "target" relationship shall be defined within a SafetyExtension which corresponds to a higher level of abstraction

Connections

Connector	Source	Target
Dependency Source -> Destination	MalfunctionMapping	MalfunctionPrototype
Generalization Source -> Destination	MalfunctionMapping	Identifiable
Aggregation Source -> Destination	MalfunctionInstanceRef	MalfunctionMapping
Aggregation Source -> Destination	MalfunctionInstanceRef	MalfunctionMapping
Dependency Source -> Destination	MalfunctionMapping	MalfunctionPrototype
Aggregation Source -> Destination	MalfunctionMapping	ErrorModelMapping

Package _instanceRef

Package Notes:

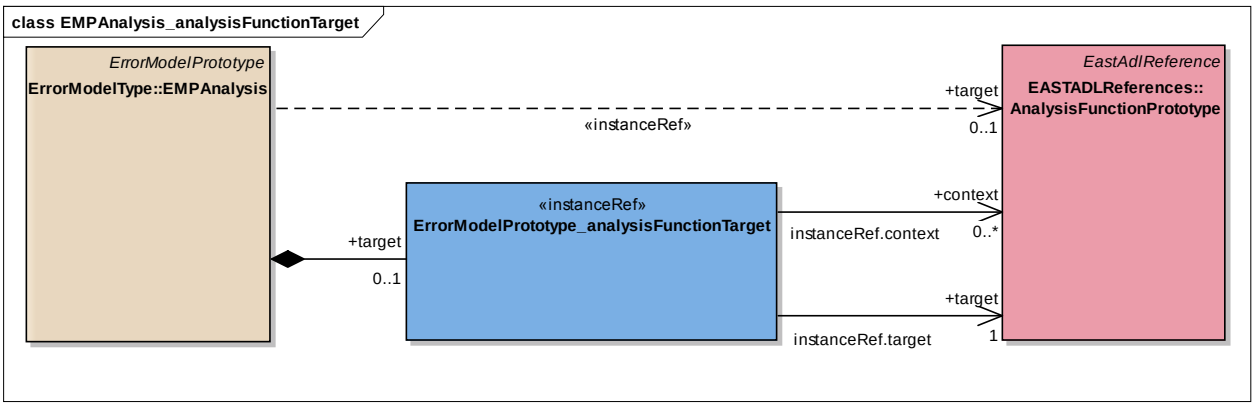


Figure 24: **EMPAnalysis_analysisFunctionTarget** - *(Class diagram)*

Diagram Notes:

Diagram for ErrorModelPrototype.

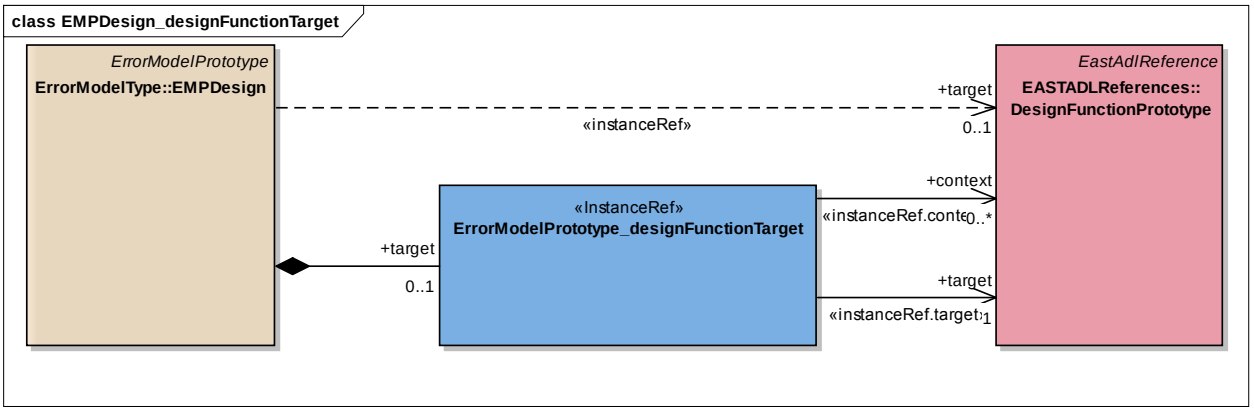


Figure 25: **EMPDesign_designFunctionTarget** - *(Class diagram)*

Diagram Notes:

Diagram for ErrorModelPrototype.

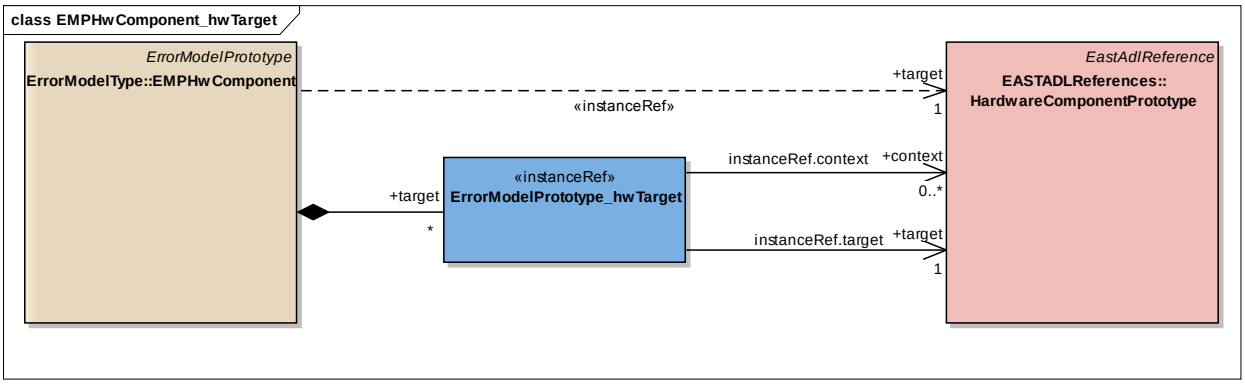


Figure 26: **EMPHwComponent_hwTarget** - *(Class diagram)*

Diagram Notes:

Figure 27: **ErrorModelInstance** - *(Class diagram)*

Diagram Notes:

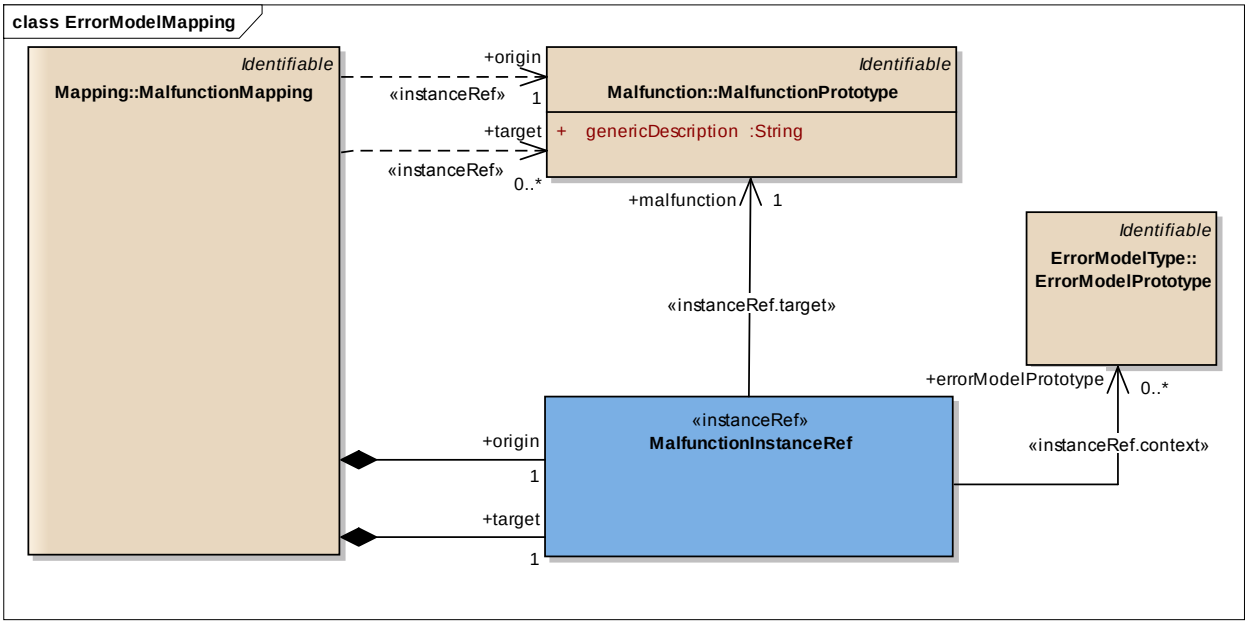


Figure 28: **ErrorModelMapping** - *(Class diagram)*

Diagram Notes:

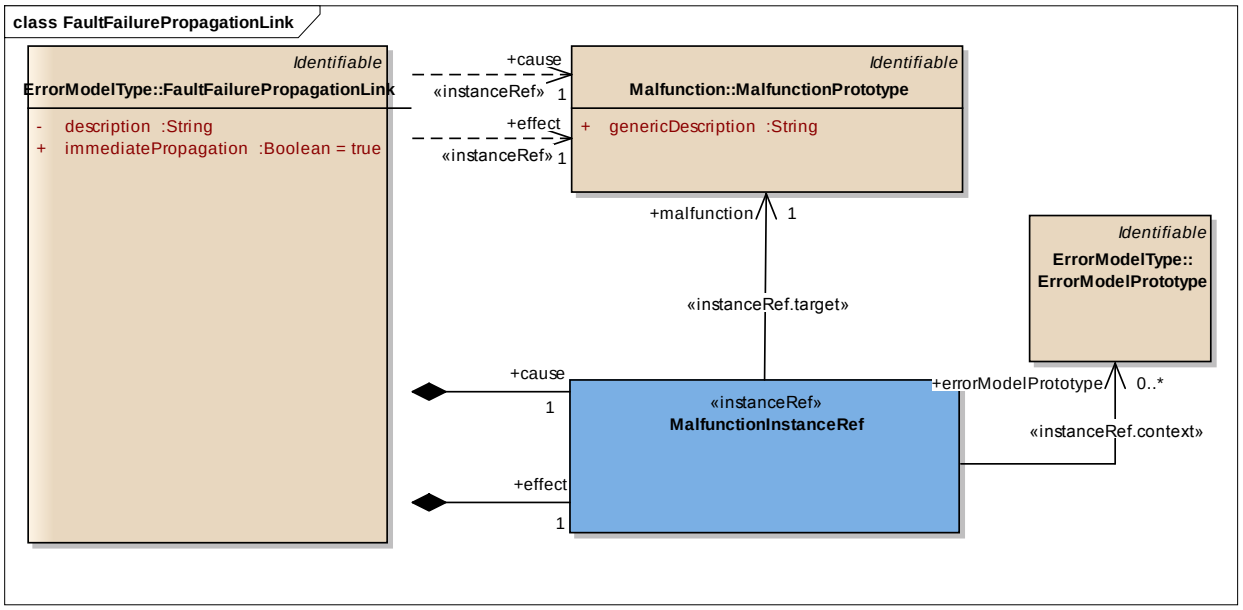


Figure 29: **FaultFailurePropagationLink** - *(Class diagram)*

Diagram Notes:

Diagram for FaultFailurePropagationLink.

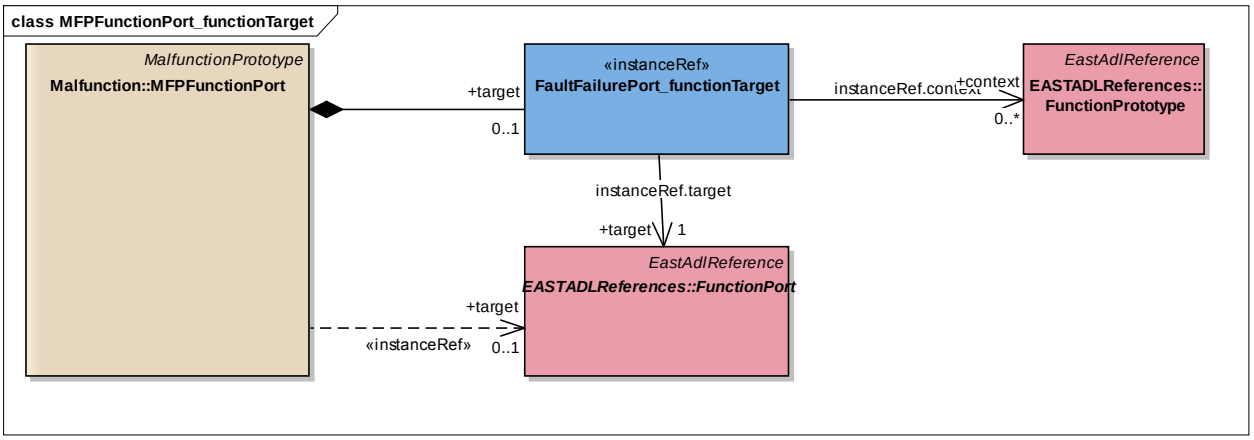


Figure 30: **MFPFunctionPort_functionTarget** - *(Class diagram)*

Diagram Notes:

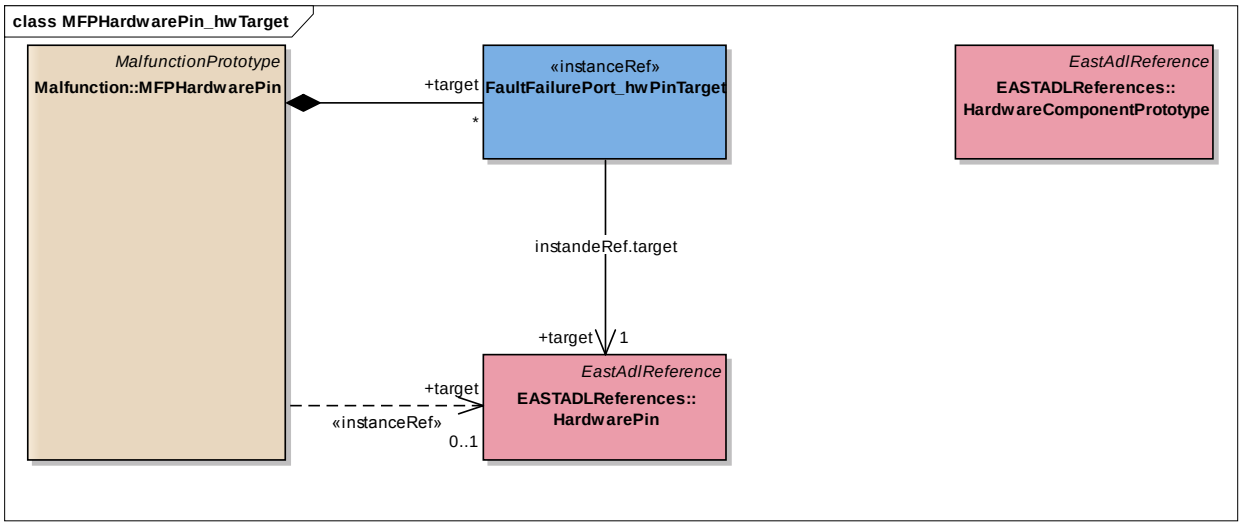


Figure 31: **MFPHardwarePin_hwTarget** - *(Class diagram)*

Diagram Notes:

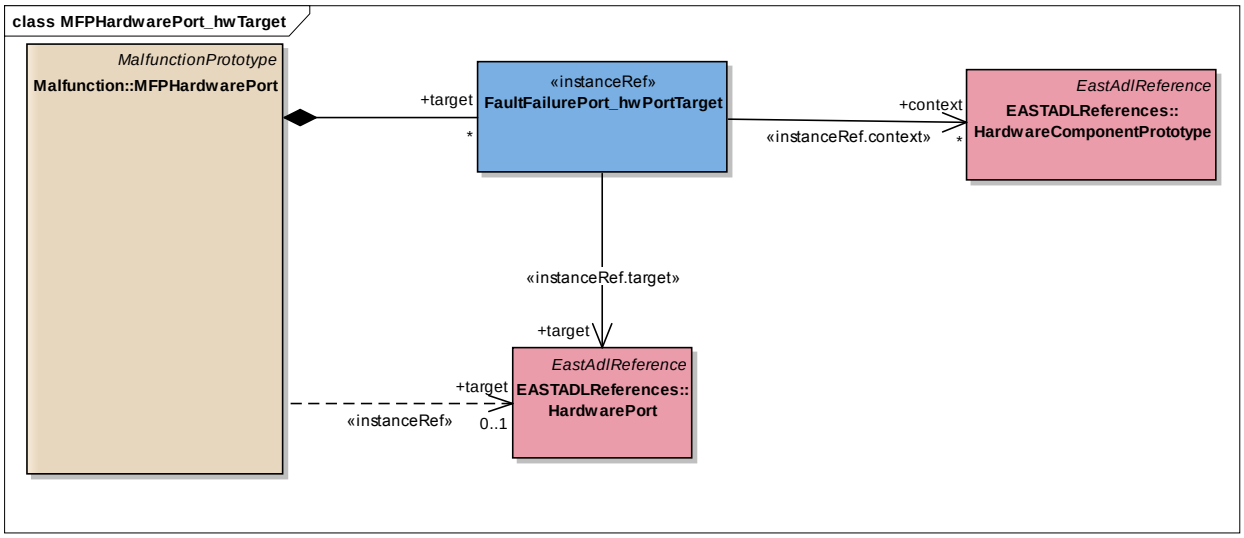


Figure 32: **MFPHardwarePort_hwTarget** - *(Class diagram)*

Diagram Notes:

Class ErrorModelInstanceRef

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	ErrorModelInstanceRef	ErrorModelPrototype
Association Source -> Destination	ErrorModelInstanceRef	ErrorModelPrototype
Aggregation Source -> Destination	ErrorModelInstanceRef	ErrorBehaviorMapping

Class ErrorModelPrototype_analysisFunctionTarget

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModelPrototype_analysisFunctionTarget	EMPAnalysis
Association instanceRef.context Source -> Destination	ErrorModelPrototype_analysisFunctionTarget	AnalysisFunctionPrototype
Association instanceRef.target Source -> Destination	ErrorModelPrototype_analysisFunctionTarget	AnalysisFunctionPrototype

Class ErrorModelPrototype_designFunctionTarget

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	ErrorModelPrototype_designFunctionTarget	DesignFunctionPrototype
Aggregation Source -> Destination	ErrorModelPrototype_designFunctionTarget	EMPDesign
Association Source -> Destination	ErrorModelPrototype_designFunctionTarget	DesignFunctionPrototype

Class ErrorModelPrototype_hwTarget

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	ErrorModelPrototype_hwTarget	EMPHwComponent
Association instanceRef.context Source -> Destination	ErrorModelPrototype_hwTarget	HardwareComponentPrototype
Association instanceRef.target Source -> Destination	ErrorModelPrototype_hwTarget	HardwareComponentPrototype

Class FaultFailurePort_functionTarget

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	FaultFailurePort_functionTarget	MFPFunctionPort
Association instanceRef.context Source -> Destination	FaultFailurePort_functionTarget	FunctionPrototype
Association instanceRef.target Source -> Destination	FaultFailurePort_functionTarget	FunctionPort

Class FaultFailurePort_hwPinTarget

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	FaultFailurePort_hwPinTarget	MFPHardwarePin
Association instandeRef.target Source -> Destination	FaultFailurePort_hwPinTarget	HardwarePin

Class FaultFailurePort_hwPortTarget

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	FaultFailurePort_hwPortTarget	HardwareComponentPrototype
Aggregation Source -> Destination	FaultFailurePort_hwPortTarget	MFPHardwarePort
Association Source -> Destination	FaultFailurePort_hwPortTarget	HardwarePort

Class MalfunctionInstanceRef

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	MalfunctionInstanceRef	HWPartFailureAnalysis
Aggregation Source -> Destination	MalfunctionInstanceRef	FaultFailurePropagationLink
Aggregation Source -> Destination	FaultFailurePropagationLink	MalfunctionInstanceRef
Aggregation Source -> Destination	MalfunctionInstanceRef	Tactic
Aggregation Source -> Destination	MalfunctionInstanceRef	MalfunctionMapping
Association Source -> Destination	MalfunctionInstanceRef	MalfunctionPrototype
Aggregation Source -> Destination	MalfunctionInstanceRef	HardwareFailureAnalysis
Aggregation Source -> Destination	MalfunctionInstanceRef	MalfunctionMapping
Association Source -> Destination	MalfunctionInstanceRef	ErrorModelPrototype

Package Hardware

Package Notes:

This package describes the top-level package for all meta model extension regarding hardware, developed in WT 3.2.2.

Package FailureFormula

Package Notes:

This sub-package contains all equations necessary for the evaluation of the hardware architecture.

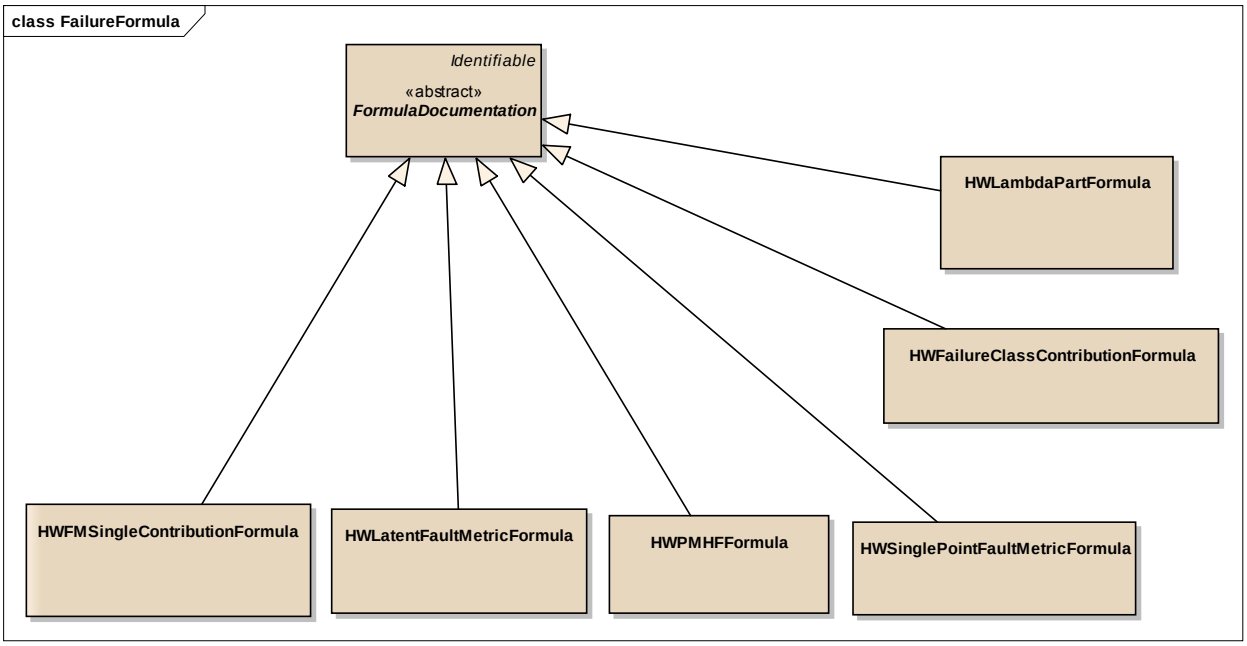


Figure 33: **FailureFormula** - *(Class diagram)*

Diagram Notes:

This diagram shows all formula expressions required for the evaluation of the hardware architecture, all derived from the class AtpFormulaExpressionString.

AtpFormulaExpressionString is derived from AUTOSAR AtpMixedString used to describe calculation formula.

Abstract FormulaDocumentation

Element Base Classes: **Identifiable**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	HWPMHFFormula	FormulaDocumentation

Connector	Source	Target
Generalization Source -> Destination	HWFailureClassContributionFormula	FormulaDocumentation
Generalization Source -> Destination	HWLambdaPartFormula	FormulaDocumentation
Generalization Source -> Destination	HWFMSingleContributionFormula	FormulaDocumentation
Generalization Source -> Destination	HWLatentFaultMetricFormula	FormulaDocumentation
Generalization Source -> Destination	FormulaDocumentation	Identifiable
Generalization Source -> Destination	HWSinglePointFaultMetricFormula	FormulaDocumentation

Class HWFMSingleContributionFormula

Element Base Classes: **FormulaDocumentation**

Element Notes:

This class describes the individual contribution of an HWFailureMode of a HWComponent to ResidualFault, SinglePointFault or Multiple Fault Latent (in FIT). It is assumed that the HWFailureMode lead to the top level malfunction (link to violation of a SafetyGoal) given by the relation to HWFault connected to a malfunction.

The formula expression shall be for each FailureMode of a safety-related HwComponent (part of the item).

SinglePointFault represents a first order fault, while DualPointFault multiple order fault (limited to 2)

The below calculation for lambdaMultiplePointFaultLatent is mostly relevant for HW metrics calculation and PMHF simplified calculation as only SafetyMechanism coverage are considered. It shall not be used for FTA based quantification as multiple point should consider the AND of the combination of the 2 failures.

lambdaSafetyComponent = Value(HWFailureRate)

SafetyComponentName = HardwareComponent Class name // to allow detect multiple counting of lambdaSafetyComponent

If (HWFault == SafeFault)

lambdaSafeFault(HWFMSingleContribution) = [Value(HWFailureRate)*failureRateDistribution(HWFailureMode)]

Else

lambdaSafeFault(HWFMSingleContribution) = 0

Endif

If (HWFault == SinglePointFault)

lambdaSinglePointFault(HWFMSingleContribution) = [Value(HWFailureRate)*failureRateDistribution(HWFailureMode)]

Else lambdaSinglePointFault(HWFMSingleContribution) = 0

Endif

If (HWFault == DualPointFault)

If (HWSafetyMechanism covers the FailureMode) //residual Fault as HWFailureMode.HWSafetyMechanism != null

lambdaResidualFault(HWFMSingleContribution) = [Value(HWFailureRate)*failureRateDistribution(HWFailureMode)] * [1 - hwDiagnosisCoverageRF(HWSafetyMechanism)/100]

lambdaMultiplePointFaultLatentM(HWFMSingleContribution) = [Value(HWFailureRate) * failureRateDistribution(HWFailureMode) * hwDiagnosisCoverageRF(HWSafetyMechanism)] * [(1 - hwDiagnosisCoverageLF(HWSafetyMechanism)/100)]

Else // assume 0% of efficiency for MPL,L metrics (from order 2)

lambdaResidualFault(HWFMSingleContribution) = 0

lambdaMultiplePointFaultLatent(HWFMSingleContribution) = [Value(HWFailureRate)*failureRateDistribution(HWFailureMode)]

Endif

Notes that value(HWFailureRate) and failureRateDistribution(HWFailureMode) are applied on the calculated value extracted from electronic design level to perform the final calculation and verification of the architectural hardware metrics and probabilistic evaluation of violation of the safety goal. The selection between allocated and calculated value is a tool feature that allow first a calculation for estimation based on allocation field of failure rate and distribution, and then verification based on HWComponentQuantifiedFMFromPartHWPart as extract from Part Analysis gives directly the lambda of the HWFailure Mode as HWFailureRate*FailureRateDistribution thanks to the relation to HWFault.

Connections

Connector	Source	Target

Connector	Source	Target
Association Source -> Destination	HWFMSingleContributionFormula	HWFailureMode
Aggregation Source -> Destination	HWFMSingleContributionFormula	HWFMSingleContribution
Association Source -> Destination	HWFMSingleContributionFormula	HWFault
Association Source -> Destination	HWFMSingleContributionFormula	HWFailureRate
Generalization Source -> Destination	HWFMSingleContributionFormula	FormulaDocumentation

Class HWFailureClassContributionFormula

Element Base Classes: **FormulaDocumentation**

Element Notes:

This class describes the calculation of the diagnostic coverage of a HWElement (from HWComponent) for the Failure Rate Class method (in %) as ratio of all fault coverage of the HW Component (safe fault, single-point fault and residual fault) and the calculation of the element FailureRateClass defined by its failure rate.

The formula expression shall be calculated for each FailureMode of a safety-related HwComponent (part of the item).

HW Element Failure Rate Class = Failure Class (safety-related failure rate component)

HW Element Residual Diagnostic Coverage = 100% - total (single point faults failure rate + residual faults failure rate) /safety related failure rate component

HW Element Latent Diagnostic Coverage = 100% - total(multiple fault latent) / ((safety related failure rate component) - total (single point faults failure rate + residual faults failure rate))

The formula expression shall be for each the top level malfunction (link to violation of the SafetyGoal).

hwElementResidualDiagnosisCoverage

HWElementFailureRateClass(hwElementFailureClass) = HWValuesFailureRateClassEnum(LambdaSafetyComponent)

HWElementFailureRateClass(hwElementResidualDiagnosisCoverage) = { 1 - (Sum (lambdaSinglePointFault(HWFMSingleContribution) + lambdaResidualFault(HWFMSingleContribution)) / LambdaSafetyComponent } * 100

HWElementFailureRateClass(hwElementLatentDiagnosisCoverage) = { 1 - Sum (lambdaMultipleFaultLatent(HWFMSingleContribution) / LambdaSafetyComponent } * 100

Note that Value(hwElementDiagnosisCoverage) is applied on estimatedValue from electronic design level to perform the final calculation and verification of the individual HWElement FailureRateClass and ElementDiagnosisCoverage. The selection between calculated and estimated value is a tool feature that allow first a calculation for estimation based on allocation field of failure rate. Only safety-related component are considered and LambdaSafetyComponent is only counted once for a HWElement (identical safetyComponentClassName).

Connections

Connector	Source	Target
Generalization Source -> Destination	HWFailureClassContributionFormula	FormulaDocumentation
Aggregation Source -> Destination	HWFailureClassContributionFormula	HWElementFailureRateClass
Association Source -> Destination	HWFailureClassContributionFormula	HWFMSingleContribution

Class HWLambdaPartFormula

Element Base Classes: **FormulaDocumentation**

Element Notes:

This class describes the lambda failure rate contribution of all HWPartFailureMode of HardwarePart to a dedicated HWFailureMode of a HWComponent.

The formula expression shall be for each HWFailureMode of a Safety Related HWComponent (related as parts of the Item) expressed from the different safety-related AUTOSAR HW Element (part of the item).

lambdaFailureMode = function all HWPartFailureMode [Value(HWPartFailureRate) * FailureRateDistribution(HWPartFailureMode), AutosarHWelement)

Connections

Connector	Source	Target
Association Source -> Destination	HWLambdaPartFormula	HWPartFailureRate
Generalization Source -> Destination	HWLambdaPartFormula	FormulaDocumentation
Aggregation Source -> Destination	HWLambdaPartFormula	HWComponentQuantifiedFMFromPart
Association Source -> Destination	HWLambdaPartFormula	HWPartFailureMode

Class HWLatentFaultMetricFormula

Element Base Classes: **FormulaDocumentation**

Element Notes:

This class describes the latent fault metric (in %) as ratio of impact of latent faults for a top level malfunction (link to violation of a SafetyGoal) .

Latent metric = 100% - total (multiple-point faults latent failure rate) /(total (safety-related HWComponent failure rate) - total (single-point faults failure rate + residual faults failure rate))

The formula expression shall be for each SafetyGoal:

Value(MultipleLatentFaultMetric) = { 1 - [Sum (lambdaMultipleFaultLatent(FMSingleContribution) / [Sum(LambdaSafetyComponent) - Sum (lambdaSinglePointFault(FMSingleContribution) + lambdaResidualFault(FMSingleContribution)]] } * 100

Value(MutiplePointFaultMteric) is applied on estimatedValue from electronic design level for final calculation and verification of the final latent fault metric. The selection between calculated and estimated value is a tool feature that allow first a calculation for estimation based on allocation field of failure rate and distribution. Only safety-related HWComponent are considered.

Sum(LambdaSafetyComponent) is only counted once for a HWElement (identical safetyComponentClassName).

Connections

Connector	Source	Target
Association Source -> Destination	HWLatentFaultMetricFormula	HWFMSingleContribution
Generalization Source -> Destination	HWLatentFaultMetricFormula	FormulaDocumentation
Aggregation Source -> Destination	HWLatentFaultMetricFormula	HWLatentFaultMetric

Class HWPMHFFormula

Element Base Classes: **FormulaDocumentation**

Element Notes:

This class describes the individual PMHF (in FIT) as probabilistic evaluation of violation of a top level malfunction (link to violation of a SafetyGoal).

PMHF = single point faults failure rate + residual faults failure rate + (total safety related faults failure rate / 10^{-9} * delta) * latent multiple point faults failure rate

The formula expression shall be for each SafetyGoal:

Value(HWPMHF) = [Sum (lambdaSinglePointFault(HWFMSingleContribution) + lambdaResidualFault(HWFMSingleContribution))] + [Sum(LambdaSafetyComponent) * 1.10-9 * exposureTime(HWPMHF) * lambdaMultiplePointLatent(HWFMSingleContribution)]

Value(HWPMHF) is applied on calculatedValue extracted from electronic design level for final calculation and verification of the final PMHF probability. The selection between calculated and estimated value is a tool feature that allow first a calculation for estimation based on allocation field of failure rate and distribution. Only Component safety relevant are considered.

Sum(xxxxValue(xxxxLambdaSafetyComponent) is applied for estimated and calculated, and only counted once (identical safetyComponentClassName).

Connections

Connector	Source	Target

Connector	Source	Target
Association Source -> Destination	HWPMHFFormula	HWFMSingleContribution
Generalization Source -> Destination	HWPMHFFormula	FormulaDocumentation
Aggregation Source -> Destination	HWPMHFFormula	HWPMHF

Class HWSinglePointFaultMetricFormula

Element Base Classes: **FormulaDocumentation**

Element Notes:

This class describes the the single-point fault metric (in %) as ratio of impact of single-point and residual faults for a top level malfunction (link to violation of a SafetyGoal).

SPF metric = 100% - total (single point faults failure rate + residual faults failure rate) / total (safety related HWComponent failure rate)

The formula expression shall be for each SafetyGoal:

Value(SinglePointFaultMetric) = { 1 - [(Sum (lambdaSinglePointFault(FMSingleContribution) + lambdaResidualFault(FMSingleContribution)) / Sum(LambdaSafetyComponent)] } * 100

Value(SinglePointFaultMetric) is applied on estimatedValue from electronic design level for final calculation and verification of the final single-point fault metric. The selection between calculated and estimated value is a tool feature that allow first a calculation for estimation based on allocation field of failure rate and distribution. Only safety-related HWComponent are considered.

Sum(LambdaSafetyComponent) is only counted once for a HWElement (identical safetyComponentClassName).

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWSinglePointFaultMetricFormula	HWSinglePointFaultMetric

Connector	Source	Target
Association Source -> Destination	HWSinglePointFaultMetricFormula	HWFMSingleContribution
Generalization Source -> Destination	HWSinglePointFaultMetricFormula	FormulaDocumentation

Package Failure

Package Notes:

This sub-package describes the failure model of the hardware as derived from the requirements of the ISO 26262.

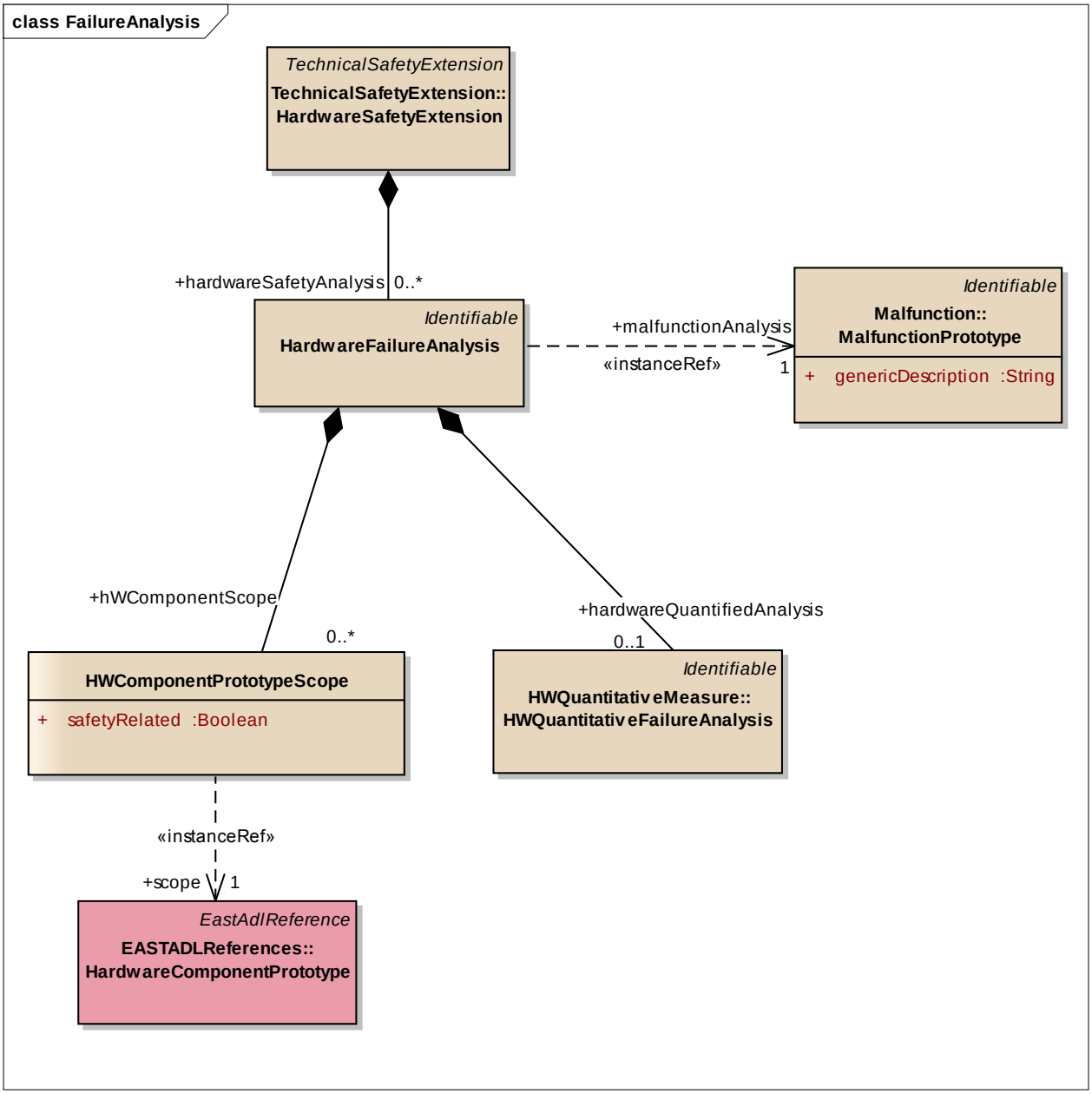


Figure 34: **FailureAnalysis** - *(Class diagram)*

Diagram Notes:

This diagram shows an overview of the hardware component failure extension root information where hardware related failure data and analysis shall be performed.

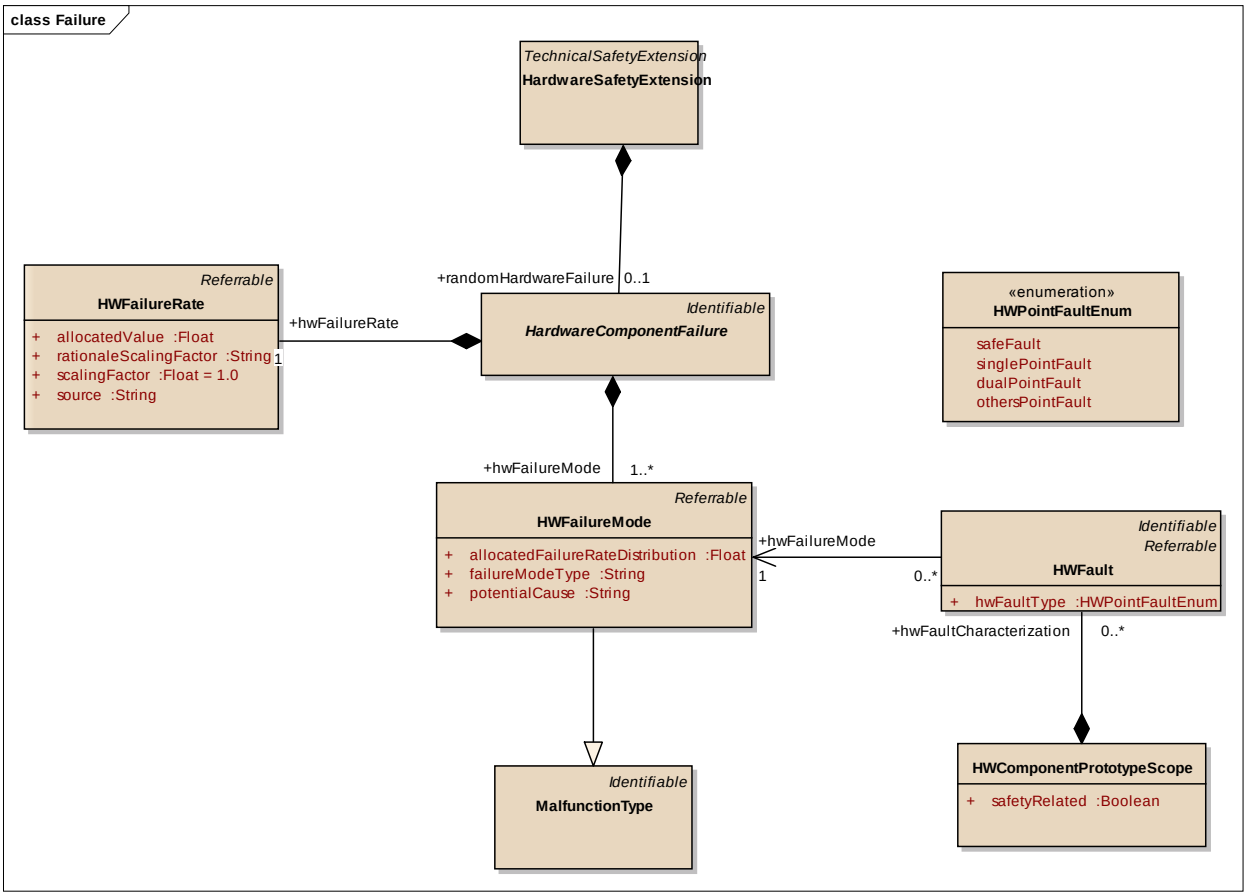


Figure 35: **Failure** - *(Class diagram)*

Diagram Notes:

This diagram shows an overview of the hardware component failure model.

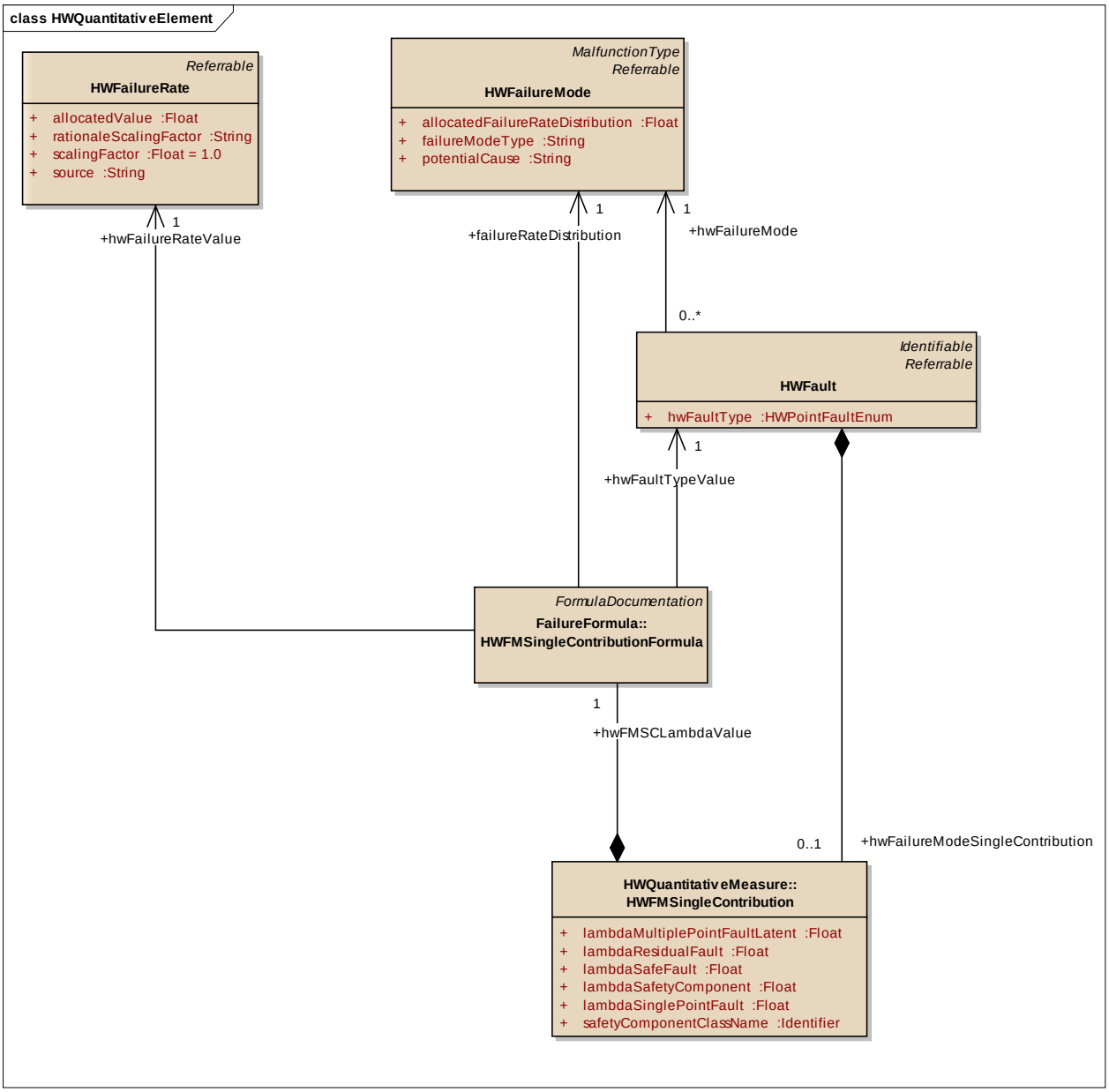


Figure 36: **HWQuantitativeElement** - *(Class diagram)*

Diagram Notes:

This diagram contains the calculation of the single failure mode contribution of HWComponent as preliminary step for the safety evaluation.

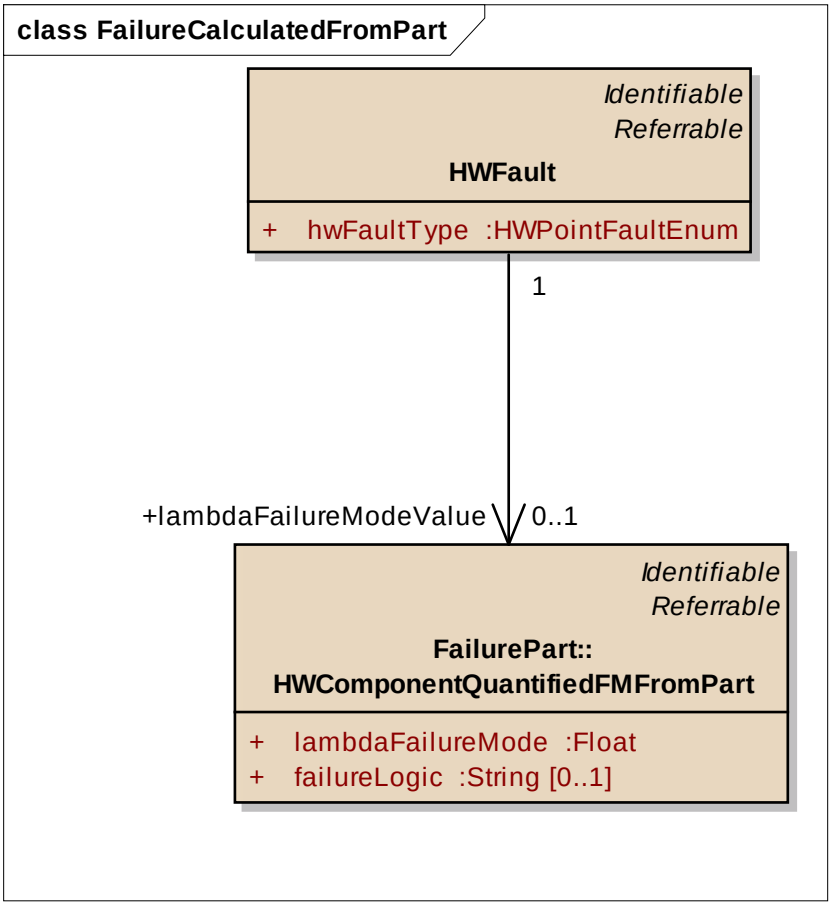


Figure 37: **FailureCalculatedFromPart** - *(Class diagram)*

Diagram Notes:

This diagram shows the instance reference of a failure mode of a hardware component on higher level and its interference with hardware element part calculations.

Class HardwareFailureAnalysis

Element Base Classes: **Identifiable**

Element Notes:

This class represent the container for all Hardware Failure Analysis.

Each safety goal (as malfunction) must lead to a safety analysis, so this class contains all the information related to the analysis as :

- the relation to the malfunction as the malfunctionPrototype for each analysis

- the HwComponentPrototypeScope to identify all hardware component specific to the context as HWComponentProtype inside a type composition

- the HWQuantifiedFailure Analysis performed on the level of the composition

Connections

Connector	Source	Target
Association Unspecified	HWComponentPrototypeScope	HardwareFailureAnalysis
Association Unspecified	HWQuantitativeFailureAnalysis	HardwareFailureAnalysis
Aggregation Source -> Destination	HardwareFailureAnalysis	HardwareSafetyExtension
Aggregation Source -> Destination	MalfunctionInstanceRef	HardwareFailureAnalysis
Dependency Source -> Destination	HardwareFailureAnalysis	MalfunctionPrototype
Generalization Source -> Destination	HardwareFailureAnalysis	Identifiable

Class HardwareComponentFailure

Element Base Classes: **Identifiable**

Element Notes:

This class describes the failure data extension for all HWComponents, including failure rate and failure mode.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HardwareComponentFailure	HardwareSafetyExtension
Generalization Source -> Destination	HardwareComponentFailure	Identifiable
Aggregation Source -> Destination	HWFailureMode	HardwareComponentFailure
Aggregation Source -> Destination	HWFailureRate	HardwareComponentFailure

Class HWFailureRate

Element Base Classes: **Referrable**

Element Notes:

This class captures the HWFailureRate of a HWComponent.

The appropriate HWFailureRate can be derived from e.g. Industry Source (see ISO Part 5 8.4.3) as an allocated value or calculated via analysis.

Connections

Connector	Source	Target
Association Source -> Destination	HWFMSingleContributionFormula	HWFailureRate
Generalization Source -> Destination	HWFailureRate	Referrable
Aggregation Source -> Destination	HWFailureRate	HardwareComponentFailure

Attributes

Attribute	Notes	Default
allocatedValue Float	FIT rate allocated to this HWComponent out of statistics for architectural evaluation and calculation of metrics and probabilistic methods. It shall be expressed in FIT.	
rationaleScalingFactor String	The rationaleScalingFactor shall provide a rationale, if a scaling factor different to 1.0 is applied.	
scalingFactor Float	The scalingFactor allows potential scaling between different sources of failure rates as described in ISO Part 5 Annex F.	1.0
source String	FIT rate source shall documented according to possible source as described in ISO 26262 Part 5 8.4.3: a) failure rate from industry source (IEC/TR 62380, IEC 61709, ...)	

Attribute	Notes	Default
	b) statistic based on return field or test c) Expert judgement	

Class HWFailureMode

Element Base Classes: **MalfunctionType, Referrable**

Element Notes:

This class describes a HWFailureMode of a HWComponent.

Each HWFailureMode of the HWComponent must have its own characterization for each linked malfunction (linked to violation of a SafetyGoal).

The HWFailureMode and HWFailureRateDistribution can be derived from e.g. Industry Source (see ISO Part 5 8.4.3).

The HWfailureMode as a specialization of A malfunction can be traced according Requirement tracing relation to a SafetyMechanismSpecification composed with QuantifiedHWDCProperty class to identify the Diagnosis Coverage value for Latent and/or Residual Fault to be able then to compute HW metrics.

Connections

Connector	Source	Target
Association Source -> Destination	HWFMSingleContributionFormula	HWFailureMode
Aggregation Source -> Destination	HWFailureMode	HardwareComponentFailure
Generalization Source -> Destination	HWFailureMode	Referrable
Association Source -> Destination	HWFault	HWFailureMode
Generalization Source -> Destination	HWFailureMode	MalfunctionType

Attributes

Attribute	Notes	Default

Attribute	Notes	Default
allocatedFailureRateDistribution Float	This attribute describes the allocated distribution of the failure rate of the specific failure mode (in percentage) of a HWComponent The sum of all failure rate distributions of all failure modes for a single hardware component must lead to the value 100% (may check for consistency).	
failureModeType String	This attribute textually describes the type of a failure mode of a HWComponent (e.g. "No value" for a sensor).	
potentialCause String	This attribute allows the documentation of the potential cause of the HWComponent failure mode (e.g. high temperature).	

Class HWFault

Element Base Classes: **Identifiable, Referrable**

Element Notes:

This class HWFault represent the characterization of a HWComponent Fault defined by tags as Safe Fault, SinglePointFault or MultiplePointFault of a specific FailureMode in a context of an Hardware Architecture.

HardwareFault can only exist for HardwareComponentPrototype when HWComponent are used given by the HWComponentPrototypeScope.

The related malfunction (link to violation of a SafetyGoals) is already linked with the FailureMode of the HardwareComponent via the HWSafetyGoalRelated meta class.

The different values are:

SafeFault: no violation of safety goal

ResidualOrSinglePointFault: direct violation of the SafetyGoal (1st order fault)

MultiplePointFault : violation of the SafetyGoal in combination with an independent failure of another component (minimum 2nd order)

Multiple-point fault for n>2 are considered as safe faults unless shown to be relevant in the technical safety concept (see ISO Part 5 7.4.3.2 Note 1).

Connections

Connector	Source	Target
Association Source -> Destination	HWFault	HWComponentQuantifiedFMFromPart
Association Source -> Destination	HWFMSingleContributionFormula	HWFault
Association Unspecified	HWFMSingleContribution	HWFault
Generalization Source -> Destination	HWFault	Referrable
Generalization Source -> Destination	HWFault	Identifiable
Association Unspecified	HWComponentPrototypeScope	HWFault
Association Source -> Destination	HWFault	HWFailureMode

Attributes

Attribute	Notes	Default
hwFaultType HWPointFaultEnum	Characterization of the Failure Mode for a single point related malfunction (linked to violation of a Safety Goal). singlePoint Hw Fault Type can be either SPF or Residual Possible Types are: • SafeFault (no violation of Safety Goal) • ResidualOrSinglePointFault (direct violation of Safety Goal (either covered by Safety Mechanism or not) • Multiple-Point-Fault (violation of Safety Goal in combination with another independent fault)	

Class HWComponentPrototypeScope

Element Base Classes:

Element Notes:

This class describes the context for the definition of a hardware component. The attribute defines a results of the analysis if the Hardware Component is safety related means impacted by the contribution to the violation of the malfunction

During modeling a design rule must be ensured that HardwareComponentType used as BaseHwComponent for the instanceRef relation is the same as the root hierarchy on the HwComponentprototypeScope creation

Connections

Connector	Source	Target
Association Unspecified	HwComponentScopeInstanceRef	HWComponentPrototypeScope
Association Unspecified	HWComponentPrototypeScope	HardwareFailureAnalysis
Dependency Source -> Destination	HWComponentPrototypeScope	HardwareComponentPrototype
Association Unspecified	HWComponentPrototypeScope	HWFault

Attributes

Attribute	Notes	Default
safetyRelated Boolean	This attribute stores the contribution of the HWComponent to a malfunction as a boolean information	

Enumeration HWMultiplePointFaultEnum

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
noMultiplePointFault		

Attribute	Notes	Default
DualPointFault		
TriplePointFault		
othersOrderPointFault		

Enumeration HWPointFaultEnum

Element Base Classes:

Element Notes:

This enumeration includes the possible characterizations for the attribute hwFaultType in Class HWFault.

For simplification and clarification only SafeFault, SinglePointFault and DualPointFault and othersPointFault are derived from the ISO Part 5 7.4.3.2.

SinglePointFault represents a first order fault, while DualPointFault represents a second order fault, and OthersPointFault order greater then two. For the hardware fault description, an cut set order of two is adequate. Therefore, an limited order of two (see ISO Part 5 7.4.3.2) can be defined. This means, that multiplePointFault represents an second order fault (dualPointFault).

The precise characterization of a HWFault (e.g. Multiple-Point-Latent) can be derived from the value of the attribute hwFaultType and a possible existence of a SafetyMechanism.

Attributes

Attribute	Notes	Default
safeFault	This literal describes the characterization as a safe fault.	
singlePointFault	This literal describes the characterization as a single-point of failure (direct violation).	
dualPointFault	This literal describes the characterization as a multiple-point fault (violation in combination with another independent fault).	

Attribute	Notes	Default
othersPointFault		

Package _instanceRef

Package Notes:

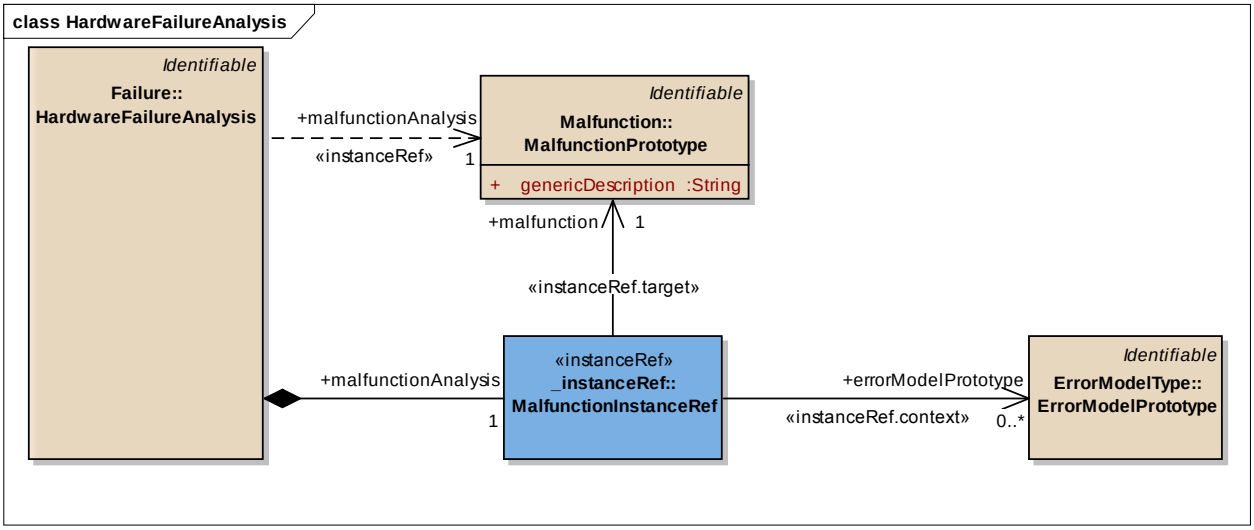


Figure 38: **HardwareFailureAnalysis** - *(Class diagram)*

Diagram Notes:

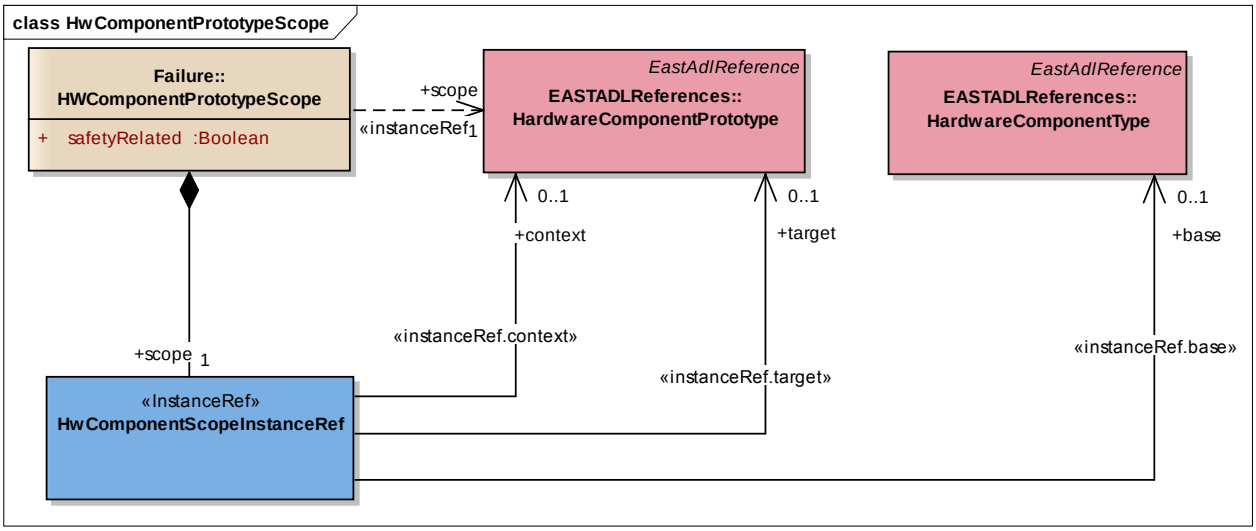


Figure 39: **HwComponentPrototypeScope** - *(Class diagram)*

Diagram Notes:

This diagram shows the context for HWComponent for an instanceRef between Type and Prototype

Class HwComponentScopeInstanceRef

Element Base Classes:

Element Notes:

This "instanceRef" meta-class is the container for holding the relation of HWComponentPrototypeScope in context of HWComponentType for the use of HWComponentPrototype.

Connections

Connector	Source	Target
Association Unspecified	HwComponentScopeInstanceRef	HWComponentPrototypeScope
Association Source -> Destination	HwComponentScopeInstanceRef	HardwareComponentType
Association Source -> Destination	HwComponentScopeInstanceRef	HardwareComponentPrototype
Association Source -> Destination	HwComponentScopeInstanceRef	HardwareComponentPrototype

Package FailurePart

Package Notes:

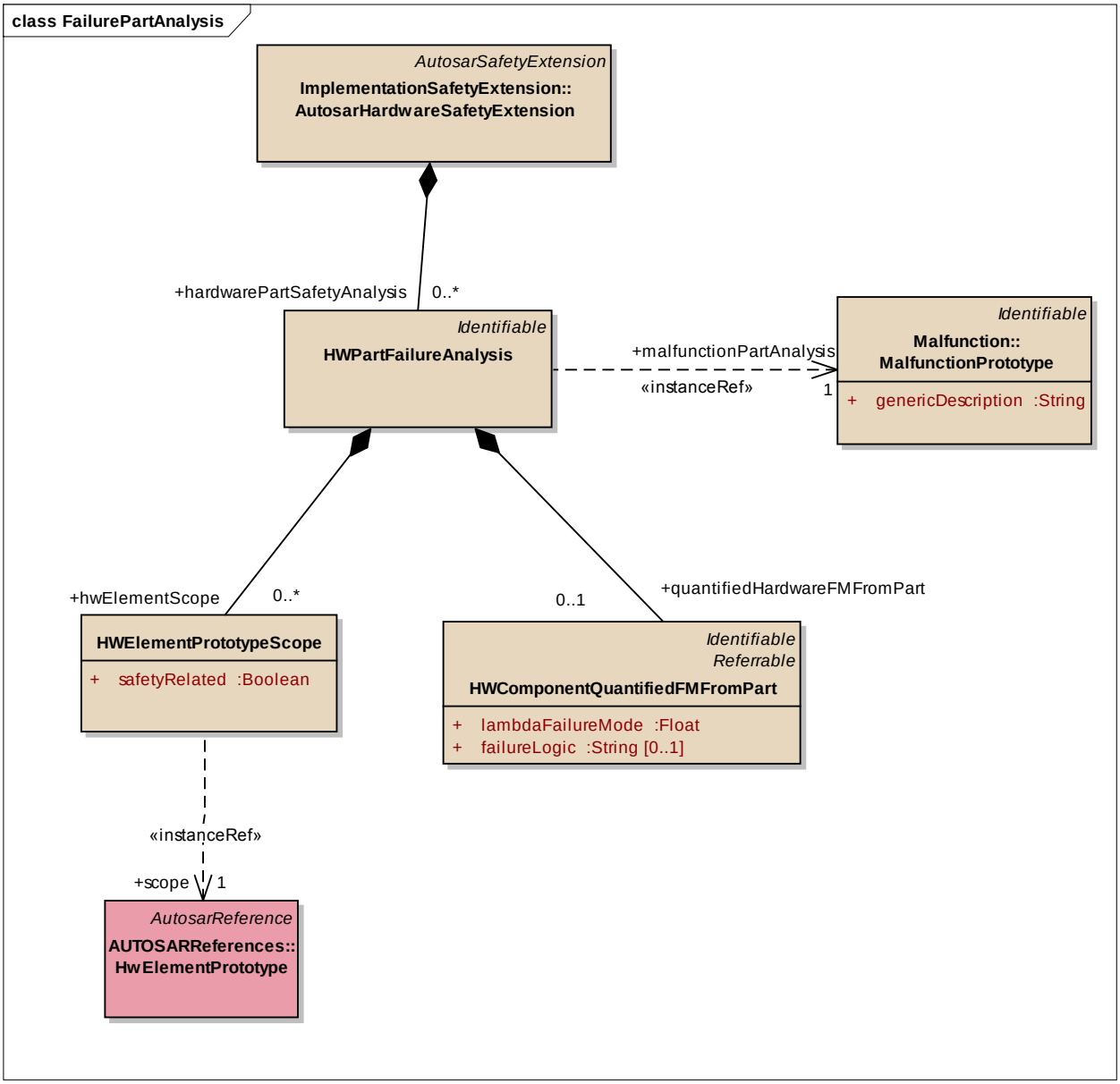


Figure 40: **FailurePartAnalysis** - *(Class diagram)*

Diagram Notes:

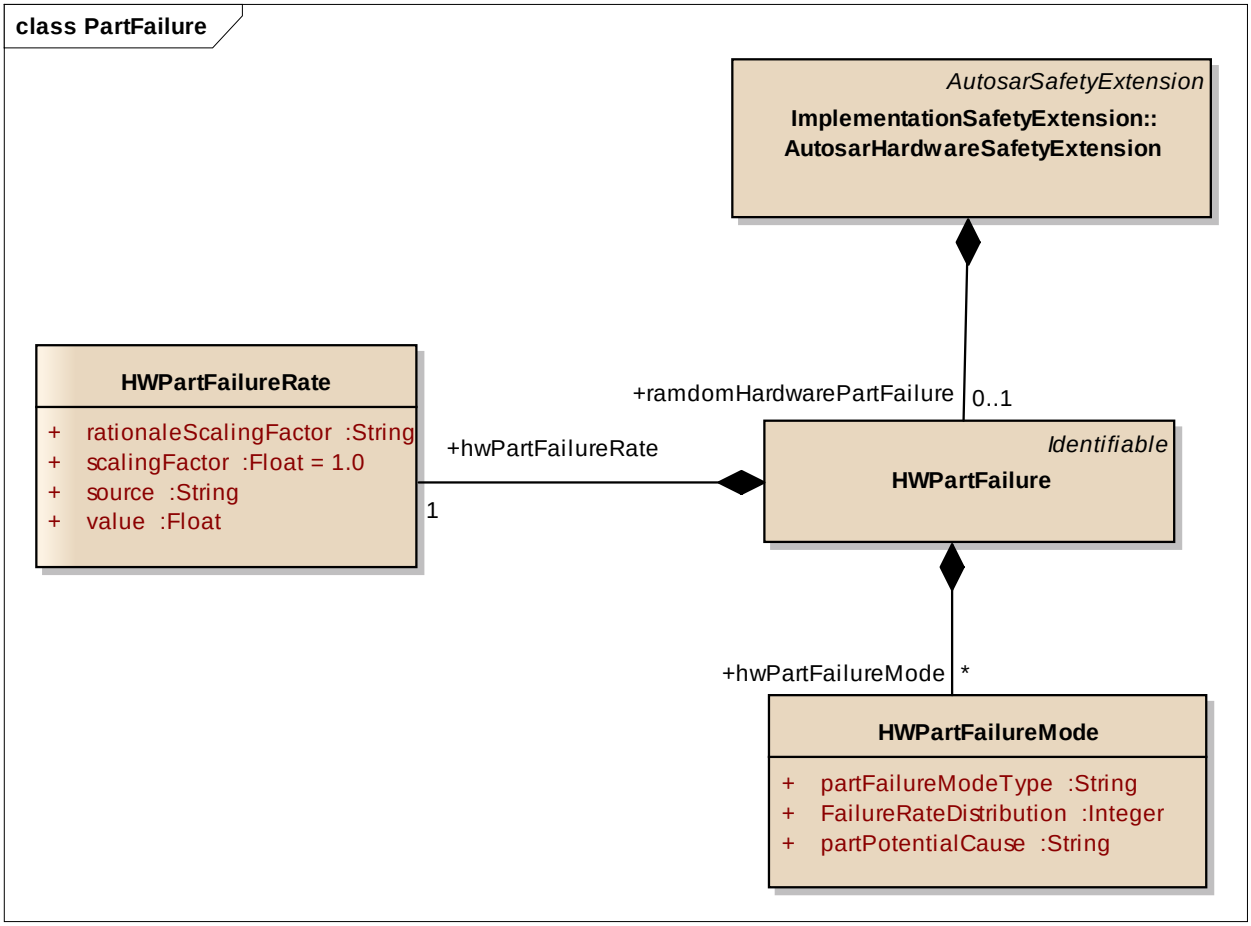


Figure 41: **PartFailure** - *(Class diagram)*

Diagram Notes:

This diagram shows the hardware part failures and its contribution to the hardware component failure on higher level.

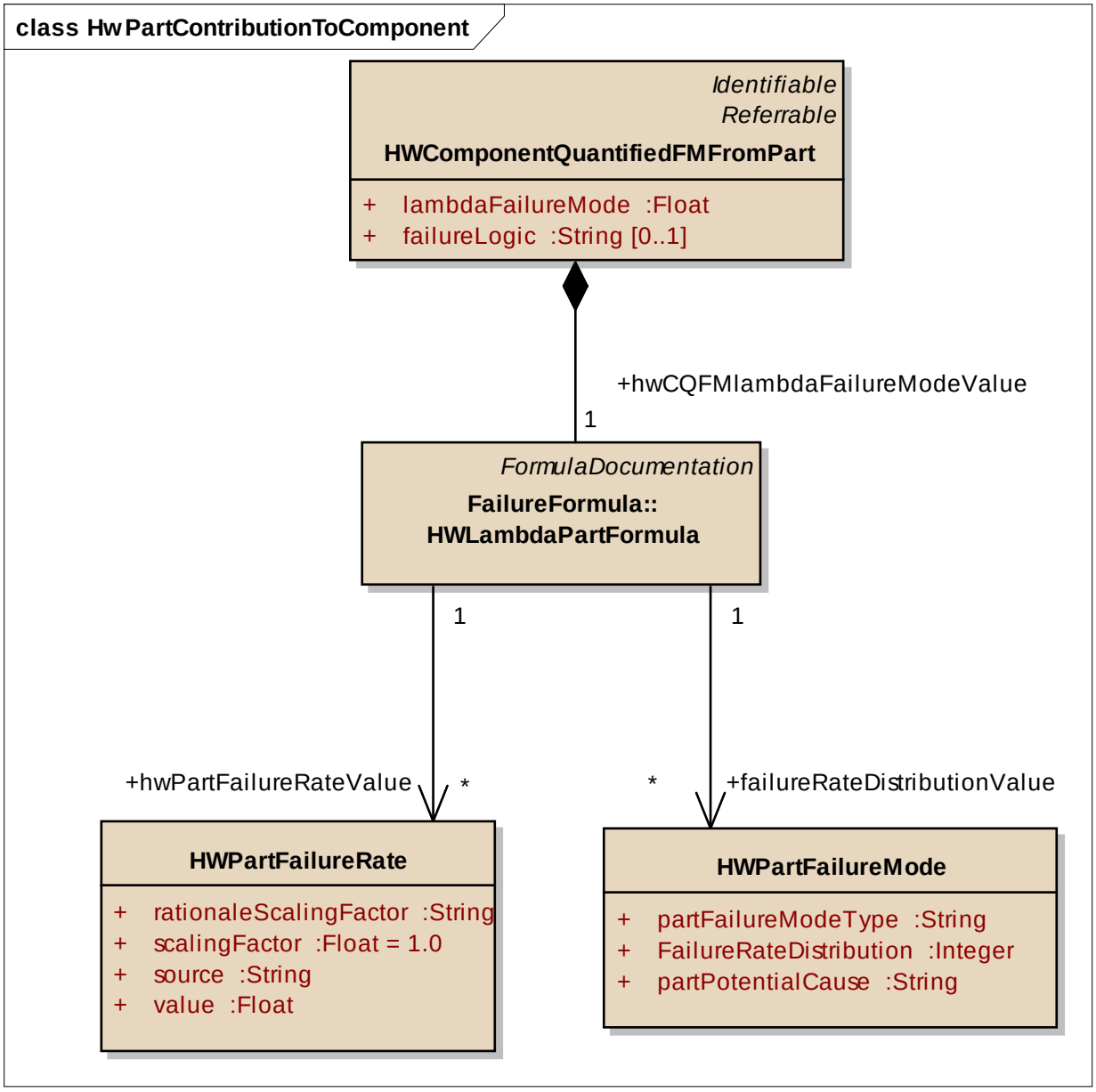


Figure 42: **HwPartContributionToComponent** - *(Class diagram)*

Diagram Notes:

This diagram shows the hardware part failures and its contribution to the hardware component failure on higher level.

Class HWComponentQuantifiedFMFromPart

Element Base Classes: **Identifiable, Referrable**

Element Notes:

This class describes the quantified failure rate of a failure mode of a HWComponent based on the contribution of each HWPartFailureMode of the related HWPart as AUTOSAR HW Element (calculated with the formula and stored in the attribute lambdaFailureMode).

The attribute SafetyComponentClassName is used to identify the HWComponent Class name for further calculation of all failure mode to the same HWComponent.

A quantified HW ComponentFailureMode must identify the related HWFailureMode of the HWComponent.

During modeling a design rule must be ensured that AutosarHWElementType used as BaseHwElement for the instanceRef relation is the same as the root hierarchy on the HwElement creation

Connections

Connector	Source	Target
Association Source -> Destination	HWFault	HWComponentQuantifiedFMFromPart
Generalization Source -> Destination	HWComponentQuantifiedFMFromPart	Identifiable
Aggregation Source -> Destination	HWLambdaPartFormula	HWComponentQuantifiedFMFromPart
Association Unspecified	HWPartFailureAnalysis	HWComponentQuantifiedFMFromPart
Generalization Source -> Destination	HWComponentQuantifiedFMFromPart	Referrable

Attributes

Attribute	Notes	Default
lambdaFailureMode Float	This attribute contains the quantified failure rate for the corresponding failure mode of the hardware component.	
failureLogic String	Define the specification of HWPartFailureMode contribution to HWFailureMode of the above abstraction layer (Block)on an external formalism or the path to the file	

Attribute	Notes	Default
	containing the external specification.	

Class HWElementPrototypeScope

Element Base Classes:

Element Notes:

This class describes the context for the definition of a hardware Element. The attribute defines a results of the analysis if the Hardware Component is safety related means impacted by the contribution to the violation of the malfunction

During modeling a design rule must be ensured that HWElementType used as BaseHwComponent for the instanceRef relation is the same as the root hierarchy on the HwElementPrototypeScope creation

Connections

Connector	Source	Target
Association Unspecified	HWElementPrototypeScope	HWPartFailureAnalysis
Dependency Source -> Destination	HWElementPrototypeScope	HwElementPrototype
Association Unspecified	HWElementPrototypeScope	HWElementScopeInstanceRef

Attributes

Attribute	Notes	Default
safetyRelated Boolean	This attribute stores the contribution of the HWElement to a malfunction as a boolean information	

Class HWPartFailure

Element Base Classes: **Identifiable**

Element Notes:

This class describes the failure data extension for all HWPart elements, including part failure rate and part failure mode.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWPartFailureRate	HWPartFailure
Association Unspecified	HWPartFailure	AutosarHardwareSafetyExtension
Aggregation Source -> Destination	HWPartFailureMode	HWPartFailure
Generalization Source -> Destination	HWPartFailure	Identifiable

Class HWPartFailureAnalysis

Element Base Classes: **Identifiable**

Element Notes:

This class represent the container for all Hardware Part Failure Analysis (Autosar HWElement) for a malfunction.

Each malfunction (as Hardware Failure Mode) must lead to a safety part analysis, so this class contains all the information related to the analysis as :

- the relation to the malfunction as the malfunctionPrototype for each analysis

- the HwElementPrototypeScope to identify all hardware Autosar Element specific to the context as HWComponentPrototype inside a type composition

- the HWComponentQuantifiedFMFromPart Analysis performed on the level of the composition of hardware part for contribution to the malfunction

Connections

Connector	Source	Target
Association Unspecified	HWElementPrototypeScope	HWPartFailureAnalysis
Aggregation Source -> Destination	MalfunctionInstanceRef	HWPartFailureAnalysis
Generalization Source -> Destination	HWPartFailureAnalysis	Identifiable

Connector	Source	Target
Association Unspecified	AutosarHardwareSafetyExtension	HWPartFailureAnalysis
Dependency Source -> Destination	HWPartFailureAnalysis	MalfunctionPrototype
Association Unspecified	HWPartFailureAnalysis	HWComponentQuantifiedFMFromPart

Class HWPartFailureMode

Element Base Classes:

Element Notes:

This class describes HWPartFailureModes of a HWPart as AUTOSAR HWElement. It aso captures the potential cause for a HWFailureMode as String (for documentation).

Each HWPartFailureMode of the Autosar HardwareElement must define a relation and contribution to a HWFailureMode of HardwareComponent (from hardware design level).

The HWFailureMode and HWFailureRateDistribution can be derived from e.g. Industry Source.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWPartFailureMode	HWPartFailure
Association Source -> Destination	HWLambdaPartFormula	HWPartFailureMode

Attributes

Attribute	Notes	Default
partFailureModeType String	This attribute textually describes the type of a failure mode of a HWPart element (e.g. "ShortCircuit" for a resistor).	
FailureRateDistribution Integer	This attribute describes the distribution of the failure rate of the HWPart element for the specific hardware part failure	

Attribute	Notes	Default
	mode in percentage.	
partPotentialCause String	This attribute allows the documentation of the potential cause of the HWPart failure mode (e.g. high temperature).	

Class HWPartFailureRate

Element Base Classes:

Element Notes:

This class captures the HWPartFailureRate of a AUTOSAR HWElement. Each AUTOSAR HWElement has one single Part HWFailureRate.

The appropriate Part FailureRate can be derived from e.g. Industry Source.

Connections

Connector	Source	Target
Association Source -> Destination	HWLambdaPartFormula	HWPartFailureRate
Aggregation Source -> Destination	HWPartFailureRate	HWPartFailure

Attributes

Attribute	Notes	Default
rationaleScalingFactor String	The rationaleScalingFactor shall provide a rationale, if a scaling factor different to 1.0 is applied.	
scalingFactor Float	The scalingFactor allows potential scaling between different sources of failure rates as described in ISO Part 5 Annex F.	1.0
source String	FIT rate source shall documented according to possible source as described in ISO 26262 Part 5 8.4.3: a) failure rate from industry source (IEC/TR 62380, IEC 61709, ...) b) statistic based on return field or test c) Expert judgement	
value	FIT rate for the hardware part element.	

Attribute	Notes	Default
Float	It shall be expressed in FIT.	

Package _instanceRef

Package Notes:

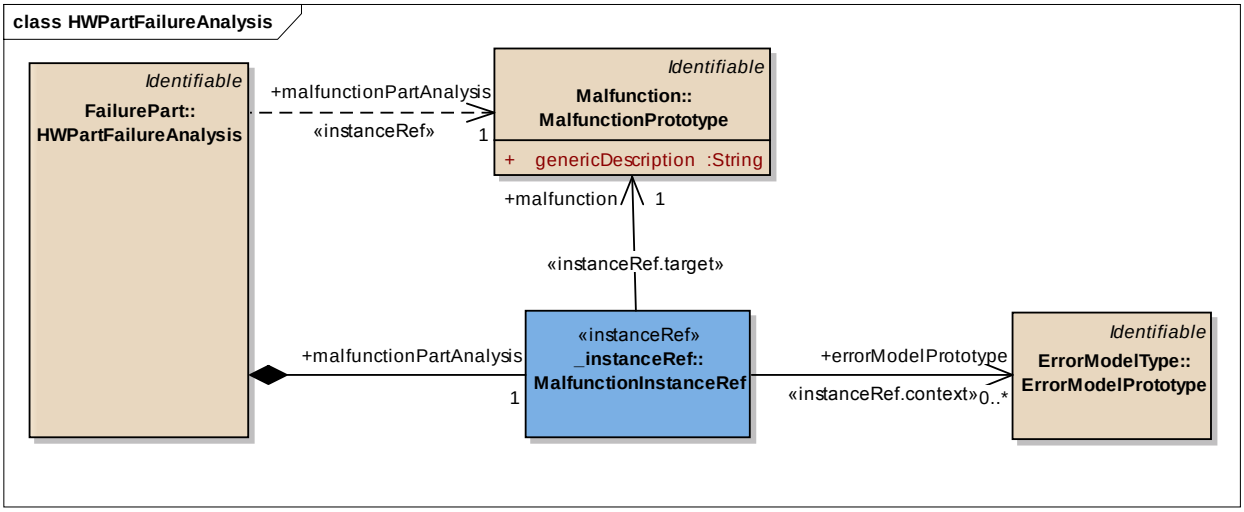


Figure 43: **HWPartFailureAnalysis** - *(Class diagram)*

Diagram Notes:

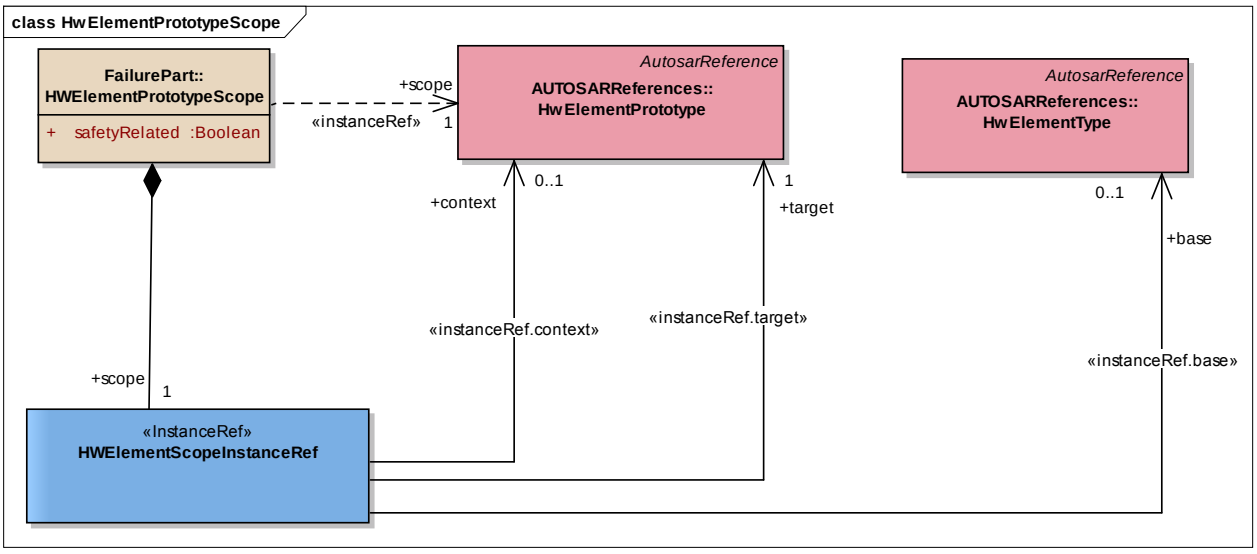


Figure 44: **HwElementPrototypeScope** - *(Class diagram)*

Diagram Notes:

This diagram shows the instance reference of a Hardware Component Quantified failure Value issued from Hardware Part.

Class HWElementScopeInstanceRef

Element Base Classes:

Element Notes:

This "instanceRef" meta-class is the container for holding the relation of HWElementScope in context of AutosarHWElementPrototype

Connections

Connector	Source	Target
Association Source -> Destination	HWElementScopeInstanceRef	HwElementPrototype
Association Source -> Destination	HWElementScopeInstanceRef	HwElementType
Association Source -> Destination	HWElementScopeInstanceRef	HwElementPrototype
Association Unspecified	HWElementPrototypeScope	HWElementScopeInstanceRef

Package HWQuantitativeMeasure

Package Notes:

This sub-package contains the storage and classification of the safety evaluation. In addition it includes the single failure mode contribution as basis for the concrete evaluation.

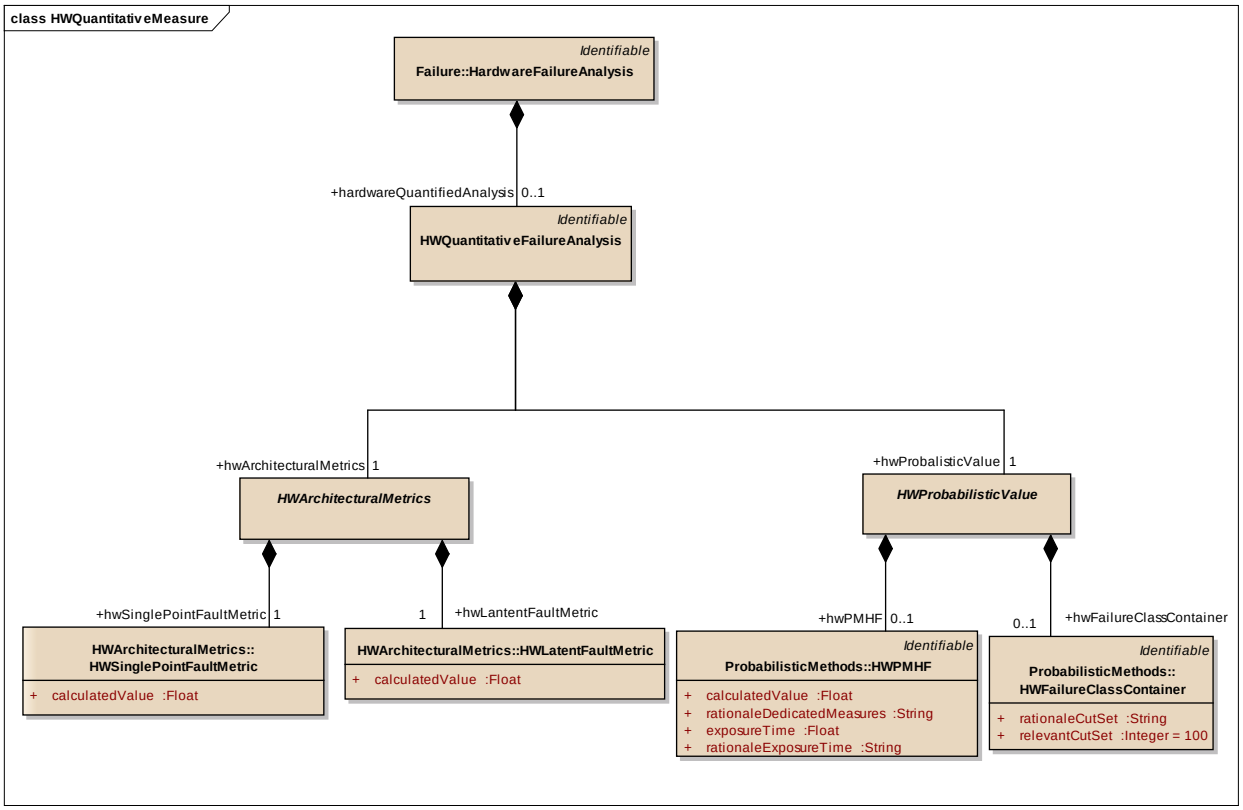


Figure 45: **HWQuantitativeMeasure** - *(Class diagram)*

Diagram Notes:

This diagram gives an overview about the quantitative analysis claimed by ISO 26262 Part 5 Clause 8 and Clause 9.

Class HWArchitecturalMetrics

Element Base Classes:

Element Notes:

This class represents an abstract definition of all quantified failure analysis required by the ISO Part 5 Clause 8. This class allows to map all meta class for the HWArchitecturalMetrics also described in the ISO Part 5-Annex C (Single-Point-Fault Metric, Latent-Fault Metric).

Each HWQuantifiedFailureAnalysis belongs to exactly one malfunction (link to violation of a SafetyGoal). The ASIL-TargetValue (e.g. ASIL-D) is derived from the SafetyGoal.

Connections

Connector	Source	Target

Connector	Source	Target
Aggregation Source -> Destination	HWSinglePointFaultMetric	HWArchitecturalMetrics
Aggregation Source -> Destination	HWArchitecturalMetrics	HWQuantitativeFailureAnalysis
Aggregation Source -> Destination	HWLatentFaultMetric	HWArchitecturalMetrics

Class HWFMSingleContribution

Element Base Classes:

Element Notes:

This class describes the single contribution in term of failure rate (lambda) to the elementary metrics of the HW Fault for each failure mode of a HWComponent. This entity is used to store preliminary element used in the context of architectural metrics and probabilistic measurement.

The calculation of the attribute is derived from the Formula Expression HWFMSingleContributionFormula

Connections

Connector	Source	Target
Association Source -> Destination	HWPMHFFormula	HWFMSingleContribution
Association Source -> Destination	HWLatentFaultMetricFormula	HWFMSingleContribution
Aggregation Source -> Destination	HWFMSingleContributionFormula	HWFMSingleContribution
Association Unspecified	HWFMSingleContribution	HWFault
Association Source -> Destination	HWSinglePointFaultMetricFormula	HWFMSingleContribution
Association Source -> Destination	HWFailureClassContributionFormula	HWFMSingleContribution

Attributes

Attribute	Notes	Default
lambdaMultiplePointFaultLatent Float	This attribute stores the specific failure rate for single failure mode contribution as multiple-point latent, lambda(MPF,L) for HW metrics calculation).	
lambdaResidualFault Float	This attribute stores the specific failure rate for single failure mode contribution as residual fault, lambda(RF).	
lambdaSafeFault Float	This attribute stores the specific failure rate for single failure mode contribution as safe fault, lambda(SF).	
lambdaSafetyComponent Float	This attribute stores the sum of specific failure rates for the hardware component for verification.	
lambdaSinglePointFault Float	This attribute stores the specific failure rate for single failure mode contribution as single-point fault, lambda(SPF).	
safetyComponentClassName Identifier	This attribute stores the name of the hardware component class.	

Class HWProbabilisticValue

Element Base Classes:

Element Notes:

This class represents an abstract definition of all failure analysis required by the ISO Part 5 Clause 9. This class allows to map all meta class for the evaluation of safety goal violation (PMHF and Failure Rate Class).

Each HWQuantifiedFailureAnalysis belongs to exactly one malfunction (link to violation of a SafetyGoal). The ASIL-TargetValue (e.g. ASIL-D) is derived from the SafetyGoal.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWProbabilisticValue	HWQuantitativeFailureAnalysis
Aggregation Source -> Destination	HWPMHF	HWProbabilisticValue
Aggregation Source -> Destination	HWFailureClassContainer	HWProbabilisticValue

Class HWQuantitativeFailureAnalysis

Element Base Classes: **Identifiable**

Element Notes:

This class represent the container for all quantified failure analysis required by the ISO 26262 Part 5 for a dedicated SafetyGoal. This class allows to cluster all meta class for the HWArchitecturalMetrics described in the ISO Part 5 Clause 8 (Single-Point-Fault Metric, Latent-Fault Metric) and probabilistic value for violation of safety goal (PMH) or Failure Class Method described in the ISO Part 5 Clause 9.

Each HWFailureAnalysis belongs to exactly one SafetyGoal. The ASIL-TargetValue (e.g. ASIL-D) is derived from the SafetyGoal.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWProbabilisticValue	HWQuantitativeFailureAnalysis
Association Unspecified	HWQuantitativeFailureAnalysis	HardwareFailureAnalysis
Generalization Source -> Destination	HWQuantitativeFailureAnalysis	Identifiable
Aggregation Source -> Destination	HWArchitecturalMetrics	HWQuantitativeFailureAnalysis

Package HWArchitecturalMetrics

Package Notes:

This sub-package describes the hardware architectural metrics as claimed by ISO 26262 Part 5 Clause 8. A detailed description of the architectural metrics can be found in ISO 26262 Part 5 Annex C.

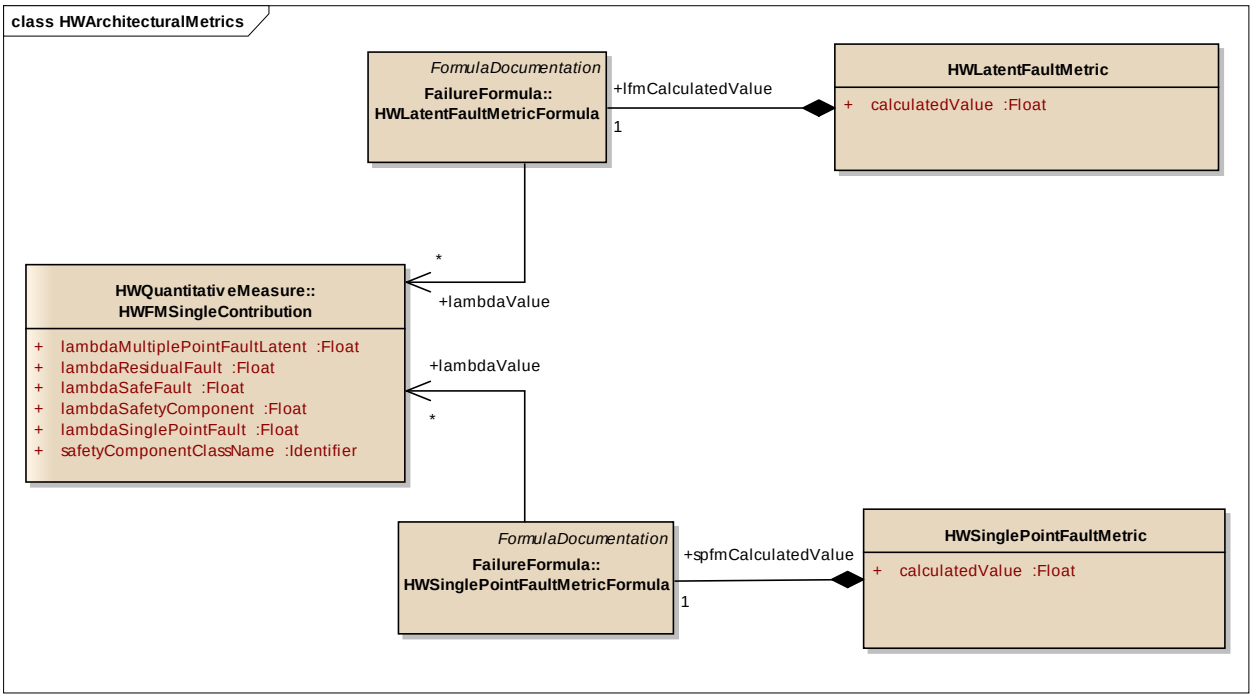


Figure 46: **HWArchitecturalMetrics** - *(Class diagram)*

Diagram Notes:

This diagram shows the calculation hardware architectural metrics as described in ISO Part 5-Clause 8 and Annex C.

Class HWLatentFaultMetric

Element Base Classes:

Element Notes:

This class is the representation of the latent fault metric, demanded by ISO Part 5 Clause 8. The latent fault metric describes the robustness of the hardware architecture to cope with multiple-point latent faults (also see ISO Part 5 Annex C).

The calculation is included in the class HWLatentFaultMetricFormula.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWLatentFaultMetric	HWArchitecturalMetrics
Aggregation	HWLatentFaultMetricFormula	HWLatentFaultMetric

Connector	Source	Target
Source -> Destination		

Attributes

Attribute	Notes	Default
calculatedValue Float	The calculatedValue is the result of the calculation of the latent fault metric (in %).	

Class HWSinglePointFaultMetric

Element Base Classes:

Element Notes:

This class is the representation of the single-point fault metric, demanded by ISO Part 5 Clause 8. The single-point fault metric describes the robustness of the hardware architecture to cope with single-point and residual faults (also see ISO Part 5 Annex C).

The calculation is included in the class HWSinglePointFaultMetricFormula.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWSinglePointFaultMetricFormula	HWSinglePointFaultMetric
Aggregation Source -> Destination	HWSinglePointFaultMetric	HWArchitecturalMetrics

Attributes

Attribute	Notes	Default
calculatedValue Float	The calculatedValue is the result of the calculation of the single-point fault metric (in %).	

Enumeration TargetValuesLFMetricEnum

Element Base Classes:

Element Notes:

Part 5-8.4.6 Table 5 (Possible source for the derivation of the target "latent-fault-metric" value)

Attributes

Attribute	Notes	Default
ASIL_D Float	This literal contains the target value for latent-fault metric for ASIL-D.	90.0
ASIL_C Float	This literal contains the target value for latent-fault metric for ASIL-C.	80.0
ASIL_B Float	This literal contains the target value for latent-fault metric for ASIL-B.	60.0

Enumeration TargetValuesSPFMetricEnum

Element Base Classes:

Element Notes:

Part 5-8.4.5 Table 4 (Possible source for the derivation of the target "single-point-fault-metric" value)

Attributes

Attribute	Notes	Default
ASIL_D Float	This literal contains the target value for single-point fault metric for ASIL-D.	99.0
ASIL_C Float	This literal contains the target value for single-point fault metric for ASIL-C.	97.0
ASIL_B Float	This literal contains the target value for single-point fault metric for ASIL-B.	90.0

Package ProbabilisticMethods

Package Notes:

This sub-package describes the residual risk of safety goal violation due to random hardware failures as claimed by ISO 26262 Part 5 Clause 9. This contains the probabilistic metric for random hardware failures (PMHF) and as an alternative the failure rate class method (FRC).

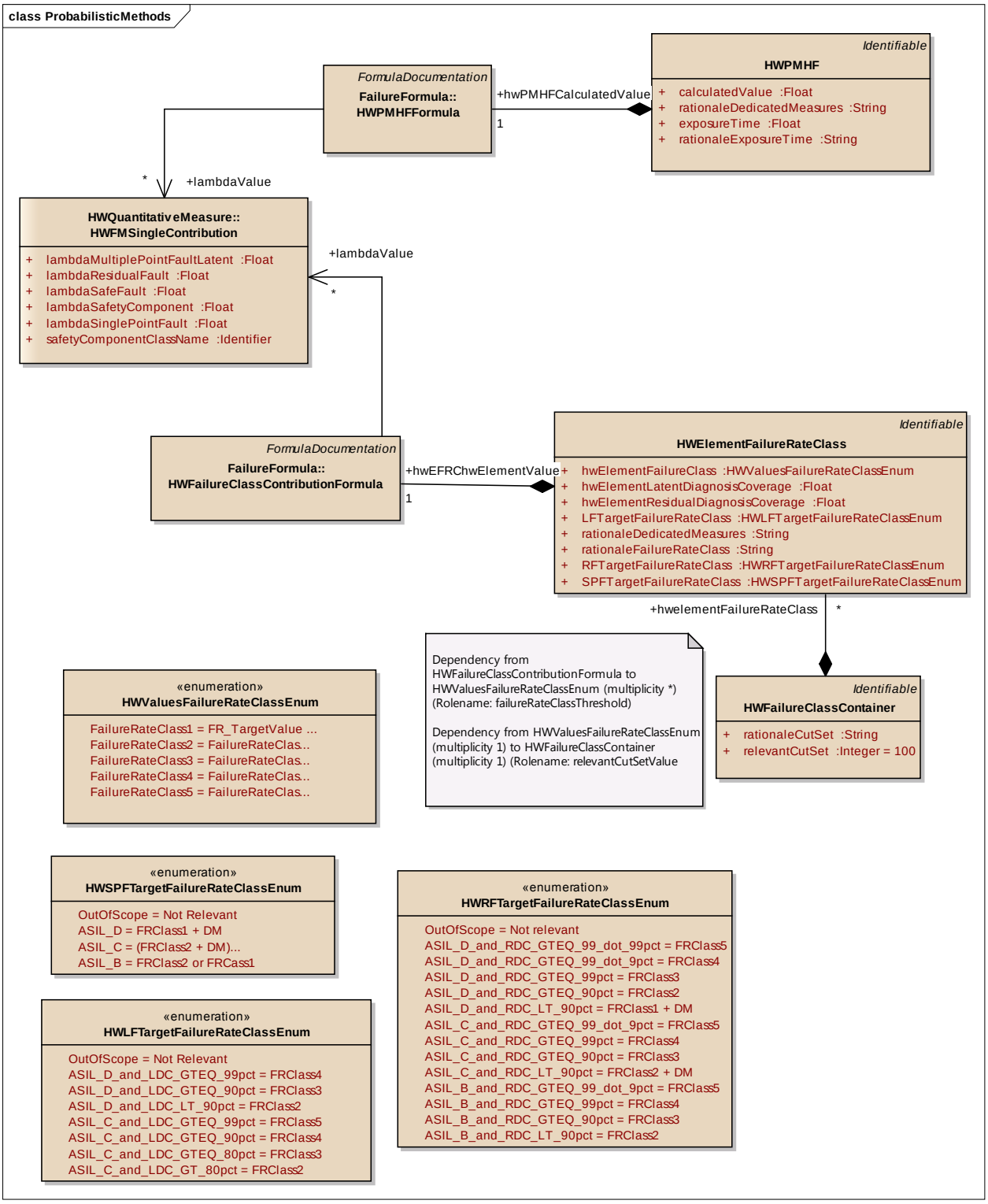


Figure 47: **ProbabilisticMethods** - *(Class diagram)*

Diagram Notes:

This diagram contains the evaluation of safety goal violation according to ISO 26262 Part 5 Clause 9. This contains the PMHF and the FRC.

Class HWElementFailureRateClass

Element Base Classes: **Identifiable**

Element Notes:

This class describes for a HWComponent, the FailureRateClass element to evaluate measure for a malfunction (link to violation of a safety goal) for a single element. This violation is based on failure rate class according to context of evaluation such as ASIL level, list of HWFault and diagnosis coverage of the HWComponent as HW Element. It allows also storing the target for failure rate class, relevant or not depending of the possible HWFault of the failure mode of the HWComponent as hardware Element. Furthermore if dedicated measures (DM) are required due to failure class target matching and the necessary information are captured as a textual description.

The calculation of the attribute HWElementFailureClass and HWElementDiagnosisCoverage is derived from the Formula Expression FMSingleContributionFormula.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWFailureClassContributionFormula	HWElementFailureRateClass
Generalization Source -> Destination	HWElementFailureRateClass	Identifiable
Aggregation Source -> Destination	HWElementFailureRateClass	HWFailureClassContainer

Attributes

Attribute	Notes	Default
hwElementFailureClass HWValuesFailureRateClassEnum	Failure Rate Class taken from HWValuesRateClassEnum based on the failure rate of the hardware component.	
hwElementLatentDiagnosisCoverage Float	The diagnostic coverage with respect to latent faults on hardware element level, calculated with the specific failure rate of all latent multiple-point faults and the overall failure rate of the hardware part element.	
hwElementResidualDiagnosisCoverage	The diagnostic coverage with respect to residual faults on hardware element level, calculated with the specific failure rate of all single-point and residual faults and the overall	

Attribute	Notes	Default
Float	failure rate of the hardware part element.	
LFTargetFailureRateClass HWLFTargetFailureRateClassEnum	Target Failure Rate Class for multiple-point latent faults, taken from HWLFTargetFailureRateClassEnum.	
rationaleDedicatedMeasures String	Provides rationale for dedicated measures, if required. According to ISO 26262 Part 5 9.4.2.4, examples for dedicated measures are a) design features such as hardware part over design (e.g. electrical or thermal stress rating) or physical separation (e.g. spacing of contacts on a printed circuit board); b) a special sample test of incoming material to reduce the risk of occurrence of this failure mode; c) a burn-in test; d) a dedicated control set as part of the control plan; and e) assignment of safety-related special characteristics.	
rationaleFailureRateClass String	Rationale for matching criteria on Failure Rate Class.	
RFTargetFailureRateClass HWRFTargetFailureRateClassEnum	Target Failure Rate Class for residual faults, taken from HWRFTargetFailureRateClassEnum.	
SPFTargetFailureRateClass HWSPFTargetFailureRateClassEnum	Target Failure Rate Class for single-point faults, taken from HWSPFTargetFailureRateClassEnum.	

Class HWFailureClassContainer

Element Base Classes: **Identifiable**

Element Notes:

This class is container to store all HW element failure class results and associated assumptions taken (number of cut-set as typical).

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWFailureClassContainer	HWProbabilisticValue
Generalization Source -> Destination	HWFailureClassContainer	Identifiable
Aggregation Source -> Destination	HWElementFailureRateClass	HWFailureClassContainer

Attributes

Attribute	Notes	Default
rationaleCutSet String	This attribute provides a textual rationale for the number of relevant cut-sets.	
relevantCutSet Integer	This attributes stores the number of relevant cut sets.	100

Enumeration HWLFTargetFailureRateClassEnum

Element Base Classes:

Element Notes:

ISO 26262 Part 5 9.4.3.11 -Table 9 (Targets of failure rate class and coverage of hardware part regarding dual-point faults)

DM: Dedicated measures

LDC: Diagnostic coverage with respect to latent faults

Additionally, OUT-OF-SCOPE was added.

Attributes

Attribute	Notes	Default
OutOfScope String	This literal describes values which are out of scope for the analysis.	Not Relevant

Attribute	Notes	Default
ASIL_D_and_LDC_GTEQ_99pct String	This literal describes a single cell with value target failure rate class 4 in the table for ASIL-D and latent diagnostic coverage >= 99%.	FRClass4
ASIL_D_and_LDC_GTEQ_90pct String	This literal describes a single cell with value target failure rate class 3 in the table for ASIL-D and latent diagnostic coverage >= 90%.	FRClass3
ASIL_D_and_LDC_LT_90pct String	This literal describes a single cell with value target failure rate class 2 in the table for ASIL-D and latent diagnostic coverage < 90%.	FRClass2
ASIL_C_and_LDC_GTEQ_99pct String	This literal describes a single cell with value target failure rate class 5 in the table for ASIL-C and latent diagnostic coverage >= 99%.	FRClass5
ASIL_C_and_LDC_GTEQ_90pct String	This literal describes a single cell with value target failure rate class 4 in the table for ASIL-C and latent diagnostic coverage >= 90%.	FRClass4
ASIL_C_and_LDC_GTEQ_80pct String	This literal describes a single cell with value target failure rate class 3 in the table for ASIL-C and latent diagnostic coverage >= 80%. Rationale provided by ISO 26262 Part 5 9.4.3.9.	FRClass3
ASIL_C_and_LDC_GT_80pct String	This literal describes a single cell with value target failure rate class 2 in the table for ASIL-C and latent diagnostic coverage < 80%. Rationale provided by ISO 26262 Part 5 9.4.3.9.	FRClass2

Class HWPMHF

Element Base Classes: **Identifiable**

Element Notes:

This class describes the Probabilistic Metric for random Hardware Failures (PMHF) as in ISO Part 5 Clause 9.4.2.

A simplified alculation is included in the class HWPMHFFormula.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HWPMHF	HWProbabilisticValue
Generalization Source -> Destination	HWPMHF	Identifiable
Aggregation Source -> Destination	HWPMHFFormula	HWPMHF

Attributes

Attribute	Notes	Default
calculatedValue Float	The calculatedValue is the result of the calculation of the PMHF (in FIT).	
rationaleDedicatedMeasures String	The attribute rationaleDedicatedMeasures shall allow to define a rationale for applied dedicated measures in the design.	
exposureTime Float	The exposure time is the duration of exposure, shall be expressed in h.	
rationaleExposureTime String	The attribute rationaleExposureTime is for Documentation of rationale for Exposure Time.	

Enumeration HWRFTargetFailureRateClassEnum

Element Base Classes:

Element Notes:

ISO 26262 Part 5 9.4.3.6 -Table 8 (Maximum failure rate classes for a given diagnostic coverage of the hardware part - residual faults).

DM: Dedicated measures

RDC: Diagnostic coverage with respect to residual faults

This class describes the threshold for Residual Failure according to ASIL level and identifying Failure Class Rate limit (FRClassx) and Dedicated Measure (DM) if necessary. Notice that RDC is addressing the hwElementResidualDiagnosisCoverage parameter of the HWElementFailureRateClass

Additionally, "OUT-OF-SCOPE" and "ASIL-D and RDC >=99.99%" according to ISO 26262 Part 5 9.4.3.7.

Attributes

Attribute	Notes	Default
OutOfScope String	This literal describes values which are out of scope for the analysis.	Not relevant
ASIL_D_and_RDC_GTEQ_99_dot_99pct String	This literal describes a single cell with value target failure rate class 5 in the table for ASIL-D and residual fault diagnostic coverage >= 99.99%. Failure Rate Class determined according to ISO 26262 Part 5 9.4.3.7.	FRClass5
ASIL_D_and_RDC_GTEQ_99_dot_9pct String	This literal describes a single cell with value target failure rate class 4 in the table for ASIL-D and residual fault diagnostic coverage >= 99.9%.	FRClass4
ASIL_D_and_RDC_GTEQ_99pct String	This literal describes a single cell with value target failure rate class 3 in the table for ASIL-D and residual fault diagnostic coverage >= 99%.	FRClass3
ASIL_D_and_RDC_GTEQ_90pct String	This literal describes a single cell with value target failure rate class 2 in the table for ASIL-D and residual fault diagnostic coverage >= 90%.	FRClass2
ASIL_D_and_RDC_LT_90pct String	This literal describes a single cell with value target failure rate class 1 + dedicated measures in the table for ASIL-D and residual fault diagnostic coverage < 90%.	FRClass1 + DM
ASIL_C_and_RDC_GTEQ_99_dot_9pct String	This literal describes a single cell with value target failure rate class 5 in the table for ASIL-C and residual fault diagnostic coverage >= 99.9%.	FRClass5
ASIL_C_and_RDC_GTEQ_99pct String	This literal describes a single cell with value target failure rate class 4 in the table for ASIL-C and residual fault diagnostic coverage >= 99%.	FRClass4
ASIL_C_and_RDC_GTEQ_90pct String	This literal describes a single cell with value target failure rate class 3 in the table for ASIL-C and residual fault diagnostic coverage >= 90%.	FRClass3
ASIL_C_and_RDC_LT_90pct String	This literal describes a single cell with value target failure rate class 2 + dedicated measures in the table for ASIL-C and residual fault diagnostic coverage < 90%.	FRClass2 + DM

Attribute	Notes	Default
ASIL_B_and_RDC_GTEQ_99_dot_9pct String	This literal describes a single cell with value target failure rate class 5 in the table for ASIL-B and residual fault diagnostic coverage >= 99.9%.	FRClass5
ASIL_B_and_RDC_GTEQ_99pct String	This literal describes a single cell with value target failure rate class 4 in the table for ASIL-B and residual fault diagnostic coverage >= 99%.	FRClass4
ASIL_B_and_RDC_GTEQ_90pct String	This literal describes a single cell with value target failure rate class 3 in the table for ASIL-B and residual fault diagnostic coverage >= 90%.	FRClass3
ASIL_B_and_RDC_LT_90pct String	This literal describes a single cell with value target failure rate class 2 in the table for ASIL-B and residual fault diagnostic coverage < 90%.	FRClass2

Enumeration HWSPFTargetFailureRateClassEnum

Element Base Classes:

Element Notes:

ISO 26262 Part 5 9.4.3.5 -Table 7 (Targets of failure rate classes of hardware parts regarding single-point faults)

DM: Dedicated measures

Additionally, OUT-OF-SCOPE was added.

Attributes

Attribute	Notes	Default
OutOfScope String	This literal describes values which are out of scope for the analysis.	Not Relevant
ASIL_D String	This literal describes a single cell with value target failure rate class 1 + dedicated measures in the table for ASIL-D.	FRClass1 + DM
ASIL_C String	This literal describes a single cell with value target failure rate class 2 + dedicated measures or failure rate class 1 in the table for ASIL-C.	(FRClass2 + DM) or FRClass1

Attribute	Notes	Default
ASIL_B String	This literal describes a single cell with value target failure rate class 2 or failure rate class 1 in the table for ASIL-B.	FRClass2 or FRCass1

Enumeration HWTargetValuesPMHFEnum

Element Base Classes:

Element Notes:

Target values for PMHF according to ISO 26262 Part 5 9.4.2.1. The values here are described in FIT (ppm/h) or FIT.

ASIL-D = 1.10-8 h-1 = 10 ppm/h

ASIL-C = ASIL-B = 1.10-7 => 100 ppm/h

Attributes

Attribute	Notes	Default
ASIL_D Float	This attributes stores target value for PMHF for ASIL-D.	10.0
ASIL_C Float	This attributes stores target value for PMHF for ASIL-C.	100.0
ASIL_B Float	This attributes stores target value for PMHF for ASIL-B.	100.0

Enumeration HWValuesFailureRateClassEnum

Element Base Classes:

Element Notes:

FailureRateClass value correspond to the maximum value applied in the Failure Rate Class X considering that lower value is Class X-1 (and 0 for class 1). The failure rate class values are determined according to ISO 26262 Part 5 9.4.3.3.

Failure Class are based on the number of relevant cutset.

Attributes

Attribute	Notes	Default
FailureRateClass1 Float	This attribute contains the maximum value for failure rate class 1 ranking (in FIT). It is computed from the allocated FIT rate to the analysis (as targetValue of HWFailureClassContainer) derided by the number of relevant cut-set of the analysis (as relevantCutSet of HWFailureClassContainer)	FR_TargetValue / relevantCutSet
FailureRateClass2 Float	This attribute contains the maximum value for failure rate class 2 ranking (in FIT).	FailureRateClass1 * 10
FailureRateClass3 Float	This attribute contains the maximum value for failure rate class 3 ranking (in FIT).	FailureRateClass2 * 10
FailureRateClass4 Float	This attribute contains the maximum value for failure rate class 4 ranking (in FIT).	FailureRateClass3 * 10
FailureRateClass5 Float	This attribute contains the maximum value for failure rate class 5 ranking (in FIT).	FailureRateClass4 * 10

Package Hazards

Package Notes:

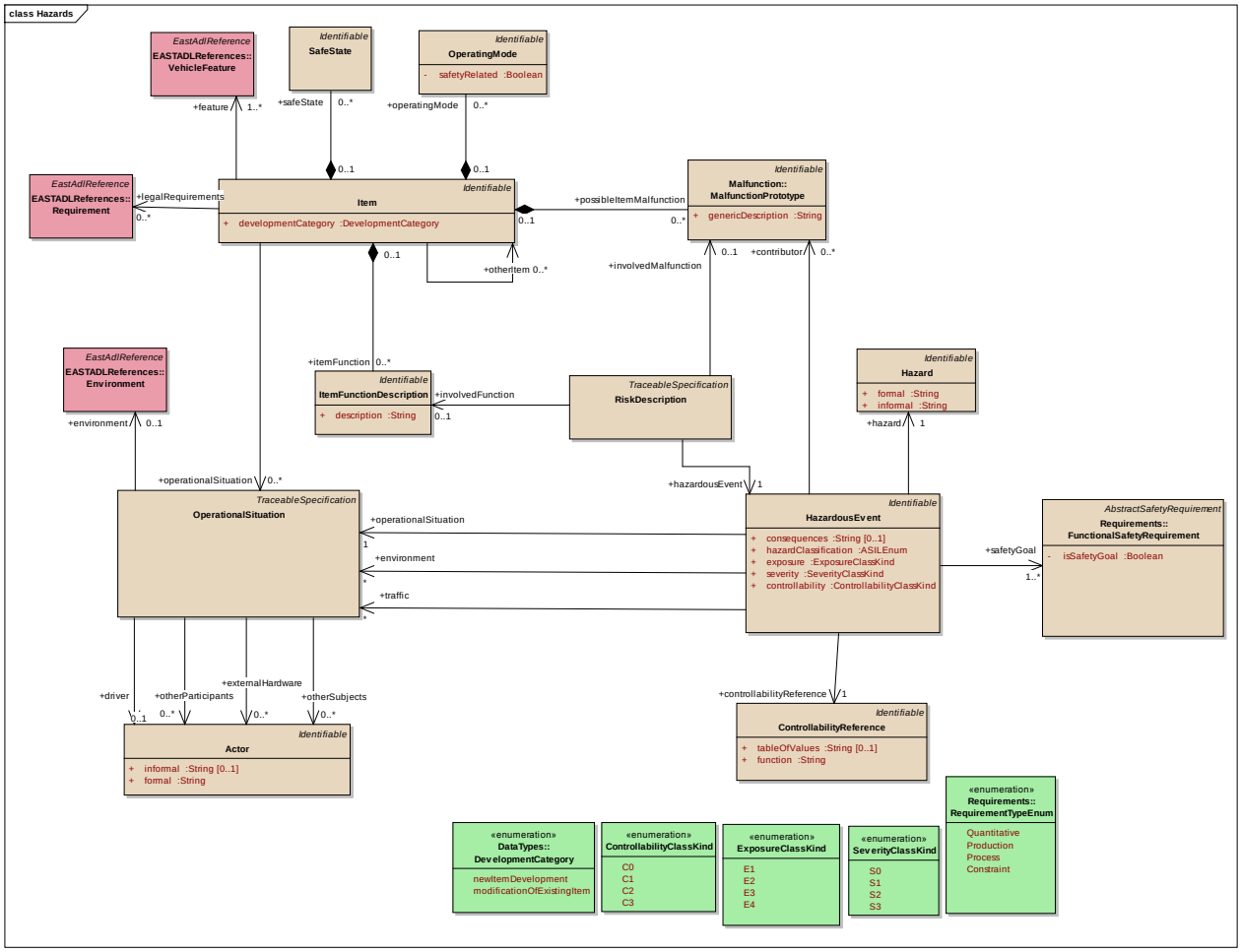


Figure 48: **Hazards** - *(Class diagram)*

Diagram Notes:

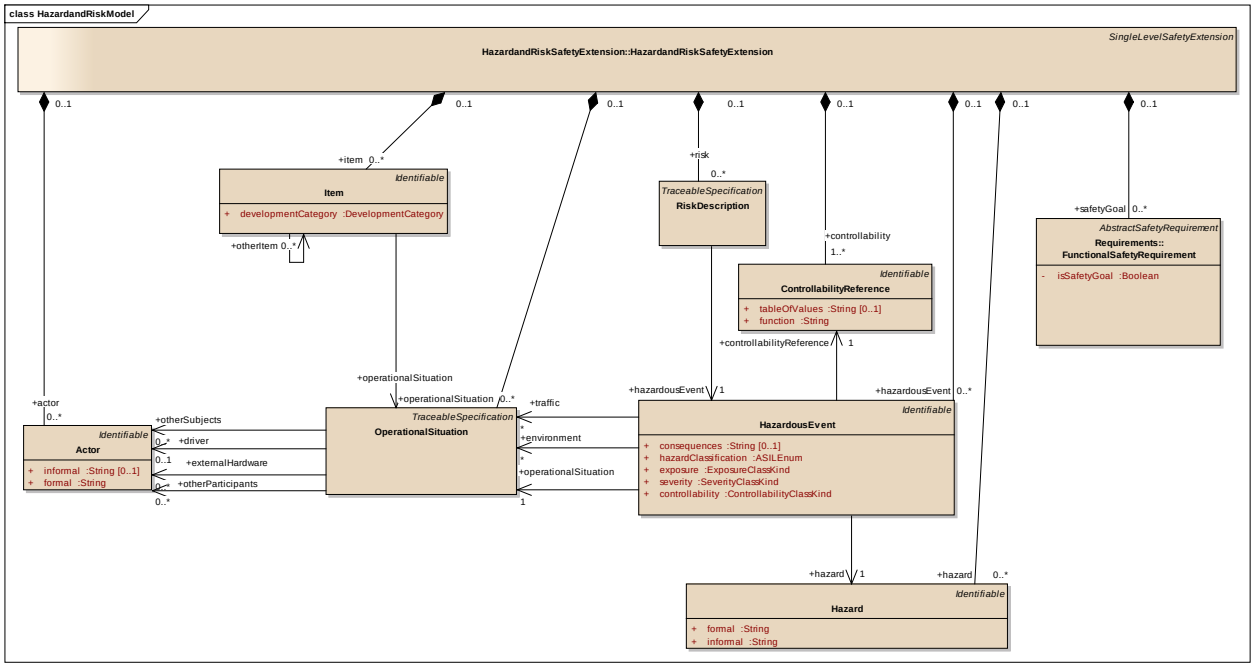


Figure 49: **HazardandRiskModel** - *(Class diagram)*

Diagram Notes:

Class Actor

Element Base Classes: **Identifiable**

Element Notes:

The “Actor” class is an abstract class which provides the structure that is needed for the contributions of different actors to the operational situation and the hazard.

Note:

The risk assessment of hazardous events focuses on the harm to each person potentially at risk – including the driver or the passengers of the vehicle causing the hazardous event, and other persons potentially at risk such as cyclists, pedestrians or occupants of other vehicles.

Reference:

ISO26262-3-7.4.3.2

Connections

Connector	Source	Target

Connector	Source	Target
Generalization Source -> Destination	Actor	Identifiable
Association Source -> Destination	OperationalSituation	Actor
Association Source -> Destination	OperationalSituation	Actor
Association Source -> Destination	OperationalSituation	Actor
Aggregation Source -> Destination	Actor	HazardandRiskSafetyExtension
Association Source -> Destination	OperationalSituation	Actor

Attributes

Attribute	Notes	Default
informal String	Provides the possibility to capture the informal description of the contribution of a particular actor	
formal String	Provides the possibility to capture the formal description of the contribution of a particular actor	

Enumeration ControllabilityClassKind

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
C0		
C1		

Attribute	Notes	Default
C2		
C3		

Class ControllabilityReference

Element Base Classes: **Identifiable**

Element Notes:

The class “ControllabilityReference” is introduced to provide the possibility to capture diagrams. These diagrams are based on road tests and enable a determination of the controllability parameter of the hazardous event.

Connections

Connector	Source	Target
Association Source -> Destination	HazardousEvent	ControllabilityReference
Generalization Source -> Destination	ControllabilityReference	Identifiable
Aggregation Source -> Destination	ControllabilityReference	HazardandRiskSafetyExtension

Attributes

Attribute	Notes	Default
tableOfValues String	Provides the possibility to capture the diagrams in form of tables of values.	
function String	Provides the possibility to capture the diagram in form of functions.	

Enumeration ExposureClassKind

Element Base Classes:

Element Notes:

The number of vehicles equipped with the item shall not be considered when estimating the probability of exposure.

Reference:

ISO 26262-3-7.4.3.5

Attributes

Attribute	Notes	Default
E1		
E2		
E3		
E4		

Class Hazard

Element Base Classes: **Identifiable**

Element Notes:

A hazard describes a potential source of harm. Important is that it is formulated in terms of behavior that can be observed on vehicle level.

Hazards shall be defined in terms of the conditions or behavior that can be observed at the vehicle level.

The hazards shall be determined systematically by using adequate techniques, such as brainstorming, checklists, quality history, FMEA and field studies.

Reference:

ISO 26262-3-7.4.2.2.2

ISO 26262-3-7.4.2.2.1

Connections

Connector	Source	Target
Aggregation Source -> Destination	Hazard	HazardandRiskSafetyExtension
Association Source -> Destination	HazardousEvent	Hazard
Generalization Source -> Destination	Hazard	Identifiable

Attributes

Attribute	Notes	Default
formal String	Provides the possibility to capture the formal description of the hazard.	
informal String	Provides the possibility to capture the informal description of the hazard.	

Class HazardousEvent

Element Base Classes: **Identifiable**

Element Notes:

The hazardous event describes a relevant outcome of combinations of a hazard and an operational situation.

An ASIL shall be determined for each hazardous event using the parameters "severity", "probability of exposure" and "controllability" in accordance with Table 4.

Semantic:

The ASIL shall be calculated automatically by using the information given in ISO26262-3-Table.4

Reference:

ISO 26262-3-7.4.2.2.3

ISO 26262-3-7.4.4.1

ISO 26262-3-Table.4

Connections

Connector	Source	Target

Connector	Source	Target
Association Source -> Destination	HazardousEvent	FunctionalSafetyRequirement
Generalization Source -> Destination	HazardousEvent	Identifiable
Aggregation Source -> Destination	HazardousEvent	HazardandRiskSafetyExtension
Association Source -> Destination	HazardousEvent	OperationalSituation
Association Source -> Destination	RiskDescription	HazardousEvent
Association Source -> Destination	HazardousEvent	OperationalSituation
Association Source -> Destination	HazardousEvent	OperationalSituation
Association Source -> Destination	HazardousEvent	ControllabilityReference
Association Source -> Destination	HazardousEvent	Hazard
Association Source -> Destination	HazardousEvent	MalfunctionPrototype

Attributes

Attribute	Notes	Default
consequences String	Provides the possibility to capture the consequences of a hazardous event. *Reference:* ISO 26262-3-7.4.2.2.4 (The consequences of hazardous events shall be identified.)	
hazardClassification ASILEnum	Based on Table 4 defined in ISO26262-3 the ASIL shall be allocated to the hazardous event based on the classification of the attributes • Controllability • Severity • Exposure	

Attribute	Notes	Default
exposure ExposureClassKind	**Exposure** is defined as the state of being in an **operational situation** that can be **hazardous** if coincident with the **failure mode** under analysis *Reference:* ISO 26262-1-1.37 ISO 26262-1-1.83 ISO 26262-1-1.57 ISO 26262-1-1.40	
severity SeverityClassKind	**Severity** is defined as the estimation of the extent of **harm** to one or more individuals that can occur in a potentially **hazardous** situation *Reference:* ISO 26262-1-1.120 ISO 26262-1-1.56 ISO 26262-1-1.57	
controllability ControllabilityClassKind	**Controllability** is defined as the ability to avoid a specified **harm** or damage through the timely reactions of the persons involved, possibly with support from **external measures** *NOTE:* 1. Persons involved can include the driver, passengers or persons in the vicinity of the vehicle's exterior. 2. The parameter C **in hazard analysis and risk assessment** represents the potential for controllability. *Reference:* ISO 26262-1.19 ISO 26262-1.56 ISO 26262-1.38 ISO 26262-1.58	

Class Item

Element Base Classes: **Identifiable**

Element Notes:

The item is, according to ISO 26262, a “system or array of systems to implement a function at the vehicle level, to which ISO 26262 is applied”. It is associated with the FunctionType since this class already allows decomposition of the function.

Note: E/E technology components in an other item are called "external measure".

Connections

Connector	Source	Target
Association Source -> Destination	Item	Item
Association Source -> Destination	Item	OperationalSituation
Aggregation Source -> Destination	MalfunctionPrototype	Item
Association Source -> Destination	Item	VehicleFeature
Generalization Source -> Destination	Item	Identifiable
Aggregation Source -> Destination	Item	HazardandRiskSafetyExtension
Association Source -> Destination	Item	Requirement
Aggregation Source -> Destination	OperatingMode	Item
Aggregation Source -> Destination	SafeState	Item
Aggregation Source -> Destination	ItemFunctionDescription	Item

Attributes

Attribute	Notes	Default
developmentCategory		

Attribute	Notes	Default
DevelopmentCategory		

Class ItemFunctionDescription

Element Base Classes: **Identifiable**

Element Notes:

Function of the item

Connections

Connector	Source	Target
Generalization Source -> Destination	ItemFunctionDescription	Identifiable
Association Source -> Destination	RiskDescription	ItemFunctionDescription
Aggregation Source -> Destination	ItemFunctionDescription	Item

Attributes

Attribute	Notes	Default
description String		

Class OperatingMode

Element Base Classes: **Identifiable**

Element Notes:

The Operating Mode is, according to ISO 26262, a “perceivable functional state of an item or element”. Therefore, it is associated with the item. Moreover, it is associated with the risk description since it describes a state of the item.

Reference:

ISO26262-1-1-81

Connections

Connector	Source	Target
Generalization Source -> Destination	OperatingMode	Identifiable
Aggregation Source -> Destination	OperatingMode	Item

Attributes

Attribute	Notes	Default
safetyRelated Boolean		

Class OperationalSituation

Element Base Classes: **TraceableSpecification**

Element Notes:

An operational situation is a scenario that may occur during a vehicles lifetime. Operational situations are formed by contributions of different actors, namely the driver (input of the driver via steering wheel, gas pedal, etc), the environment (e.g. road and lighting conditions), and other participants (pedestrians, other vehicles, etc).

The operational situations and operating modes in which an item's malfunctioning behavior will result in a hazardous event shall be described, both for cases when the vehicle is correctly used and when it is incorrectly used in a foreseeable way

Reference:

ISO 26262-3-7.4.2.1.1

ISO 26262-3-Annex.B-TableB.3.

Connections

Connector	Source	Target
Association Source -> Destination	OperationalSituation	Actor
Aggregation Source -> Destination	OperationalSituation	HazardandRiskSafetyExtension

Connector	Source	Target
Association Source -> Destination	Item	OperationalSituation
Association Source -> Destination	OperationalSituation	Environment
Association Source -> Destination	HazardousEvent	OperationalSituation
Association Source -> Destination	HazardousEvent	OperationalSituation
Association Source -> Destination	HazardousEvent	OperationalSituation
Association Source -> Destination	OperationalSituation	Actor
Generalization Source -> Destination	OperationalSituation	TraceableSpecification
Association Source -> Destination	OperationalSituation	Actor
Association Source -> Destination	OperationalSituation	Actor

Class RiskDescription

Element Base Classes: **TraceableSpecification**

Element Notes:

Description how the malfunction contributes to the hazardous event

Connections

Connector	Source	Target
Association Source -> Destination	RiskDescription	MalfunctionPrototype
Generalization Source -> Destination	RiskDescription	TraceableSpecification

Connector	Source	Target
Association Source -> Destination	RiskDescription	HazardousEvent
Association Source -> Destination	RiskDescription	ItemFunctionDescription
Aggregation Source -> Destination	RiskDescription	HazardandRiskSafetyExtension

Class SafeState

Element Base Classes: **Identifiable**

Element Notes:

The safe state is defined as an operating mode of an item without an unreasonable level of risk.

The safe state shall be reached within the allocated FaultTolerantTimeInterval.

In case that the safe state is defiened for a emergency operation it shall be reached within the EmergencyOperationTimeInterval.

Reference:

ISO 26262-1-1.102

Connections

Connector	Source	Target
Association Source -> Destination	WarningAndDegradationHandle	SafeState
Generalization Source -> Destination	SafeState	Identifiable
Aggregation Source -> Destination	SafeState	Item

Enumeration SeverityClassKind

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
S0		
S1		
S2		
S3		

Package Requirements

Package Notes:

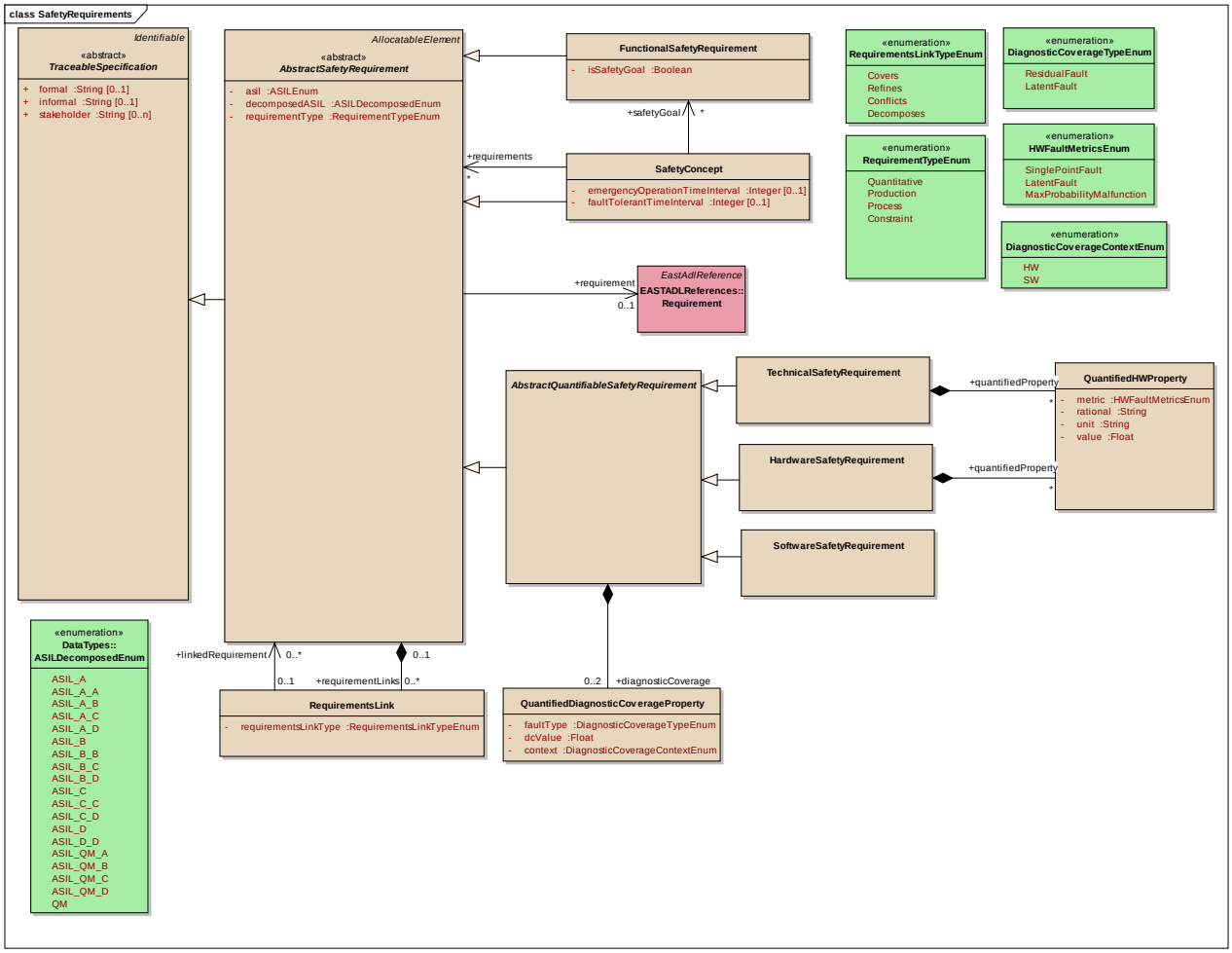


Figure 50: **SafetyRequirements** - *(Class diagram)*

Diagram Notes:

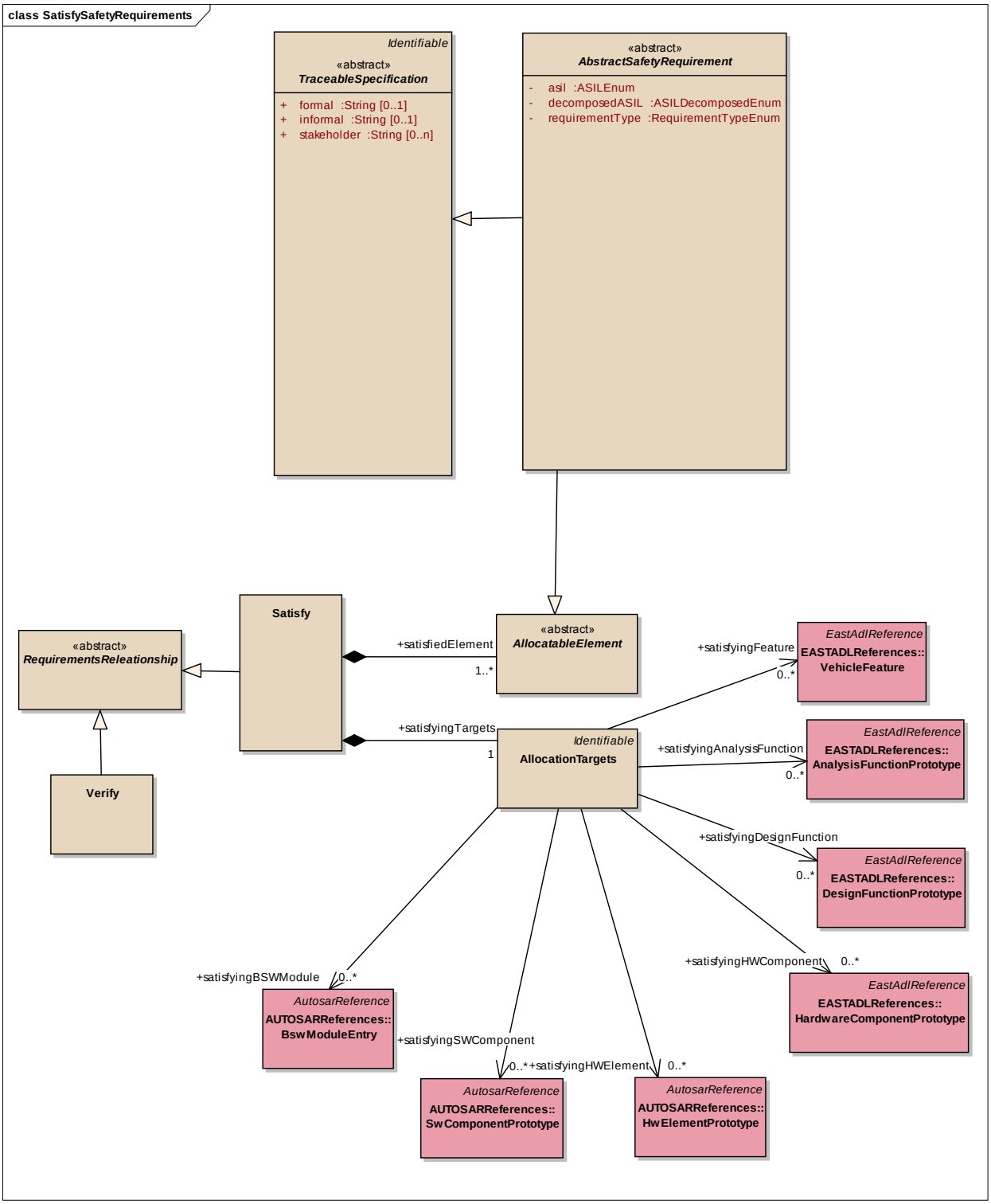


Figure 51: **SatisfySafetyRequirements** - *(Class diagram)*

Diagram Notes:

Class AbstractQuantifiableSafetyRequirement

Element Base Classes: **AbstractSafetyRequirement**

Element Notes:

AbstractQuantifiableSafetyRequirement is used to specify requirements which contain qualified properties. Examples of such properties are diagnostic coverage and hardware metrics.

Connections

Connector	Source	Target
Generalization Source -> Destination	SoftwareSafetyRequirement	AbstractQuantifiableSafetyRequirement
Generalization Source -> Destination	TechnicalSafetyRequirement	AbstractQuantifiableSafetyRequirement
Aggregation Source -> Destination	QuantifiedDiagnosticCoverageProperty	AbstractQuantifiableSafetyRequirement
Generalization Source -> Destination	AbstractQuantifiableSafetyRequirement	AbstractSafetyRequirement
Generalization Source -> Destination	HardwareSafetyRequirement	AbstractQuantifiableSafetyRequirement

Abstract AbstractSafetyRequirement

Element Base Classes: **AllocatableElement, TraceableSpecification**

Element Notes:

In the SAFE meta model AbstratcSafetyRequirement is used as the as the abstract superclass for

- SafetyGoals,
- FunctionalSafetyRequirements and
- TechnicalSafetyRequirements.

Connections

Connector	Source	Target
Generalization Source -> Destination	SafetyConcept	AbstractSafetyRequirement
Association Source -> Destination	RequirementsLink	AbstractSafetyRequirement
Generalization Source -> Destination	FunctionalSafetyRequirement	AbstractSafetyRequirement
Aggregation Source -> Destination	AbstractSafetyRequirement	Goal
Generalization Source -> Destination	Context	AbstractSafetyRequirement
Association Source -> Destination	AbstractSafetyRequirement	Requirement
Generalization Source -> Destination	Assumption	AbstractSafetyRequirement
Generalization Source -> Destination	AbstractSafetyRequirement	TraceableSpecification
Generalization Source -> Destination	AbstractQuantifiableSafetyRequirement	AbstractSafetyRequirement
Generalization Source -> Destination	AbstractSafetyRequirement	AllocatableElement
Aggregation Source -> Destination	Tactic	AbstractSafetyRequirement
Association Source -> Destination	SafetyConcept	AbstractSafetyRequirement
Aggregation Source -> Destination	RequirementsLink	AbstractSafetyRequirement

Attributes

Attribute	Notes	Default
asil ASILEnum	ASIL which has been assigned via decomposition	

Attribute	Notes	Default
decomposedASIL ASILDecomposedEnum		
requirementType RequirementTypeEnum		

Abstract AllocatableElement

Element Base Classes:

Element Notes:

Elements which can be allocated.

Connections

Connector	Source	Target
Aggregation Source -> Destination	AllocatableElement	Satisfy
Generalization Source -> Destination	AbstractSafetyRequirement	AllocatableElement

Class AllocationTargets

Element Base Classes: **Identifiable**

Element Notes:

Elements to which AllocatableElements can be allocated.

Connections

Connector	Source	Target
Association Source -> Destination	AllocationTargets	SwComponentPrototype
Association Source -> Destination	AllocationTargets	AnalysisFunctionPrototype

Connector	Source	Target
Association Source -> Destination	AllocationTargets	HardwareComponentPrototype
Generalization Source -> Destination	AllocationTargets	Identifiable
Aggregation Source -> Destination	AllocationTargets	Satisfy
Association Source -> Destination	AllocationTargets	BswModuleEntry
Association Source -> Destination	AllocationTargets	HwElementPrototype
Association Source -> Destination	AllocationTargets	VehicleFeature
Association Source -> Destination	AllocationTargets	DesignFunctionPrototype

Enumeration DiagnosticCoverageContextEnum

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
HW		
SW		

Enumeration DiagnosticCoverageTypeEnum

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
ResidualFault		
LatentFault		

Class FunctionalSafetyRequirement

Element Base Classes: **AbstractSafetyRequirement**

Element Notes:

FunctionalSafetyRequirements are used to specify

- implementation-independent safety behavior
- implementation-independent safety measures.

FunctionalSafetyRequirements shall contain safety-related attributes.

At least one functional safety requirement shall be specified for each safety goal.

FunctionalSafetyRequirements are allocated to the FunctionComponent of the item.

A safety goal shall be determined for each hazardous event with an ASIL evaluated in the hazard analysis. If similar safety goals are determined, these may be combined into one safety goal.

Constraint:

HasAsilDecomposed shall only be set if attribute redundancy is set true.

Reference:

ISO 26262-1-1.53

ISO 26262-3: 8.4.2.3

ISO 26262-3-7.4.4.3

Connections

Connector	Source	Target
Aggregation Source -> Destination	FunctionalSafetyRequirement	HazardandRiskSafetyExtension
Association Source -> Destination	HazardousEvent	FunctionalSafetyRequirement
Generalization Source -> Destination	FunctionalSafetyRequirement	AbstractSafetyRequirement
Generalization Source -> Destination	WarningAndDegradation	FunctionalSafetyRequirement
Aggregation Source -> Destination	FunctionalSafetyRequirement	FunctionalSafetyExtension
Association Source -> Destination	SafetyConcept	FunctionalSafetyRequirement

Attributes

Attribute	Notes	Default
isSafetyGoal Boolean		

Enumeration HWFaultMetricsEnum

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
SinglePointFault		
LatentFault		
MaxProbabilityMalfunction		

Attribute	Notes	Default

Class HardwareSafetyRequirement

Element Base Classes: **AbstractQuantifiableSafetyRequirement**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	QuantifiedHWProperty	HardwareSafetyRequirement
Generalization Source -> Destination	HardwareSafetyRequirement	AbstractQuantifiableSafetyRequirement
Aggregation Source -> Destination	HardwareSafetyRequirement	ImplementationSafetyExtension

Class QuantifiedDiagnosticCoverageProperty

Element Base Classes:

Element Notes:

QuantifiedHWDiagnosticCoverageProperty defines the diagnostic coverage information for a given AbstractQuantifiableSafetyRequirement.

Connections

Connector	Source	Target
Aggregation Source -> Destination	QuantifiedDiagnosticCoverageProperty	AbstractQuantifiableSafetyRequirement

Attributes

Attribute	Notes	Default
faultType DiagnosticCoverageTypeEnum		
dcValue Float		
context DiagnosticCoverageContextEnum		

Class QuantifiedHWProperty

Element Base Classes:

Element Notes:

QuantifiedHWProperty specifies the quantitative properties for a given AbstractQuantifiableSafetyRequirement

Connections

Connector	Source	Target
Aggregation Source -> Destination	QuantifiedHWProperty	HardwareSafetyRequirement
Aggregation Source -> Destination	QuantifiedHWProperty	TechnicalSafetyRequirement

Attributes

Attribute	Notes	Default
metric HWFaultMetricsEnum		
rational String		
unit String		
value Float		

Enumeration RequirementTypeEnum

Element Base Classes:

Element Notes:

This enumeration specifies the kind of safety requirement

Attributes

Attribute	Notes	Default
Quantitative		
Production		
Process		
Constraint		

Class RequirementsLink

Element Base Classes:

Element Notes:

RequirementsLink is owned by requirement. It is used to establish links between requirements. The type of the link is determined by the RequirementsLinkType.

Connections

Connector	Source	Target
Association Source -> Destination	RequirementsLink	AbstractSafetyRequirement
Aggregation Source -> Destination	RequirementsLink	AbstractSafetyRequirement

Attributes

Attribute	Notes	Default
requirementsLinkType RequirementsLinkTypeEnum	The type of the link between requirements is determined by the RequirementsLinkType.	

Enumeration RequirementsLinkTypeEnum

Element Base Classes:

Element Notes:

RequirementsLinkType defines the type of the link between requirements.

Covers: The requirement which owns a link of this type covers the linked requirements

Refines: The requirement which owns a link of this type refines the linked requirements

Conflicts: The requirement which owns a link of this type conflicts with the linked requirements

Decomposes: The requirement which owns a link of this type decomposes the linked requirements

Attributes

Attribute	Notes	Default
Covers		
Refines		
Conflicts		
Decomposes		

Abstract RequirementsReleationship

Element Base Classes:

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	Satisfy	RequirementsReleationship
Generalization Source -> Destination	Verify	RequirementsReleationship
Aggregation Source -> Destination	RequirementsReleationship	SafetyExtension

Class SafetyConcept

Element Base Classes: **AbstractSafetyRequirement**

Element Notes:

This container provides the possibility to group different requirements belonging to the realization of a safety concept. For example, a functional safety concept consisting of requirements specifying what kind of error detection must be done and what kind of handling for detected errors must be done.

Furthermore, the container provides two attributes relevant for safety concepts, fault tolerant time interval (FTTI) and emergency operation time interval. These attributes can be used to derive the time quantum allocated to each requirement belonging to the concept.

Connections

Connector	Source	Target
Generalization Source -> Destination	SafetyConcept	AbstractSafetyRequirement
Aggregation Source -> Destination	SafetyConcept	MultiLevelSafetyExtension
Association Source -> Destination	SafetyConcept	FunctionalSafetyRequirement
Association Source -> Destination	SafetyConcept	AbstractSafetyRequirement

Attributes

Attribute	Notes	Default
emergencyOperationTimeInterval Integer		
faultTolerantTimeInterval Integer		

Class Satisfy

Element Base Classes: **RequirementsReleationship**

Element Notes:

Satisfy allows allocating requirements to elements of the preliminary architecture as well as HW-Elements and SW-Elements

Connections

Connector	Source	Target
Aggregation Source -> Destination	AllocatableElement	Satisfy
Generalization Source -> Destination	Satisfy	RequirementsReleationship
Aggregation Source -> Destination	AllocationTargets	Satisfy

Class SoftwareSafetyRequirement

Element Base Classes: **AbstractQuantifiableSafetyRequirement**

Element Notes:

Represents the refinement of technical safety requirements namely software safety requirements (SSR). Every SSR is realized in the form of a safety mechanism (partially automatically generated) which traces back to its originating SSR.

Connections

Connector	Source	Target
Generalization Source -> Destination	CHROMOSOMEHealthMonitorNotification	SoftwareSafetyRequirement
Generalization Source -> Destination	SoftwareSafetyRequirement	AbstractQuantifiableSafetyRequirement
Generalization Source -> Destination	CpuSelfTest	SoftwareSafetyRequirement
Generalization Source -> Destination	Filter	SoftwareSafetyRequirement
Generalization Source -> Destination	HeartbeatSender	SoftwareSafetyRequirement
Generalization Source -> Destination	ActuatorMonitor	SoftwareSafetyRequirement
Aggregation Source -> Destination	SoftwareSafetyRequirement	ImplementationSafetyExtension
Generalization Source -> Destination	HeartbeatReceiver	SoftwareSafetyRequirement
Generalization Source -> Destination	Heartbeat	SoftwareSafetyRequirement
Association Source -> Destination	CodeGenerationConfiguration	SoftwareSafetyRequirement
Generalization Source -> Destination	HealthMonitor	SoftwareSafetyRequirement
Generalization Source -> Destination	GradientCheck	SoftwareSafetyRequirement
Generalization Source -> Destination	Voter	SoftwareSafetyRequirement
Generalization Source -> Destination	CRC	SoftwareSafetyRequirement
Generalization Source -> Destination	Comparison	SoftwareSafetyRequirement
Generalization Source -> Destination	AlivenessMonitor	SoftwareSafetyRequirement

Connector	Source	Target
Generalization Source -> Destination	MemorySelfTest	SoftwareSafetyRequirement
Generalization Source -> Destination	ControlFlowMonitor	SoftwareSafetyRequirement
Generalization Source -> Destination	ContextRangeCheck	SoftwareSafetyRequirement

Class TechnicalSafetyRequirement

Element Base Classes: **AbstractQuantifiableSafetyRequirement**

Element Notes:

TechnicalSafetyRequirements are used to specify the implementation of the associated FunctionComponent . They are derived by the allocated FunctionalSafetyRequirements.

TechnicalSafetyRequirements include the specification of safety-related failure mitigation.

The technical safety requirements specification refines the functional safety concept, considering both the functional concept and the preliminary architectural assumptions

The technical safety requirements are allocated to hardware and software, and, if applicable, on other technologies

Reference:

ISO26262-1-1.133

ISO26262-4- 6.4.9

Connections

Connector	Source	Target
Aggregation Source -> Destination	TechnicalSafetyRequirement	TechnicalSafetyExtension
Generalization Source -> Destination	TechnicalSafetyRequirement	AbstractQuantifiableSafetyRequirement

Connector	Source	Target
Aggregation Source -> Destination	CodeGenerationConfiguration	TechnicalSafetyRequirement
Aggregation Source -> Destination	QuantifiedHWProperty	TechnicalSafetyRequirement

Abstract TraceableSpecification

Element Base Classes: **Identifiable**

Element Notes:

Abstract superclass for elements which can be traces (e.g. Requirements)

Connections

Connector	Source	Target
Generalization Source -> Destination	RiskDescription	TraceableSpecification
Generalization Source -> Destination	TraceableSpecification	Identifiable
Generalization Source -> Destination	ErrorModelType	TraceableSpecification
Generalization Source -> Destination	AbstractSafetyRequirement	TraceableSpecification
Generalization Source -> Destination	OperationalSituation	TraceableSpecification

Attributes

Attribute	Notes	Default
formal String		
informal String	Informal description of the element	
stakeholder String	Stakeholders for the element	

Class Verify

Element Base Classes: **RequirementsReleationship**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	Verify	RequirementsReleationship

Package FunctionalSafetyRequirements

Package Notes:

Package WarningAndDegradation

Package Notes:

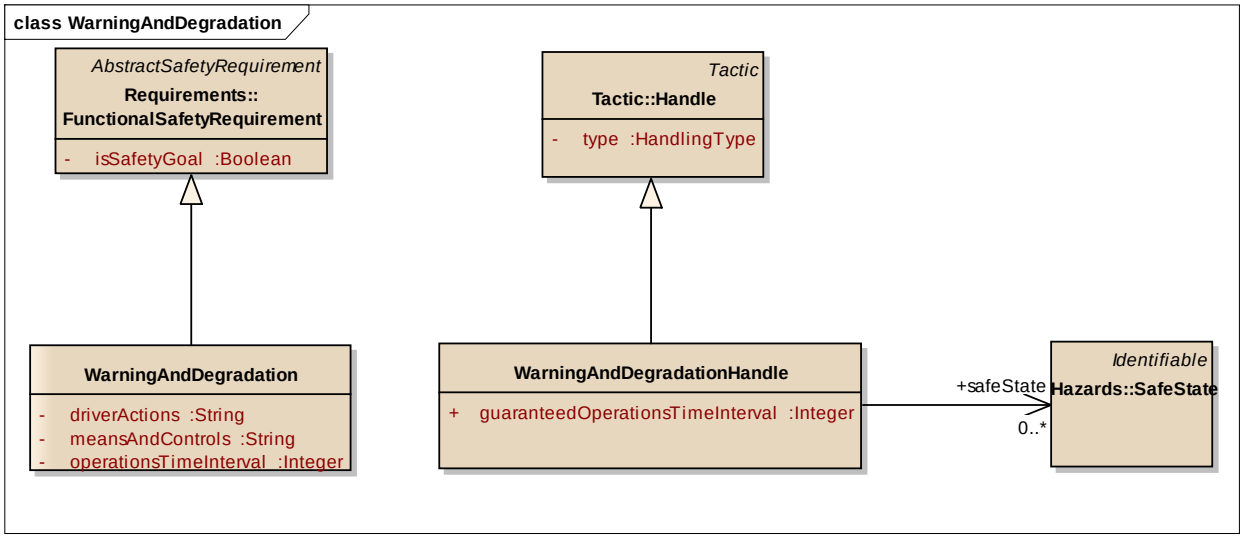


Figure 52: **WarningAndDegradation** - *(Class diagram)*

Diagram Notes:

Class WarningAndDegradation

Element Base Classes: **FunctionalSafetyRequirement**

Element Notes:

The warning- and degradation concept is a specification of how to alert the driver of potentially reduced functionality and of how to provide this reduced functionality to reach a safe state.

The warning and degradation concept shall be specified by FunctionalSafetyRequirements

The warning- and degradation-concept describes the system transition of the system in case of occurrence of a safety relevant failure to the safe state that is defined for that failure.

The warn- and degradation-concept also contains functional safety requirements. The functional safety requirements of the warn- and degradation-concept shall be allocated to the safe state that shall be reached.

The functional safety requirements of the warn- and degradation-concept shall be classified with the same ASIL as the allocated safe state.

The exit condition of the safe state back to the normal operation mode is also part of the warn-and degradation-concept.

Reference:

ISO 26262-3-8.4.2.5

Connections

Connector	Source	Target
Generalization Source -> Destination	WarningAndDegradation	FunctionalSafetyRequirement

Attributes

Attribute	Notes	Default
driverActions String		
meansAndControls String		
operationsTimeInterval Integer		

Class WarningAndDegradationHandle

Element Base Classes: **Handle**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	WarningAndDegradationHandle	SafeState
Generalization Source -> Destination	WarningAndDegradationHandle	Handle

Attributes

Attribute	Notes	Default
guaranteedOperationsTimeInterval Integer		

Package SafetyCase

Package Notes:

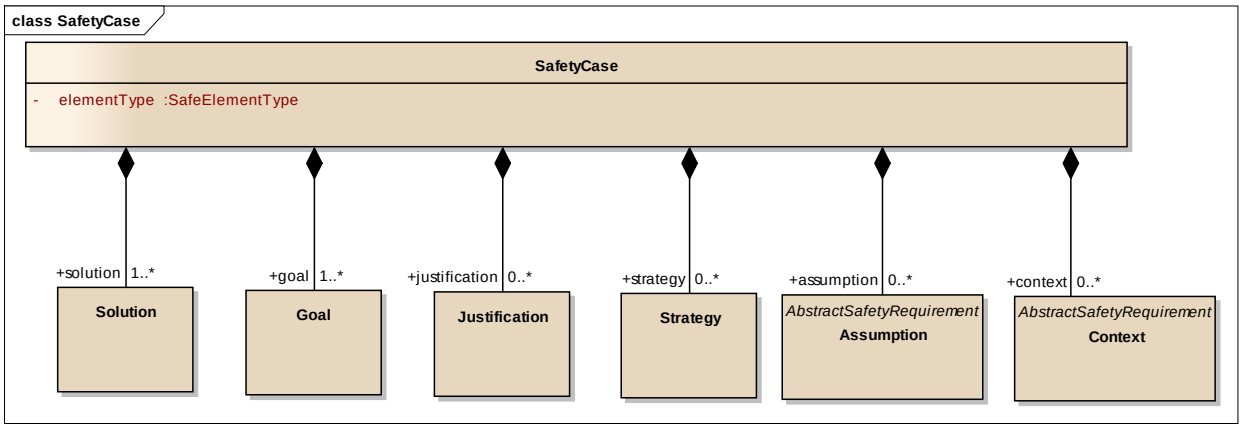


Figure 53: **SafetyCase** - *(Class diagram)*

Diagram Notes:

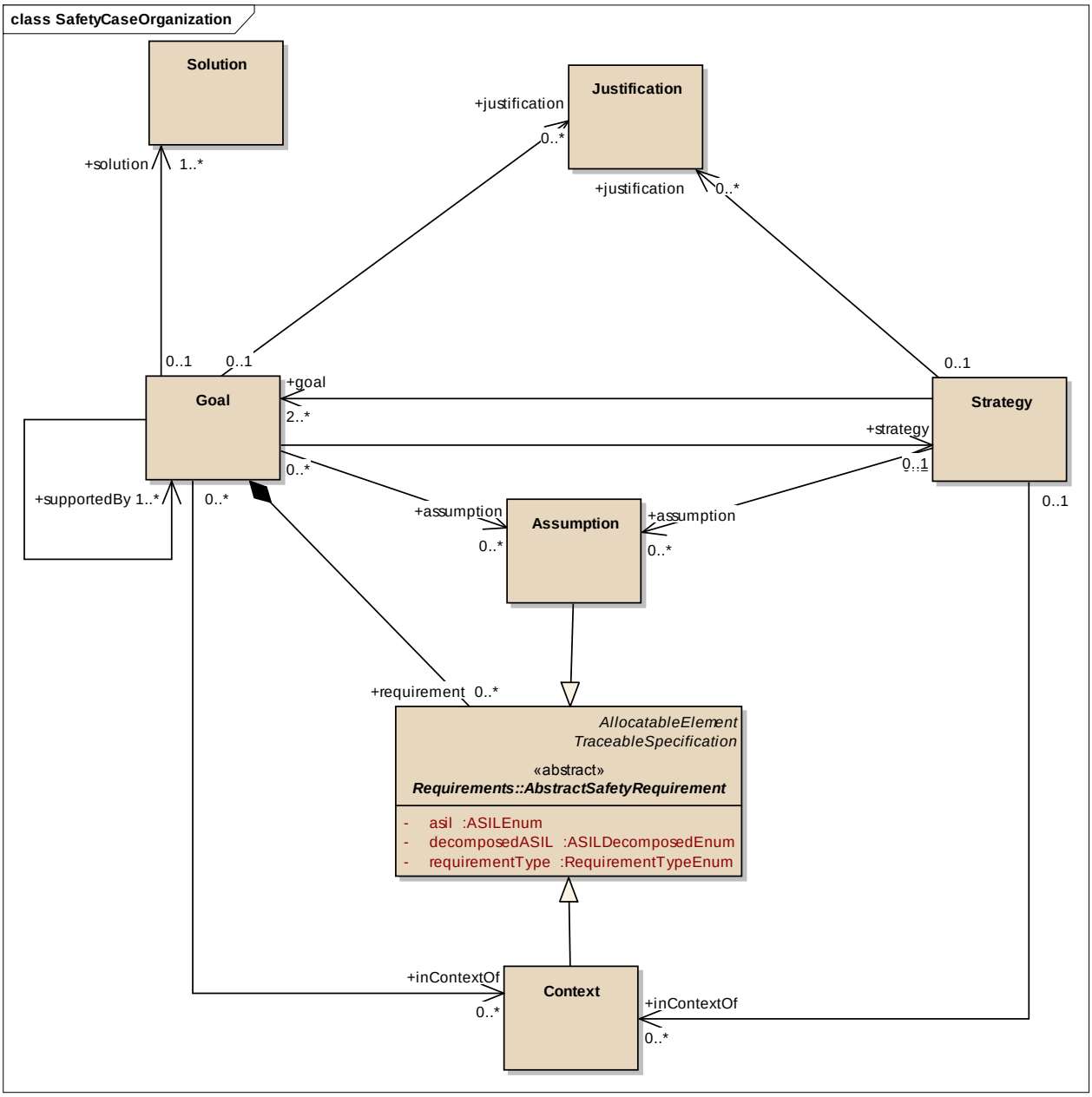


Figure 54: **SafetyCaseOrganization** - *(Class diagram)*

Diagram Notes:

Class Assumption

Element Base Classes: **AbstractSafetyRequirement**

Element Notes:

- An assumption is an intentionally unsubstantiated statement. The scope of an assumption is the entire

argument. Having connected an assumption to a goal, the assumption is taken to be connected to the entirety of the argument supporting this goal.
- Therefore, it is not necessary to restate the assumption in the supporting argument.
- rendered as an oval with the letter 'A' at the bottom-right.

Connections

Connector	Source	Target
Association Source -> Destination	Goal	Assumption
Aggregation Source -> Destination	Assumption	SafetyCase
Generalization Source -> Destination	Assumption	AbstractSafetyRequirement
Association Source -> Destination	Strategy	Assumption

Class Context

Element Base Classes: **AbstractSafetyRequirement**

Element Notes:

- Claims can only be asserted to be true in a specified context. Context elements are used to make this relationship clear.
- A ***context***, rendered as an oblong rectangle with rounded out sides, presents a contextual artifact. This can be a reference to contextual information, or a statement.
- Where used, contexts define or constrain the scope over which the claim is made.

Connections

Connector	Source	Target
Association Source -> Destination	Goal	Context
Association Source -> Destination	Strategy	Context

Connector	Source	Target
Generalization Source -> Destination	Context	AbstractSafetyRequirement
Aggregation Source -> Destination	Context	SafetyCase

Class Goal

Element Base Classes:

Element Notes:

- rendered as a rectangle,
- presents a claim forming part of the argument.
- One or more sub-goals may be declared for a given goal. This structure then asserts that if the claims presented in the sub-goals are true, this is sufficient to establish that the claim in the main goal is true.

Connections

Connector	Source	Target
Association Source -> Destination	Goal	Justification
Association Source -> Destination	Goal	Assumption
Aggregation Source -> Destination	AbstractSafetyRequirement	Goal
Association Source -> Destination	Goal	Context
Association Source -> Destination	Goal	Solution
Association Source -> Destination	Goal	Goal
Association Source -> Destination	Strategy	Goal
Aggregation Source -> Destination	Goal	SafetyCase

Connector	Source	Target
Association Source -> Destination	Goal	Strategy

Class Justification

Element Base Classes:

Element Notes:

- rendered as an oval with the letter 'J' at the bottom-right,
- presents a statement of rationale,
- and does not alter the meaning of the claim made in the goal, but provides rationale for its inclusion or its phrasing.
- Should an equivalent justification be required elsewhere in the argument, it will need to be re-stated or re-linked.

Connections

Connector	Source	Target
Association Source -> Destination	Goal	Justification
Aggregation Source -> Destination	Justification	SafetyCase
Association Source -> Destination	Strategy	Justification

Class SafetyCase

Element Base Classes:

Element Notes:

Defines safety case as defined in ISO26262.

Connections

Connector	Source	Target
Aggregation Source -> Destination	Justification	SafetyCase
Aggregation Source -> Destination	Solution	SafetyCase
Aggregation Source -> Destination	Assumption	SafetyCase
Aggregation Source -> Destination	SafetyCase	MultiLevelSafetyExtension
Aggregation Source -> Destination	Context	SafetyCase
Aggregation Source -> Destination	Goal	SafetyCase
Aggregation Source -> Destination	Strategy	SafetyCase

Attributes

Attribute	Notes	Default
elementType SafeElementType		

Class Solution

Element Base Classes:

Element Notes:

- A ***solution***, rendered as a circle, presents a reference to an evidence item or items.
- Multiple solutions may satisfy a goal.
- Multiple goals may be satisfied by one solution.

Connections

Connector	Source	Target
Generalization Source -> Destination	DesignSpecification	Solution

Connector	Source	Target
Generalization Source -> Destination	AnalysisReport	Solution
Association Source -> Destination	Goal	Solution
Aggregation Source -> Destination	Solution	SafetyCase
Generalization Source -> Destination	TestProtocol	Solution

Class Strategy

Element Base Classes:

Element Notes:

- rendered as a parallelogram,
- describe the nature of the inference that exists between a goal and its supporting goal(s)
- and are used to describe the nature of the inference which is asserted as existing between sub-goals and the parent goal.

Connections

Connector	Source	Target
Association Source -> Destination	Strategy	Justification
Association Source -> Destination	Strategy	Context
Association Source -> Destination	Strategy	Goal
Association Source -> Destination	Goal	Strategy
Association Source -> Destination	Strategy	Assumption
Aggregation Source -> Destination	Strategy	SafetyCase

Connector	Source	Target

Package Solution

Package Notes:

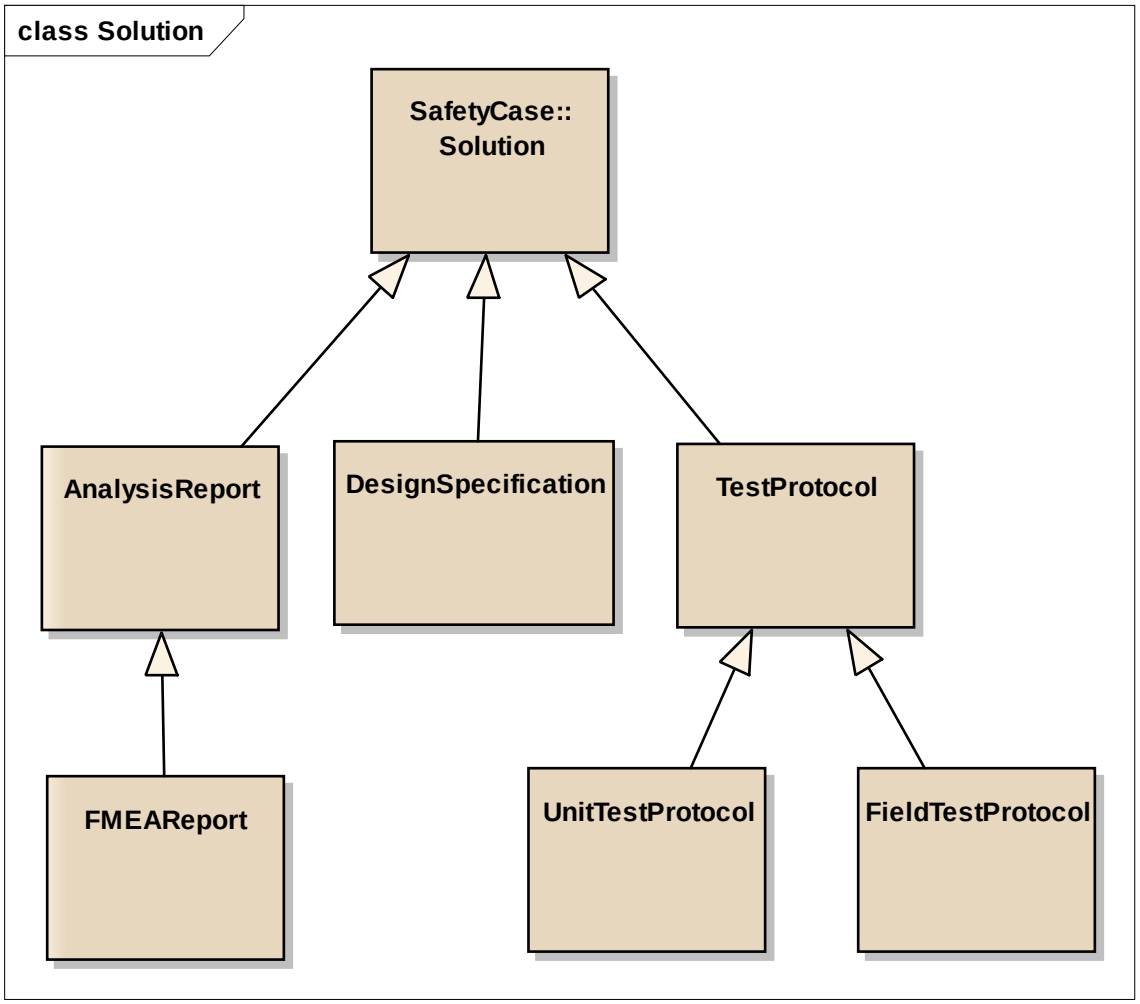


Figure 55: **Solution** - *(Class diagram)*

Diagram Notes:

Class AnalysisReport

Element Base Classes: **Solution**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	AnalysisReport	Solution
Generalization Source -> Destination	FMEAReport	AnalysisReport

Class DesignSpecification

Element Base Classes: **Solution**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	DesignSpecification	Solution

Class FMEAReport

Element Base Classes: **AnalysisReport**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FMEAReport	AnalysisReport

Class FieldTestProtocol

Element Base Classes: **TestProtocol**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FieldTestProtocol	TestProtocol

Class TestProtocol

Element Base Classes: **Solution**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FieldTestProtocol	TestProtocol
Generalization Source -> Destination	TestProtocol	Solution
Generalization Source -> Destination	UnitTestProtocol	TestProtocol

Class UnitTestProtocol

Element Base Classes: **TestProtocol**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	UnitTestProtocol	TestProtocol

Package SoftwareSafetyRequirements

Package Notes:

Package ControlFlowMonitor

Package Notes:

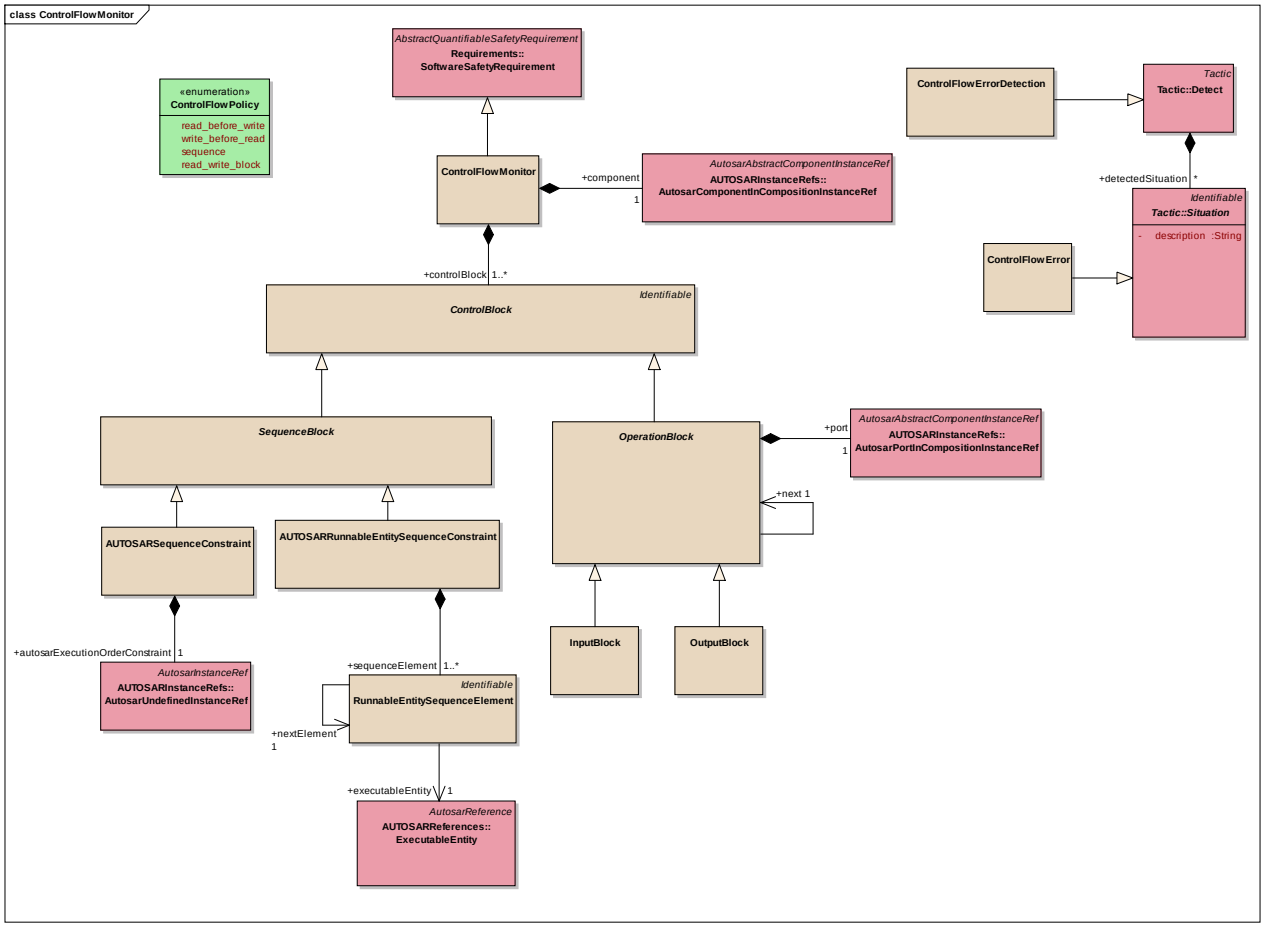


Figure 56: **ControlFlowMonitor** - *(Class diagram)*

Diagram Notes:

Class AUTOSARRunnableEntitySequenceConstraint

Element Base Classes: **SequenceBlock**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	RunnableEntitySequenceElement	AUTOSARRunnableEntitySequenceConstraint

Connector	Source	Target
Generalization Source -> Destination	AUTOSARRunnableEntitySequenceConstraint	SequenceBlock

Class AUTOSARSequenceConstraint

Element Base Classes: **SequenceBlock**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarUndefinedInstanceRef	AUTOSARSequenceConstraint
Generalization Source -> Destination	AUTOSARSequenceConstraint	SequenceBlock

Class ControlBlock

Element Base Classes: **Identifiable**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	ControlBlock	ControlFlowMonitor
Generalization Source -> Destination	OperationBlock	ControlBlock
Generalization	ControlBlock	Identifiable

Connector	Source	Target
Source -> Destination		
Generalization Source -> Destination	SequenceBlock	ControlBlock

Class ControlFlowError

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ControlFlowError	Situation

Class ControlFlowErrorDetection

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ControlFlowErrorDetection	Detect

Class ControlFlowMonitor

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

Connections

Connector	Source	Target
Aggregation Source -> Destination	ControlBlock	ControlFlowMonitor
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	ControlFlowMonitor
Generalization Source -> Destination	ControlFlowMonitor	SoftwareSafetyRequirement

Enumeration ControlFlowPolicy

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
read_before_write		
write_before_read		
sequence		
read_write_block		

Class InputBlock

Element Base Classes: **OperationBlock**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	InputBlock	OperationBlock

Class OperationBlock

Element Base Classes: **ControlBlock**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	OperationBlock	OperationBlock
Generalization Source -> Destination	InputBlock	OperationBlock
Generalization Source -> Destination	OperationBlock	ControlBlock
Aggregation Source -> Destination	AutosarPortInCompositionInstanceRef	OperationBlock
Generalization Source -> Destination	OutputBlock	OperationBlock

Class OutputBlock

Element Base Classes: **OperationBlock**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	OutputBlock	OperationBlock

Class RunnableEntitySequenceElement

Element Base Classes: **Identifiable**

Element Notes:

Connections

Connector	Source	Target
Association Source -> Destination	RunnableEntitySequenceElement	RunnableEntitySequenceElement
Generalization Source -> Destination	RunnableEntitySequenceElement	Identifiable
Aggregation Source -> Destination	RunnableEntitySequenceElement	AUTOSARRunnableEntitySequenceConstraint
Association Source -> Destination	RunnableEntitySequenceElement	ExecutableEntity

Class SequenceBlock

Element Base Classes: **ControlBlock**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	AUTOSARSequenceConstraint	SequenceBlock
Generalization Source -> Destination	AUTOSARRunnableEntitySequenceConstraint	SequenceBlock
Generalization Source -> Destination	SequenceBlock	ControlBlock

Package AlivenessMonitor

Package Notes:

This package groups the elements related to the software safety mechanism Aliveness Monitor

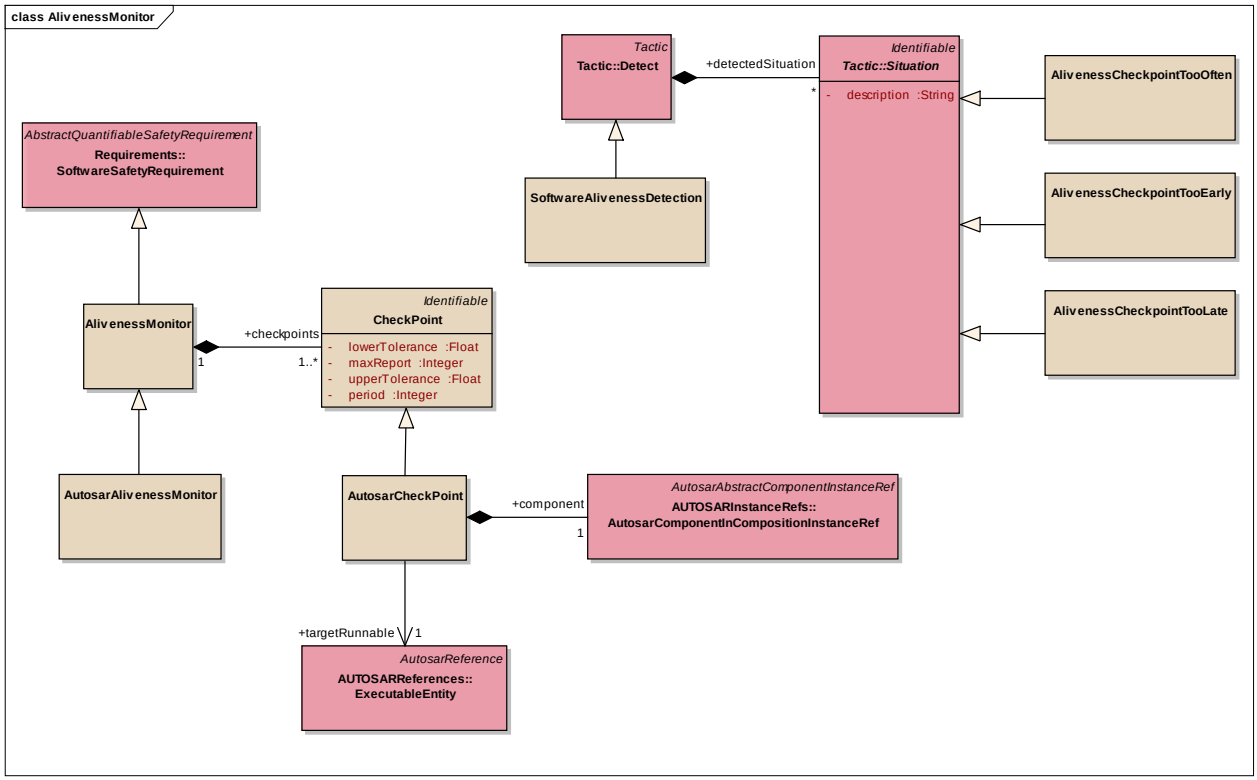


Figure 57: **AlivenessMonitor** - *(Class diagram)*

Diagram Notes:

Class AlivenessCheckpointTooEarly

Element Base Classes: **Situation**

Element Notes:

Represents the situation in which an aliveness monitor receives the notification of a checkpoint reached before the expected point in time.

Connections

Connector	Source	Target
Generalization Source -> Destination	AlivenessCheckpointTooEarly	Situation

Class AlivenessCheckpointTooLate

Element Base Classes: **Situation**

Element Notes:

Represents the situation in which an aliveness monitor receives the notification of a checkpoint reached after the expected point in time.

Connections

Connector	Source	Target
Generalization Source -> Destination	AlivenessCheckpointTooLate	Situation

Class AlivenessCheckpointTooOften

Element Base Classes: **Situation**

Element Notes:

Represents the situation in which an aliveness monitor receives too many notifications that a checkpoint has been reached.

Connections

Connector	Source	Target
Generalization Source -> Destination	AlivenessCheckpointTooOften	Situation

Class AlivenessMonitor

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

The aliveness monitor contains the set of mapped checkpoint definitions to executable entities. The set of checkpoints contained in the mechanism defines the scope of a specific mechanism instance.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarAlivenessMonitor	AlivenessMonitor
Aggregation Source -> Destination	CheckPoint	AlivenessMonitor
Generalization Source -> Destination	AlivenessMonitor	SoftwareSafetyRequirement

Class AutosarAlivenessMonitor

Element Base Classes: **AlivenessMonitor**

Element Notes:

AUTOSAR specific aliveness monitor meta class

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarAlivenessMonitor	AlivenessMonitor

Class AutosarCheckPoint

Element Base Classes: **CheckPoint**

Element Notes:

AUTOSAR specific checkpoint

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarCheckPoint	CheckPoint
Association Source -> Destination	AutosarCheckPoint	ExecutableEntity

Connector	Source	Target
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	AutosarCheckPoint

Class CheckPoint

Element Base Classes: **Identifiable**

Element Notes:

A checkpoint provides the attributes necessary for checking if a given executable unity is being triggered correctly by the system.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarCheckPoint	CheckPoint
Aggregation Source -> Destination	CheckPoint	AlivenessMonitor
Generalization Source -> Destination	CheckPoint	Identifiable

Attributes

Attribute	Notes	Default
lowerTolerance Float		
maxReport Integer		
upperTolerance Float		
period Integer		

Class SoftwareAlivenessDetection

Element Base Classes: **Detect**

Element Notes:

Detection of the aliveness property through software

Connections

Connector	Source	Target
Generalization Source -> Destination	SoftwareAlivenessDetection	Detect

Package ActuatorMonitor

Package Notes:

This package groups the elements related to the software safety mechanism Actuator Monitor

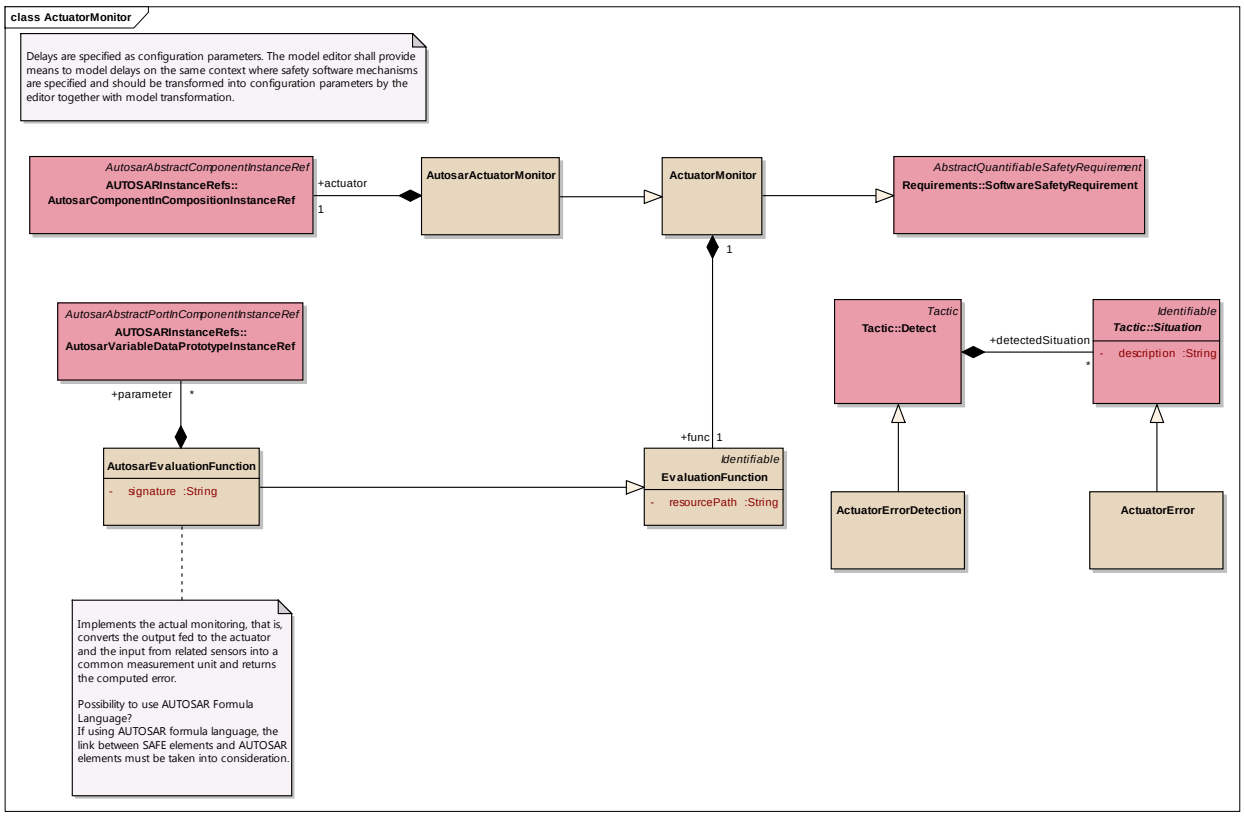


Figure 58: **ActuatorMonitor** - *(Class diagram)*

Diagram Notes:

Class ActuatorError

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ActuatorError	Situation

Class ActuatorErrorDetection

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ActuatorErrorDetection	Detect

Class ActuatorMonitor

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

This meta-class represents the actuator monitor requirement. It extends the meta-class SSR from SAFE meta model.

To define an actuator monitor requirement the engineer has to reference the monitored actuator through the ActuatorMonitor attribute actuator and the inputs through which the monitoring occurs, using the attribute sensors. Furthermore, the engineer has to specify an evaluation function which maps input from sensors to the provided actuator output.

Connections

Connector	Source	Target
Generalization Source -> Destination	ActuatorMonitor	SoftwareSafetyRequirement
Generalization Source -> Destination	AutosarActuatorMonitor	ActuatorMonitor
Aggregation Source -> Destination	EvaluationFunction	ActuatorMonitor

Class AutosarActuatorMonitor

Element Base Classes: **ActuatorMonitor**

Element Notes:

Specific meta class for the AUTOSAR actuator monitor software safety requirement.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarActuatorMonitor	ActuatorMonitor
Aggregation Source -> Destination	AutosarComponentInCompositionInstanceRef	AutosarActuatorMonitor

Class AutosarEvaluationFunction

Element Base Classes: **EvaluationFunction**

Element Notes:

Evaluation function defined by the engineer to relate the input from sensors to the output value provided by the actuator controller.

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarEvaluationFunction
NoteLink	<anonymous>	AutosarEvaluationFunction
Generalization Source -> Destination	AutosarEvaluationFunction	EvaluationFunction

Attributes

Attribute	Notes	Default
signature String		

Class EvaluationFunction

Element Base Classes: **Identifiable**

Element Notes:

Abstract element representing an user defined evaluation function for deciding if the actuator behaves correctly.

Connections

Connector	Source	Target
Generalization Source -> Destination	EvaluationFunction	Identifiable
Generalization Source -> Destination	AutosarEvaluationFunction	EvaluationFunction
Aggregation Source -> Destination	EvaluationFunction	ActuatorMonitor

Attributes

Attribute	Notes	Default
resourcePath String		

Package CRCChecksum

Package Notes:

This package groups the elements related to the software safety mechanism CRC Checksum

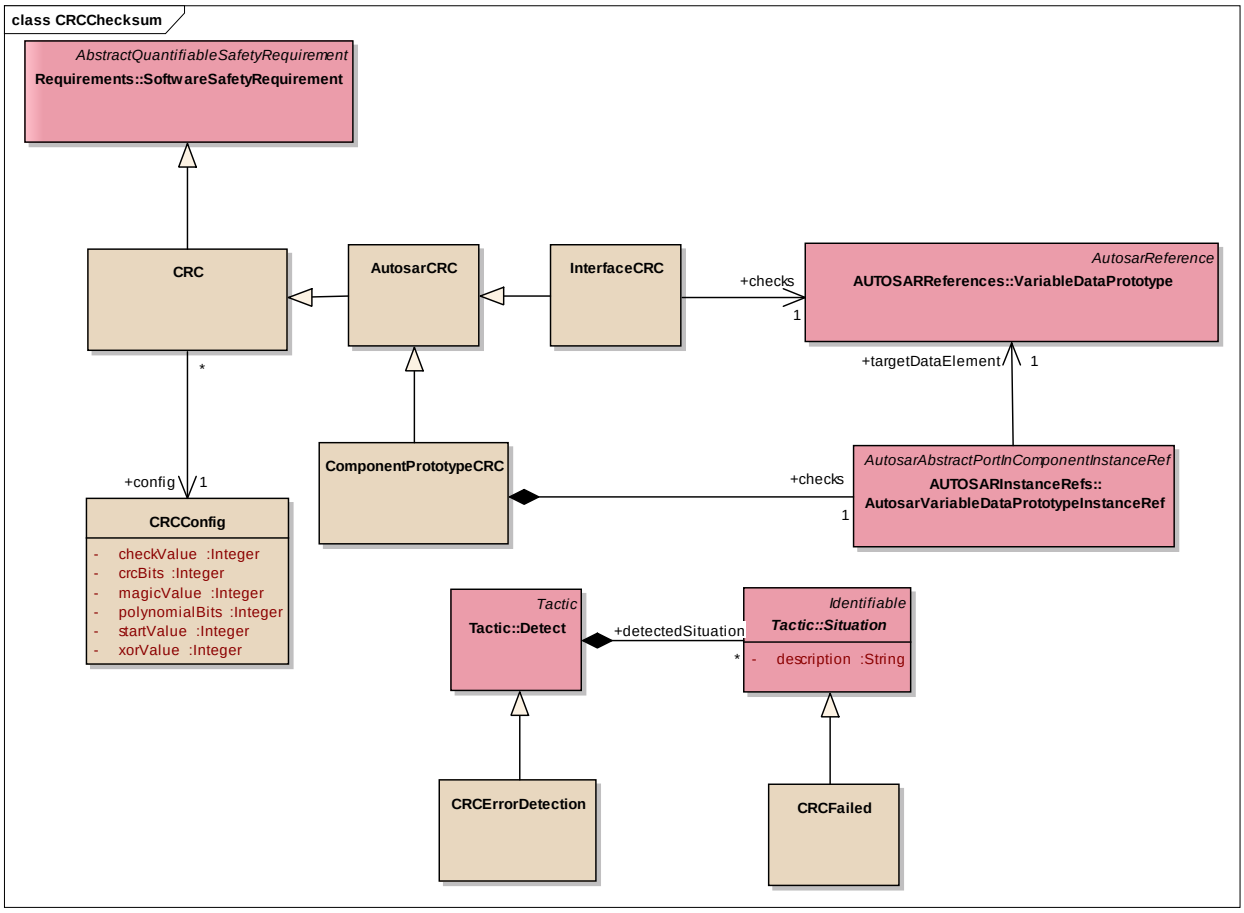


Figure 59: **CRCChecksum** - *(Class diagram)*

Diagram Notes:

Class AutosarCRC

Element Base Classes: **CRC**

Element Notes:

AUTOSAR specific CRC requirement.

Connections

Connector	Source	Target
Generalization Source -> Destination	InterfaceCRC	AutosarCRC
Generalization Source -> Destination	ComponentPrototypeCRC	AutosarCRC
Generalization Source -> Destination	AutosarCRC	CRC

Class CRC

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

This meta-class represents the requirement cyclic-redundancy-check.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarCRC	CRC
Association Source -> Destination	CRC	CRCConfig
Generalization Source -> Destination	CRC	SoftwareSafetyRequirement

Class CRCConfig

Element Base Classes:

Element Notes:

This meta-class provides the means for configuring the CRC requirement. The attributes are used by both specializations of the abstract version of the SSR.

Connections

Connector	Source	Target
Association Source -> Destination	CRC	CRCConfig

Attributes

Attribute	Notes	Default
checkValue Integer		
crcBits Integer		
magicValue Integer		
polynomialBits Integer		
startValue Integer		
xorValue Integer		

Class CRCErrorDetection

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	CRCErrorDetection	Detect

Connector	Source	Target

Class CRCFailed

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	CRCFailed	Situation

Class ComponentPrototypeCRC

Element Base Classes: **AutosarCRC**

Element Notes:

This meta-class is the specialization of the abstract SSR CRC which is deployed for a specific instance of a component providing a given interface.

Connections

Connector	Source	Target
Generalization Source -> Destination	ComponentPrototypeCRC	AutosarCRC
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	ComponentPrototypeCRC

Class InterfaceCRC

Element Base Classes: **AutosarCRC**

Element Notes:

This meta-class is the specialization of the CRC SSR deployed as the property of data elements of a given interface. This means the CRC is a property of the interface and not of a specific instance (realization) of the interface.

Connections

Connector	Source	Target
Generalization Source -> Destination	InterfaceCRC	AutosarCRC
Association Source -> Destination	InterfaceCRC	VariableDataPrototype

Package Comparison

Package Notes:

This package groups the elements related to the software safety mechanism comparison

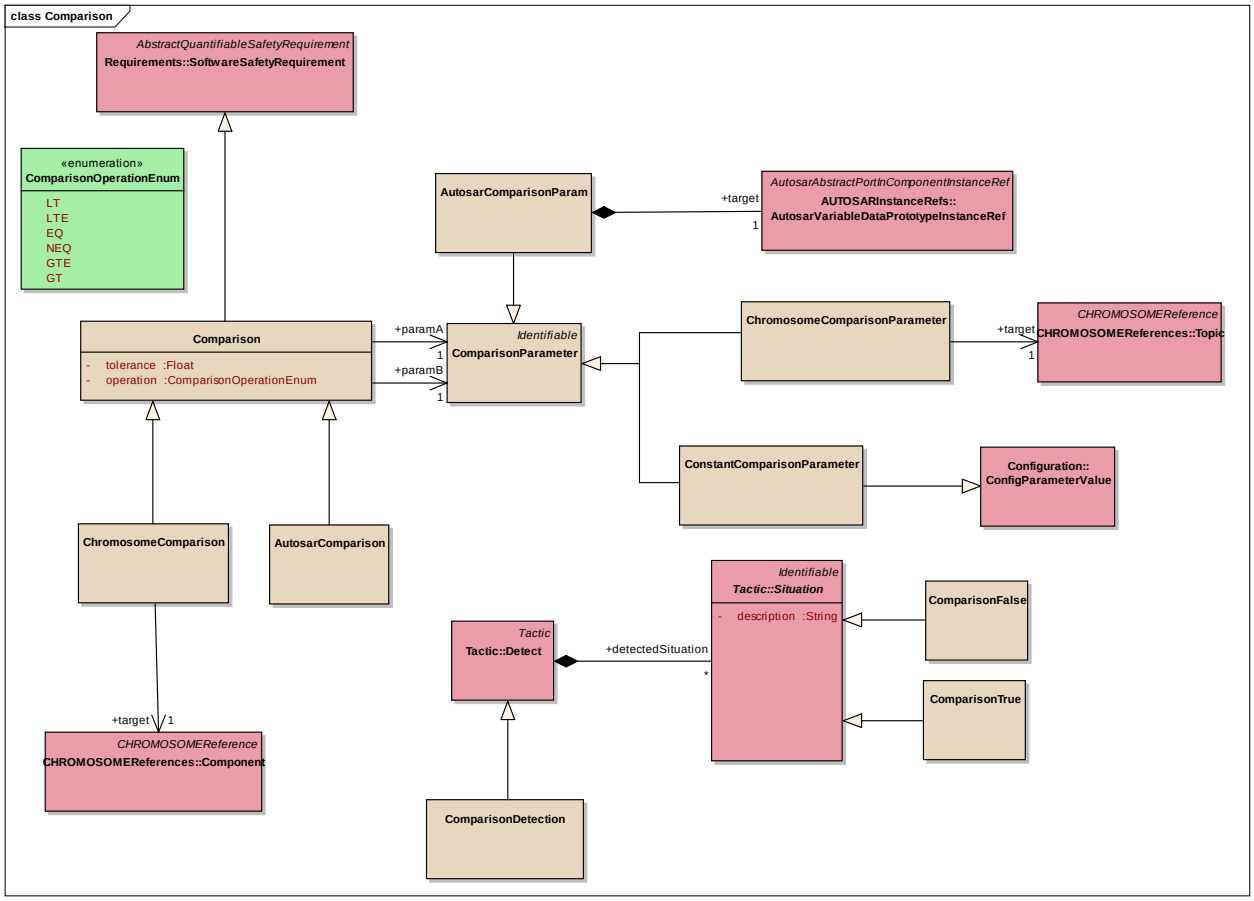


Figure 60: **Comparison** - *(Class diagram)*

Diagram Notes:

Class AutosarComparison

Element Base Classes: **Comparison**

Element Notes:

AUTOSAR specific comparison SSR. The compared values are provided as AUTOSAR variable instance references and the operation applied to inputs is defined through the Operation enumeration.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarComparison	Comparison

Class AutosarComparisonParam

Element Base Classes: **ComparisonParameter**

Element Notes:

AUTOSAR specific comparison parameters. Allows to reference AUTOSAR elements used in the comparison.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarComparisonParam	ComparisonParameter
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarComparisonParam

Class ChromosomeComparison

Element Base Classes: **Comparison**

Element Notes:

Chromosome comparison is referencing a specific Chromosome component instance, which is implementing comparison of one or multiple topics.

Connections

Connector	Source	Target
Generalization Source -> Destination	ChromosomeComparison	Comparison
Association Source -> Destination	ChromosomeComparison	Component

Class ChromosomeComparisonParameter

Element Base Classes: **ComparisonParameter**

Element Notes:

Chromosome specific comparison parameter. Allows to reference a specific Chromosome topic, which acts as a comparison entity.

Connections

Connector	Source	Target
Generalization Source -> Destination	ChromosomeComparisonParameter	ComparisonParameter
Association Source -> Destination	ChromosomeComparisonParameter	Topic

Class Comparison

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

Value comparison SSR.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarComparison	Comparison
Association Source -> Destination	Comparison	ComparisonParameter
Association Source -> Destination	Comparison	ComparisonParameter
Generalization Source -> Destination	ChromosomeComparison	Comparison
Generalization Source -> Destination	Comparison	SoftwareSafetyRequirement

Attributes

Attribute	Notes	Default
tolerance Float		
operation ComparisonOperationEnum		

Class ComparisonDetection
Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ComparisonDetection	Detect

Class ComparisonFalse
Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ComparisonFalse	Situation

Enumeration ComparisonOperationEnum
Element Base Classes:

Element Notes:

Enumeration providing the valid operations for a comparison SSR.

Attributes

Attribute	Notes	Default
LT		
LTE		
EQ		
NEQ		
GTE		
GT		

Class ComparisonParameter

Element Base Classes: **Identifiable**

Element Notes:

Abstract comparison parameter element.

Connections

Connector	Source	Target
Generalization Source -> Destination	ChromosomeComparisonParameter	ComparisonParameter
Generalization Source -> Destination	ConstantComparisonParameter	ComparisonParameter
Association Source -> Destination	Comparison	ComparisonParameter

Connector	Source	Target
Generalization Source -> Destination	AutosarComparisonParam	ComparisonParameter
Association Source -> Destination	Comparison	ComparisonParameter
Generalization Source -> Destination	ComparisonParameter	Identifiable

Class ComparisonTrue

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ComparisonTrue	Situation

Class ConstantComparisonParameter

Element Base Classes: **ComparisonParameter, ConfigParameterValue**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ConstantComparisonParameter	ConfigParameterValue
Generalization Source -> Destination	ConstantComparisonParameter	ComparisonParameter

Package ContextRangeCheck

Package Notes:

This package groups the elements related to the software safety mechanism Context Range Check

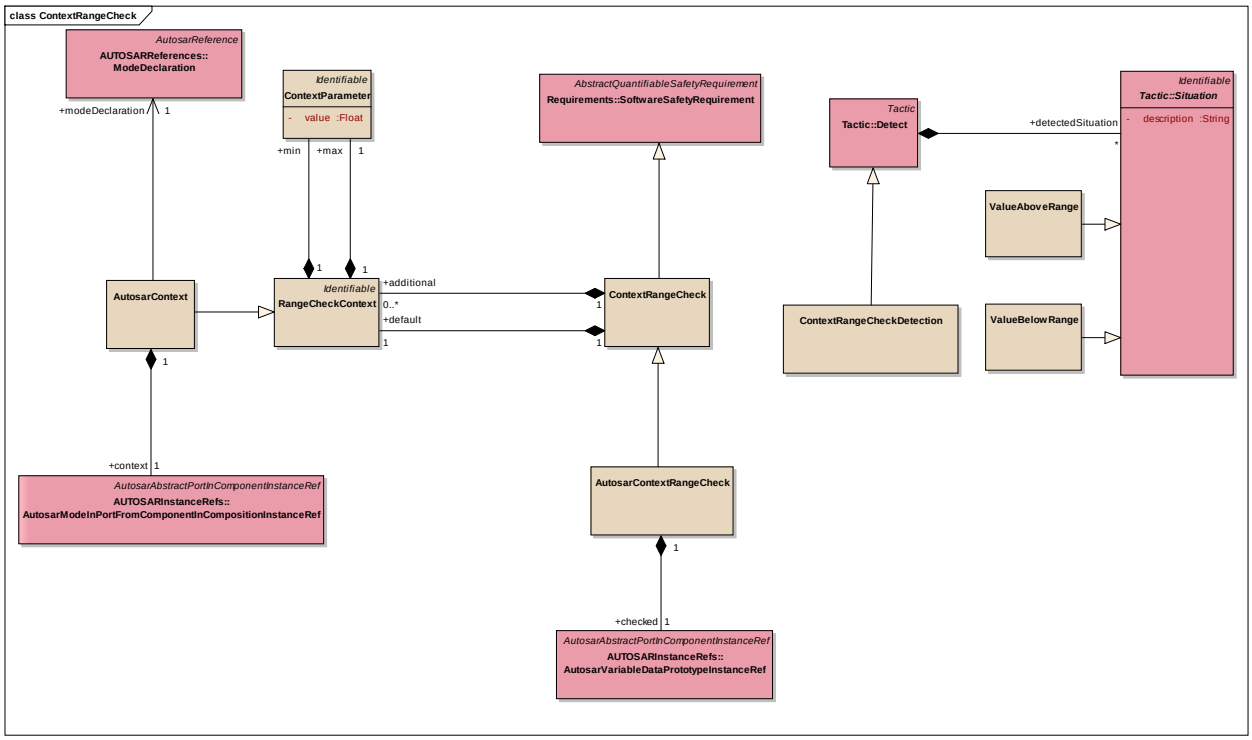


Figure 61: **ContextRangeCheck** - *(Class diagram)*

Diagram Notes:

Class AutosarContext

Element Base Classes: **RangeCheckContext**

Element Notes:

AUTOSAR specific meta class for defining the context used by the range check SSR.

Connections

Connector	Source	Target
Aggregation Source -> Destination	AutosarModeInPortFromComponentInCompositionInstanceRef	AutosarContext
Association Source -> Destination	AutosarContext	ModeDeclaration
Generalization Source -> Destination	AutosarContext	RangeCheckContext

Class AutosarContextRangeCheck

Element Base Classes: **ContextRangeCheck**

Element Notes:

AUTOSAR specific meta class defining the context range check SSR.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarContextRangeCheck	ContextRangeCheck
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarContextRangeCheck

Class ContextParameter

Element Base Classes: **Identifiable**

Element Notes:

Provides the valid range parameters for a given context.

Connections

Connector	Source	Target

Connector	Source	Target
Aggregation Source -> Destination	ContextParameter	RangeCheckContext
Aggregation Source -> Destination	ContextParameter	RangeCheckContext
Generalization Source -> Destination	ContextParameter	Identifiable

Attributes

Attribute	Notes	Default
value Float		

Class ContextRangeCheck

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

This meta-class defines a range check SSR whose range is defined based on the current system/component context (e.g. AUTOSAR mode).

Connections

Connector	Source	Target
Aggregation Source -> Destination	RangeCheckContext	ContextRangeCheck
Generalization Source -> Destination	AutosarContextRangeCheck	ContextRangeCheck
Aggregation Source -> Destination	RangeCheckContext	ContextRangeCheck
Generalization Source -> Destination	ContextRangeCheck	SoftwareSafetyRequirement

Class ContextRangeCheckDetection

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ContextRangeCheckDetection	Detect

Class RangeCheckContext

Element Base Classes: **Identifiable**

Element Notes:

Defines the parameters used by the range check SSR when the system is in a given AUTOSAR mode.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarContext	RangeCheckContext
Aggregation Source -> Destination	ContextParameter	RangeCheckContext
Aggregation Source -> Destination	ContextParameter	RangeCheckContext
Aggregation Source -> Destination	RangeCheckContext	ContextRangeCheck
Generalization Source -> Destination	RangeCheckContext	Identifiable
Aggregation Source -> Destination	RangeCheckContext	ContextRangeCheck

Class ValueAboveRange

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ValueAboveRange	Situation

Class ValueBelowRange

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	ValueBelowRange	Situation

Package CpuSelfTest

Package Notes:

This package groups the elements related to the software safety mechanism CPU Self Test

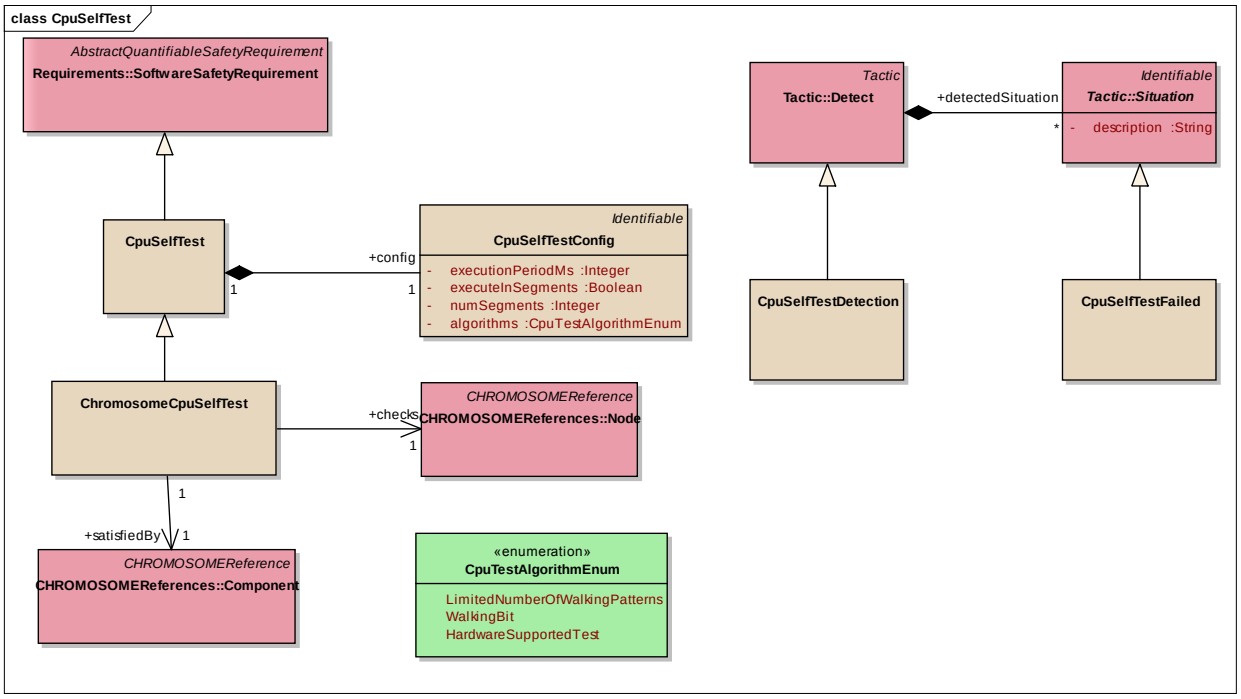


Figure 62: **CpuSelfTest** - *(Class diagram)*

Diagram Notes:

Class ChromosomeCpuSelfTest

Element Base Classes: **CpuSelfTest**

Element Notes:

CHROMOSOME specific CPU self-test meta class.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeCpuSelfTest	Component
Generalization Source -> Destination	ChromosomeCpuSelfTest	CpuSelfTest
Association Source -> Destination	ChromosomeCpuSelfTest	Node

Class CpuSelfTest

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

This meta-class element defines a CPU self-test SSR. The entity implementing such an SSR is executed periodically according to its configuration, and may trigger error handlers in case of error detection.

Connections

Connector	Source	Target
Generalization Source -> Destination	CpuSelfTest	SoftwareSafetyRequirement
Generalization Source -> Destination	ChromosomeCpuSelfTest	CpuSelfTest
Aggregation Source -> Destination	CpuSelfTestConfig	CpuSelfTest

Class CpuSelfTestConfig

Element Base Classes: **Identifiable**

Element Notes:

Configuration meta class to configure CPU test SSR.

Connections

Connector	Source	Target
Generalization Source -> Destination	CpuSelfTestConfig	Identifiable
Aggregation Source -> Destination	CpuSelfTestConfig	CpuSelfTest

Attributes

Attribute	Notes	Default
executionPeriodMs Integer		

Attribute	Notes	Default
executeInSegments Boolean		
numSegments Integer		
algorithms CpuTestAlgorithmEnum		

Class CpuSelfTestDetection

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	CpuSelfTestDetection	Detect

Class CpuSelfTestFailed

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	CpuSelfTestFailed	Situation

Enumeration CpuTestAlgorithmEnum

Element Base Classes:

Element Notes:

Defines possible CPU test algorithms.

Attributes

Attribute	Notes	Default
LimitedNumberOfWalkingPatterns		
WalkingBit		
HardwareSupportedTest		

Package Filter

Package Notes:

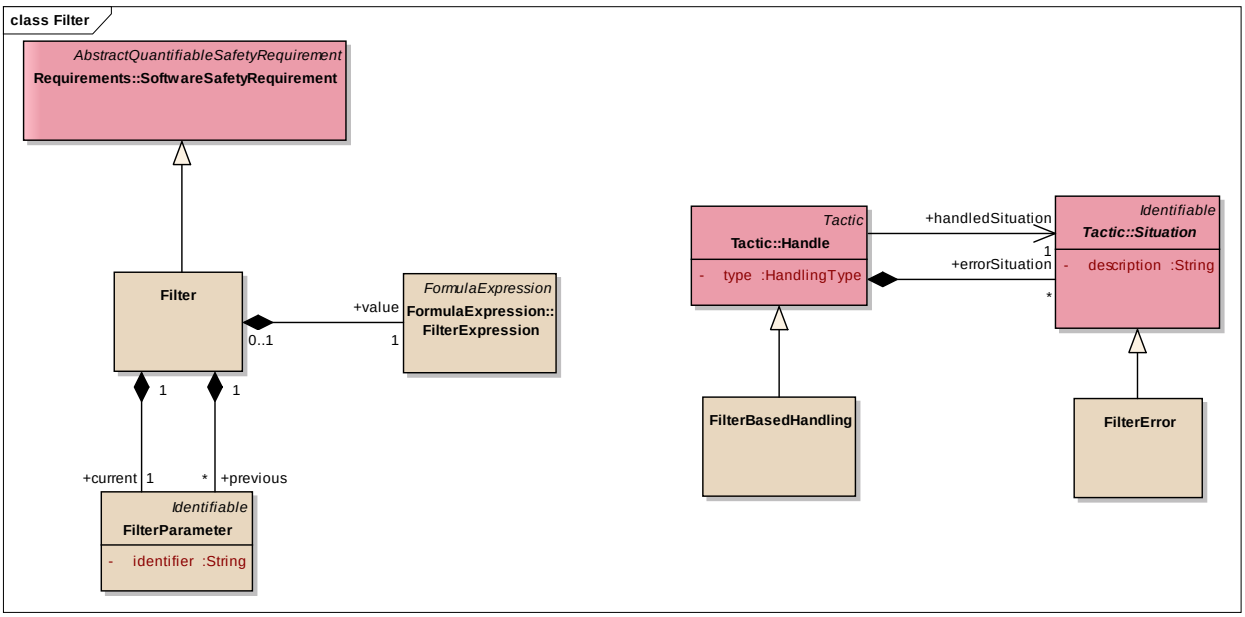


Figure 63: **Filter** - *(Class diagram)*

Diagram Notes:

Class Filter

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

The Filter meta element represents a error handling specification which defines how an erroneous value provided to it can be corrected. The mechanism can be referred by detection software safety requirements.

Connections

Connector	Source	Target
Generalization Source -> Destination	Filter	SoftwareSafetyRequirement
Aggregation Source -> Destination	FilterParameter	Filter
Aggregation Source -> Destination	FilterParameter	Filter
Aggregation Source -> Destination	FilterExpression	Filter

Class FilterBasedHandling

Element Base Classes: **Handle**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FilterBasedHandling	Handle

Class FilterError

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	FilterError	Situation

Class FilterParameter

Element Base Classes: **Identifiable**

Element Notes:

A FilterParameter meta element defines the possible values which can be referred within a Filter. These are used to calculate and correct erroneous values provided to a filter realization.

Connections

Connector	Source	Target
Aggregation Source -> Destination	FilterParameter	Filter
Generalization Source -> Destination	FilterParameter	Identifiable
Aggregation Source -> Destination	FilterParameter	Filter

Attributes

Attribute	Notes	Default
identifier String		

Package GradientCheck

Package Notes:

This package groups the elements related to the software safety mechanism Gradient Check

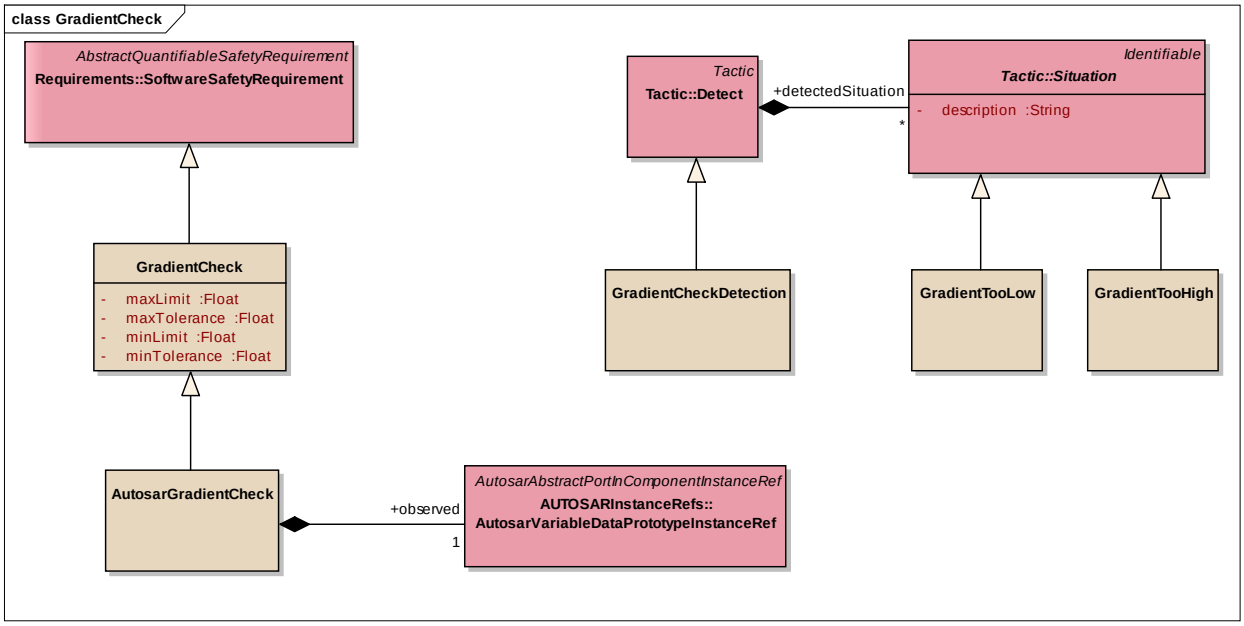


Figure 64: **GradientCheck** - *(Class diagram)*

Diagram Notes:

Class AutosarGradientCheck

Element Base Classes: **GradientCheck**

Element Notes:

AUTOSAR specific meta class for defining the gradient check SSR.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarGradientCheck	GradientCheck
Aggregation Source -> Destination	AutosarVariableDataPrototypeInstanceRef	AutosarGradientCheck

Connector	Source	Target

Class GradientCheck

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

This meta-class element defines a gradient check SSR. The limit and tolerance attributes are used to check if the values are varying within a valid range. The SSR is executed periodically according to its period attribute.

Connections

Connector	Source	Target
Generalization Source -> Destination	AutosarGradientCheck	GradientCheck
Generalization Source -> Destination	GradientCheck	SoftwareSafetyRequirement

Attributes

Attribute	Notes	Default
maxLimit Float		
maxTolerance Float		
minLimit Float		
minTolerance Float		

Class GradientCheckDetection

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	GradientCheckDetection	Detect

Class GradientTooHigh

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	GradientTooHigh	Situation

Class GradientTooLow

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	GradientTooLow	Situation

Package HealthMonitor

Package Notes:

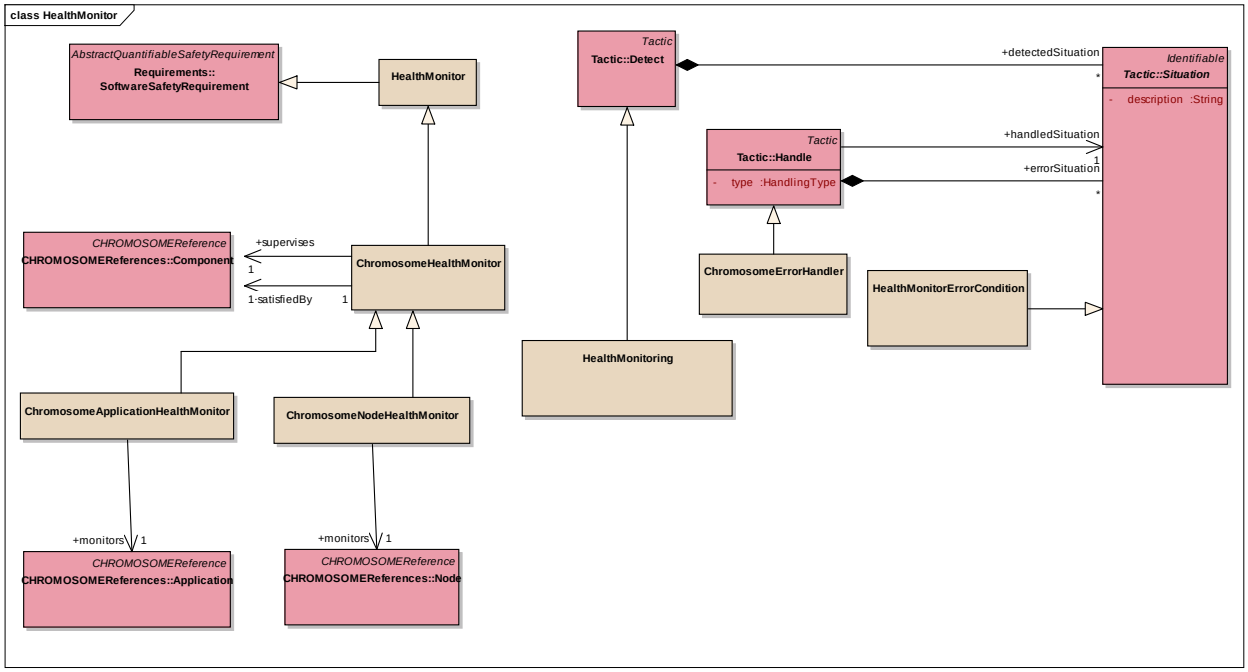


Figure 65: **HealthMonitor** - *(Class diagram)*

Diagram Notes:

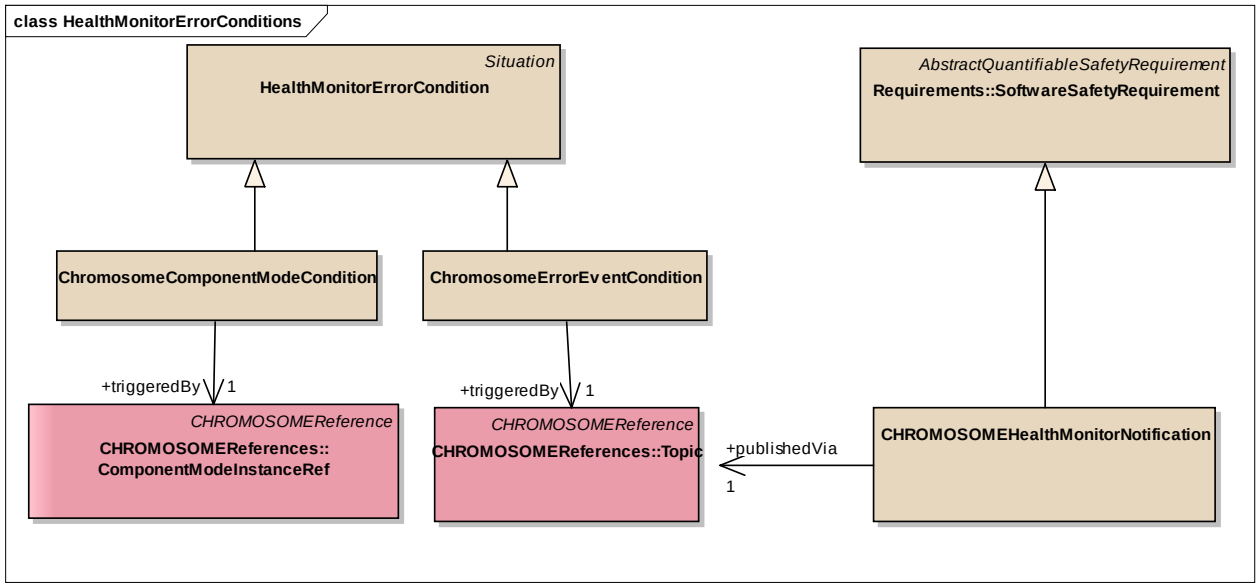


Figure 66: **HealthMonitorErrorConditions** - *(Class diagram)*

Diagram Notes:

Class CHROMOSOMEHealthMonitorNotification

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

Provides a mechanism to delegate error handling to the CHROMOSOME HealthMonitor.

Connections

Connector	Source	Target
Generalization Source -> Destination	CHROMOSOMEHealthMonitorNotification	SoftwareSafetyRequirement
Association Source -> Destination	CHROMOSOMEHealthMonitorNotification	Topic

Class ChromosomeApplicationHealthMonitor

Element Base Classes: **ChromosomeHealthMonitor**

Element Notes:

CHROMOSOME specific Health Monitor meta class for monitoring a Chromosome Application as a set of CHROMOSOME components.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeApplicationHealthMonitor	Application
Generalization Source -> Destination	ChromosomeApplicationHealthMonitor	ChromosomeHealthMonitor

Connector	Source	Target

Class ChromosomeComponentModeCondition

Element Base Classes: **HealthMonitorErrorCondition**

Element Notes:

A CHROMOSOME specific condition referencing a Mode of a certain CHROMOSOME component instance. The condition is fired when the component is monitored by the Health Monitor and enters mode specified by triggeredBy relation.

Connections

Connector	Source	Target
Generalization Source -> Destination	ChromosomeComponentModeCondition	HealthMonitorErrorCondition
Association Source -> Destination	ChromosomeComponentModeCondition	ComponentModeInstanceRef

Class ChromosomeErrorEventCondition

Element Base Classes: **HealthMonitorErrorCondition**

Element Notes:

A CHROMOSOME specific error condition, which is triggered by reception of a message via a predefined CHROMOSOME topic referenced by triggeredBy relation.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeErrorEventCondition	Topic

Connector	Source	Target
Generalization Source -> Destination	ChromosomeErrorEventCondition	HealthMonitorErrorCondition

Class ChromosomeErrorHandler

Element Base Classes: **Handle**

Element Notes:

CHROMOSOME specific user-defined error handling function.

Connections

Connector	Source	Target
Generalization Source -> Destination	ChromosomeErrorHandler	Handle

Class ChromosomeHealthMonitor

Element Base Classes: **HealthMonitor**

Element Notes:

CHROMOSOME specific Health Monitor meta class.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeHealthMonitor	Component
Generalization Source -> Destination	ChromosomeApplicationHealthMonitor	ChromosomeHealthMonitor
Association Source -> Destination	ChromosomeHealthMonitor	Component
Generalization	ChromosomeHealthMonitor	HealthMonitor

Connector	Source	Target
Source -> Destination		
Generalization Source -> Destination	ChromosomeNodeHealthMonitor	ChromosomeHealthMonitor

Class ChromosomeNodeHealthMonitor

Element Base Classes: **ChromosomeHealthMonitor**

Element Notes:

CHROMOSOME specific Health Monitor meta class for monitoring a Chromosome Node as a set of CHROMOSOME components.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeNodeHealthMonitor	Node
Generalization Source -> Destination	ChromosomeNodeHealthMonitor	ChromosomeHealthMonitor

Class HealthMonitor

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

This meta-class element defines a health monitor SSR.

Connections

Connector	Source	Target
Generalization Source -> Destination	HealthMonitor	SoftwareSafetyRequirement
Generalization Source -> Destination	ChromosomeHealthMonitor	HealthMonitor

Class HealthMonitorErrorCondition

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	HealthMonitorErrorCondition	Situation
Generalization Source -> Destination	ChromosomeComponentModeCondition	HealthMonitorErrorCondition
Generalization Source -> Destination	ChromosomeErrorEventCondition	HealthMonitorErrorCondition

Class HealthMonitoring

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	HealthMonitoring	Detect

Package Heartbeat

Package Notes:

This package groups the elements related to the software safety mechanism Heartbeat.

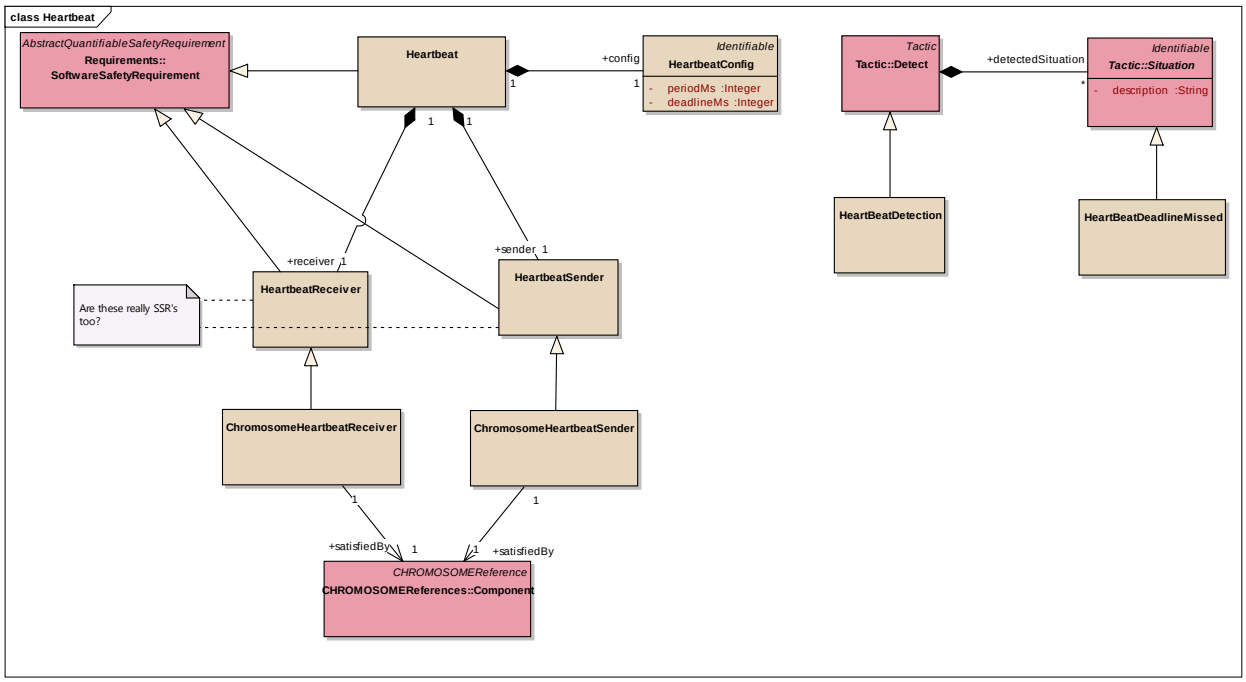


Figure 67: **Heartbeat** - *(Class diagram)*

Diagram Notes:

Class ChromosomeHeartbeatReceiver

Element Base Classes: **HeartbeatReceiver**

Element Notes:

A CHROMOSOME specific meta class to define a heartbeat receiver

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeHeartbeatReceiver	Component
Generalization Source -> Destination	ChromosomeHeartbeatReceiver	HeartbeatReceiver

Class ChromosomeHeartbeatSender

Element Base Classes: **HeartbeatSender**

Element Notes:

A CHROMOSOME specific meta class to define a heartbeat sender.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeHeartbeatSender	Component
Generalization Source -> Destination	ChromosomeHeartbeatSender	HeartbeatSender

Class HeartBeatDeadlineMissed

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	HeartBeatDeadlineMissed	Situation

Class HeartBeatDetection

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	HeartBeatDetection	Detect

Class Heartbeat

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

A Heartbeat SSR requires that two runnable entities on different nodes are instantiated and one of them monitors the signals arriving from the other one.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HeartbeatConfig	Heartbeat
Aggregation Source -> Destination	HeartbeatSender	Heartbeat
Aggregation Source -> Destination	HeartbeatReceiver	Heartbeat
Generalization Source -> Destination	Heartbeat	SoftwareSafetyRequirement

Class HeartbeatConfig

Element Base Classes: **Identifiable**

Element Notes:

Configuration of Heartbeat SSR: timing of source and receiver.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HeartbeatConfig	Heartbeat
Generalization Source -> Destination	HeartbeatConfig	Identifiable

Attributes

Attribute	Notes	Default
periodMs Integer		
deadlineMs Integer		

Class HeartbeatReceiver

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

An SSR representing the receiver of the heartbeat signal.

Connections

Connector	Source	Target
Generalization Source -> Destination	HeartbeatReceiver	SoftwareSafetyRequirement
Aggregation Source -> Destination	HeartbeatReceiver	Heartbeat
NoteLink Source -> Destination	<anonymous>	HeartbeatReceiver
Generalization Source -> Destination	ChromosomeHeartbeatReceiver	HeartbeatReceiver

Class HeartbeatSender

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

An SSR representing the sender of the heartbeat signal.

Connections

Connector	Source	Target
Generalization Source -> Destination	HeartbeatSender	SoftwareSafetyRequirement
Aggregation Source -> Destination	HeartbeatSender	Heartbeat
NoteLink Source -> Destination	<anonymous>	HeartbeatSender
Generalization Source -> Destination	ChromosomeHeartbeatSender	HeartbeatSender

Package MemorySelfTest

Package Notes:

This package groups the elements related to the software safety mechanism Memory (RAM) Self Test

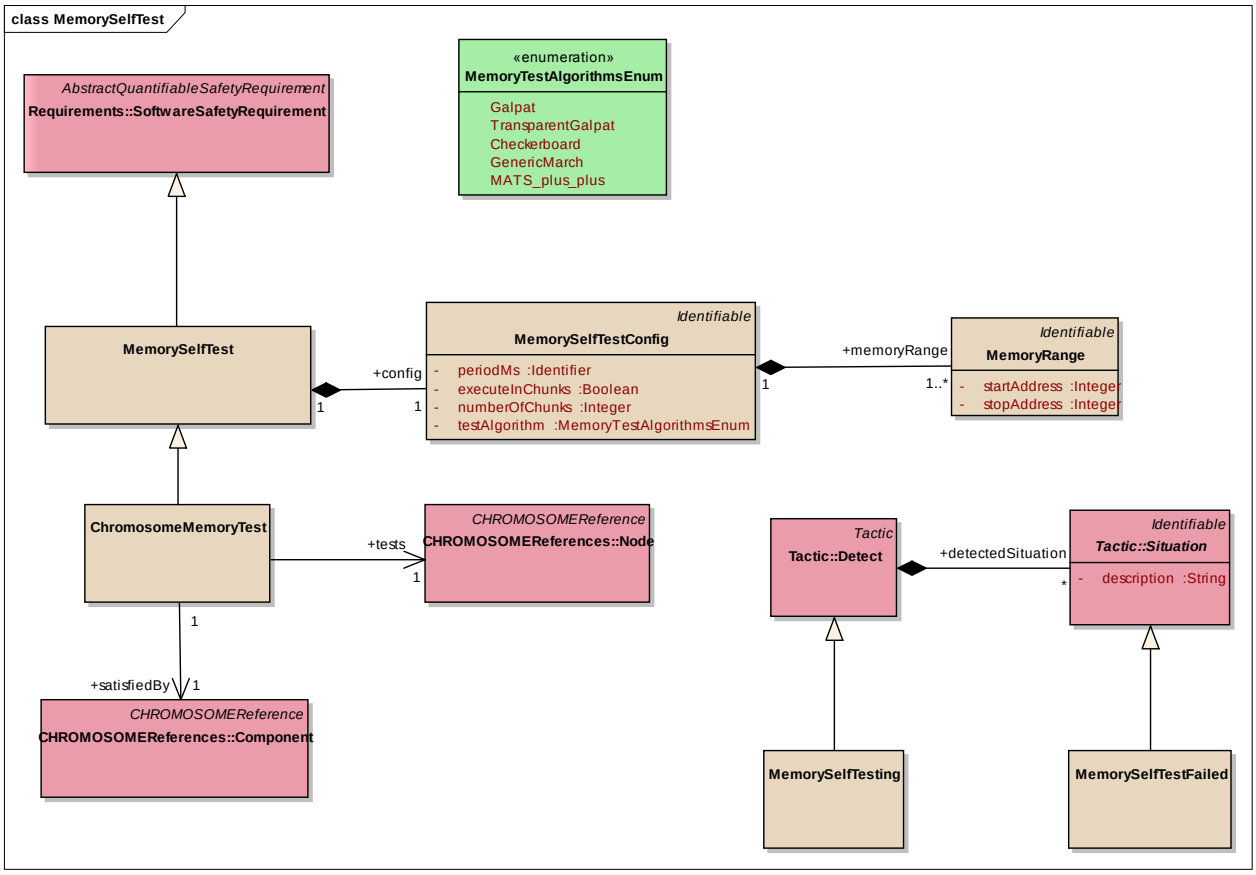


Figure 68: **MemorySelfTest** - *(Class diagram)*

Diagram Notes:

Class ChromosomeMemoryTest

Element Base Classes: **MemorySelfTest**

Element Notes:

A CHROMOSOME-specific MemorySelfTest requirement.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeMemoryTest	Node
Generalization Source -> Destination	ChromosomeMemoryTest	MemorySelfTest

Connector	Source	Target
Association Source -> Destination	ChromosomeMemoryTest	Component

Class MemoryRange

Element Base Classes: **Identifiable**

Element Notes:

An address range is a class to specify the exact range of addresses in the node address space to be tested.

Connections

Connector	Source	Target
Aggregation Source -> Destination	MemoryRange	MemorySelfTestConfig
Generalization Source -> Destination	MemoryRange	Identifiable

Attributes

Attribute	Notes	Default
startAddress Integer		
stopAddress Integer		

Class MemorySelfTest

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

A meta class defining Memory Self-Test SSR.

Connections

Connector	Source	Target

Connector	Source	Target
Aggregation Source -> Destination	MemorySelfTestConfig	MemorySelfTest
Generalization Source -> Destination	ChromosomeMemoryTest	MemorySelfTest
Generalization Source -> Destination	MemorySelfTest	SoftwareSafetyRequirement

Class MemorySelfTestConfig

Element Base Classes: **Identifiable**

Element Notes:

Specifies configuration parameters for MemorySelfTest SSR.

Connections

Connector	Source	Target
Generalization Source -> Destination	MemorySelfTestConfig	Identifiable
Aggregation Source -> Destination	MemorySelfTestConfig	MemorySelfTest
Aggregation Source -> Destination	MemoryRange	MemorySelfTestConfig

Attributes

Attribute	Notes	Default
periodMs Identifier		
executeInChunks Boolean		
numberOfChunks Integer		
testAlgorithm MemoryTestAlgorithmsEn		

Attribute	Notes	Default
um		

Class MemorySelfTestFailed

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	MemorySelfTestFailed	Situation

Class MemorySelfTesting

Element Base Classes: **Detect**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	MemorySelfTesting	Detect

Enumeration MemoryTestAlgorithmsEnum

Element Base Classes:

Element Notes:

Defines different memory test algorithms that can be defined in a configuration

Attributes

Attribute	Notes	Default
Galpat		
TransparentGalpat		
Checkerboard		
GenericMarch		
MATS_plus_plus		

Package Voting

Package Notes:

This package groups the elements related to the software safety mechanism Voting

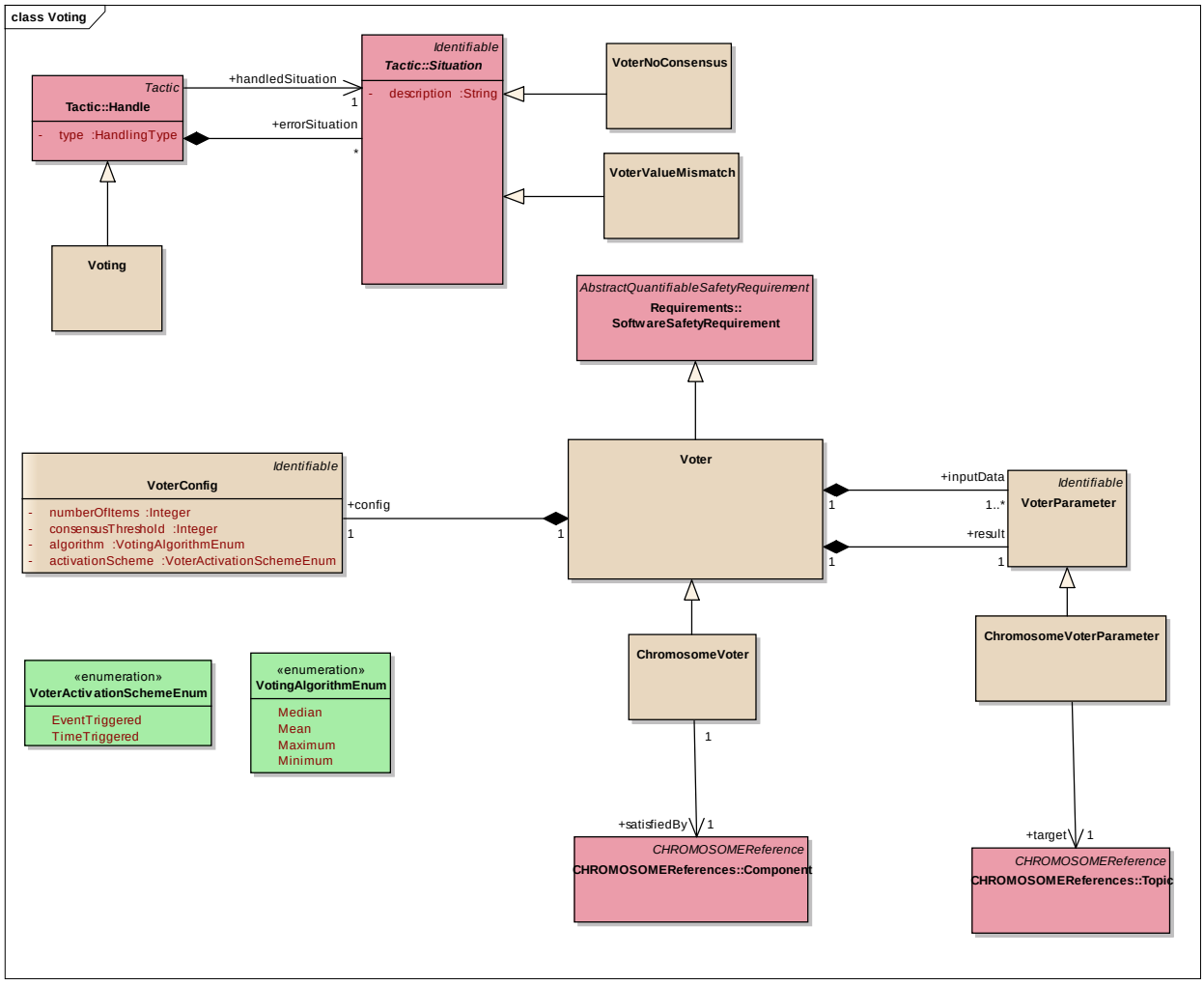


Figure 69: **Voting** - *(Class diagram)*

Diagram Notes:

Class ChromosomeVoter

Element Base Classes: **Voter**

Element Notes:

A CHROMOSOME-specific meta class for Voting

Connections

Connector	Source	Target

Connector	Source	Target
Generalization Source -> Destination	ChromosomeVoter	Voter
Association Source -> Destination	ChromosomeVoter	Component

Class ChromosomeVoterParameter

Element Base Classes: **VoterParameter**

Element Notes:

A CHROMOSOME-specific voting parameter meta class.

Connections

Connector	Source	Target
Association Source -> Destination	ChromosomeVoterParameter	Topic
Generalization Source -> Destination	ChromosomeVoterParameter	VoterParameter

Class Voter

Element Base Classes: **SoftwareSafetyRequirement**

Element Notes:

A meta-class representing the Voting software safety requirement.

Connections

Connector	Source	Target
Aggregation Source -> Destination	VoterParameter	Voter
Generalization Source -> Destination	ChromosomeVoter	Voter

Connector	Source	Target
Aggregation Source -> Destination	VoterParameter	Voter
Generalization Source -> Destination	Voter	SoftwareSafetyRequirement
Aggregation Source -> Destination	VoterConfig	Voter

Enumeration VoterActivationSchemeEnum

Element Base Classes:

Element Notes:

Possible activation schemes for a Voting component.

Attributes

Attribute	Notes	Default
EventTriggered		
TimeTriggered		

Class VoterConfig

Element Base Classes: **Identifiable**

Element Notes:

Configuration of a Voting SSR

Connections

Connector	Source	Target
Generalization Source -> Destination	VoterConfig	Identifiable

Connector	Source	Target
Aggregation Source -> Destination	VoterConfig	Voter

Attributes

Attribute	Notes	Default
numberOfItems Integer		
consensusThreshold Integer		
algorithm VotingAlgorithmEnum		
activationScheme VoterActivationSchemeEnum		

Class VoterNoConsensus

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	VoterNoConsensus	Situation

Class VoterParameter

Element Base Classes: **Identifiable**

Element Notes:

A meta-class for representing input data and outputs of a Voting SSR.

Connections

Connector	Source	Target
Aggregation Source -> Destination	VoterParameter	Voter
Generalization Source -> Destination	ChromosomeVoterParameter	VoterParameter
Aggregation Source -> Destination	VoterParameter	Voter
Generalization Source -> Destination	VoterParameter	Identifiable

Class VoterValueMismatch

Element Base Classes: **Situation**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	VoterValueMismatch	Situation

Class Voting

Element Base Classes: **Handle**

Element Notes:

Connections

Connector	Source	Target

Connector	Source	Target
Generalization Source -> Destination	Voting	Handle

Enumeration VotingAlgorithmEnum

Element Base Classes:

Element Notes:

Possible voting algorithms: if the values are different, which one should be returned?

Attributes

Attribute	Notes	Default
Median		
Mean		
Maximum		
Minimum		

Package Tactic

Package Notes:

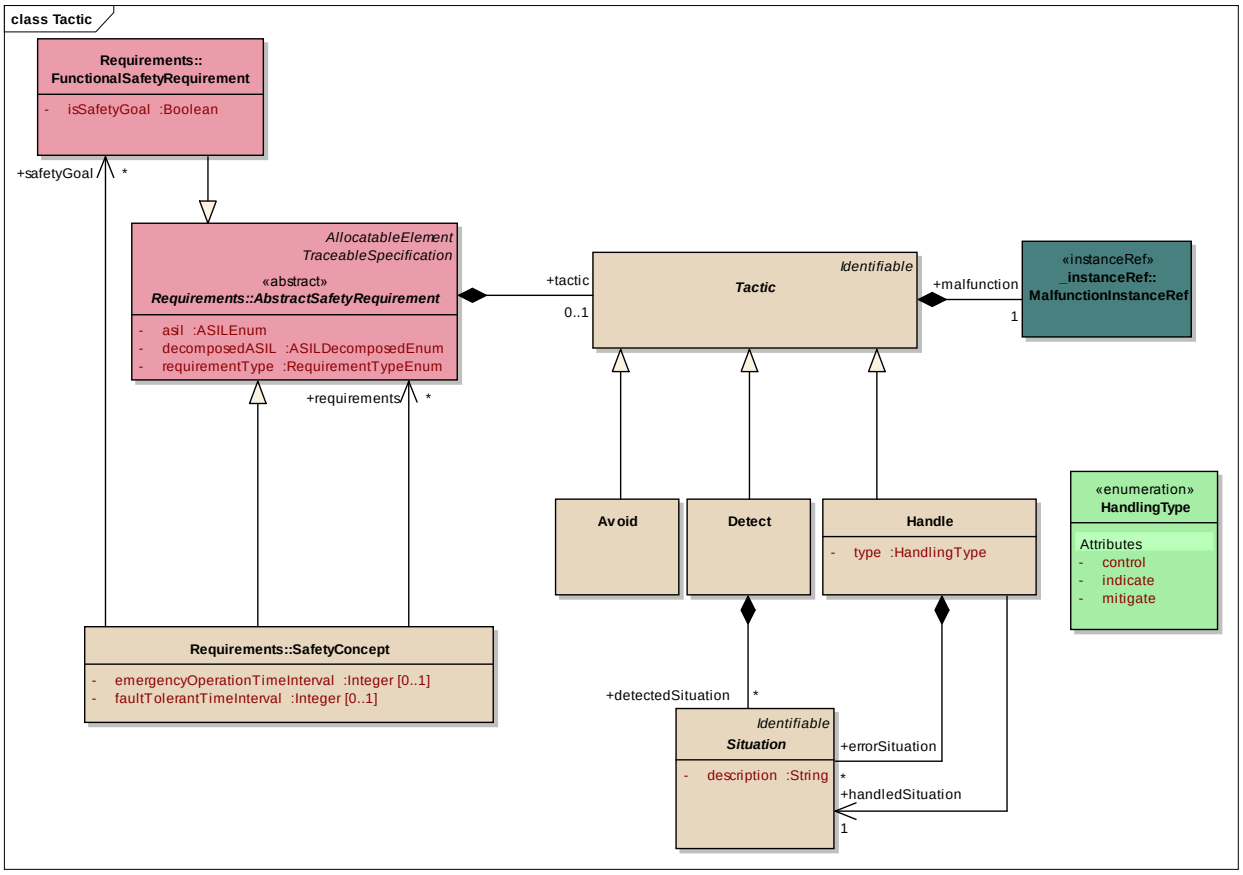


Figure 70: **Tactic** - *(Class diagram)*

Diagram Notes:

Class Avoid

Element Base Classes: **Tactic**

Element Notes:

Classifies requirements as avoidance requirements.

Connections

Connector	Source	Target
Generalization Source -> Destination	Avoid	Tactic

Class Detect

Element Base Classes: **Tactic**

Element Notes:

Classifies requirements as detection requirements

Connections

Connector	Source	Target
Generalization Source -> Destination	HealthMonitoring	Detect
Generalization Source -> Destination	ControlFlowErrorDetection	Detect
Generalization Source -> Destination	GradientCheckDetection	Detect
Generalization Source -> Destination	MemorySelfTesting	Detect
Generalization Source -> Destination	HeartBeatDetection	Detect
Generalization Source -> Destination	ComparisonDetection	Detect
Generalization Source -> Destination	ActuatorErrorDetection	Detect
Generalization Source -> Destination	Detect	Tactic
Generalization Source -> Destination	ContextRangeCheckDetection	Detect
Generalization Source -> Destination	CpuSelfTestDetection	Detect
Generalization Source -> Destination	CRCErrorDetection	Detect
Generalization Source -> Destination	SoftwareAlivenessDetection	Detect
Aggregation Source -> Destination	Situation	Detect

Class Handle

Element Base Classes: **Tactic**

Element Notes:

Classifies requirements as handling requirements.

Connections

Connector	Source	Target
Generalization Source -> Destination	FilterBasedHandling	Handle
Generalization Source -> Destination	Handle	Tactic
Generalization Source -> Destination	ChromosomeErrorHandler	Handle
Association Source -> Destination	Handle	Situation
Generalization Source -> Destination	WarningAndDegradationHandle	Handle
Generalization Source -> Destination	Voting	Handle
Aggregation Source -> Destination	Situation	Handle

Attributes

Attribute	Notes	Default
type HandlingType		

Enumeration HandlingType

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
control		
indicate		
mitigate		

Class Situation

Element Base Classes: **Identifiable**

Element Notes:

Defines the possible error detection situations a given requirement might be able to detect. For example, at the functional safety level, a detection requirement might detect "vehicle yaw rate too high", this is an error situation related to the detection of vehicle yaw rate malfunctions.

Connections

Connector	Source	Target
Generalization Source -> Destination	HealthMonitorErrorCondition	Situation
Generalization Source -> Destination	AlivenessCheckpointTooEarly	Situation
Generalization Source -> Destination	MemorySelfTestFailed	Situation
Generalization Source -> Destination	ValueAboveRange	Situation
Generalization Source -> Destination	FilterError	Situation
Generalization	AlivenessCheckpointTooOften	Situation

Connector	Source	Target
Source -> Destination		
Generalization Source -> Destination	ActuatorError	Situation
Generalization Source -> Destination	VoterNoConsensus	Situation
Generalization Source -> Destination	ValueBelowRange	Situation
Association Source -> Destination	Handle	Situation
Generalization Source -> Destination	GradientTooLow	Situation
Generalization Source -> Destination	AlivenessCheckpointTooLate	Situation
Generalization Source -> Destination	ControlFlowError	Situation
Generalization Source -> Destination	Situation	Identifiable
Generalization Source -> Destination	VoterValueMismatch	Situation
Generalization Source -> Destination	CpuSelfTestFailed	Situation
Generalization Source -> Destination	CRCFailed	Situation
Generalization Source -> Destination	GradientTooHigh	Situation
Aggregation Source -> Destination	Situation	Handle
Generalization Source -> Destination	ComparisonTrue	Situation
Aggregation Source -> Destination	Situation	Detect
Generalization	HeartBeatDeadlineMissed	Situation

Connector	Source	Target
Source -> Destination		
Generalization Source -> Destination	ComparisonFalse	Situation

Attributes

Attribute	Notes	Default
description String		

Class Tactic

Element Base Classes: **Identifiable**

Element Notes:

The Tactic element is responsible for defining the role a given requirement has regarding error management. Three kinds of tactic are allowed for requirements, namely, avoid, detect and handle. The tactic also refers to the malfunction being avoided, detected or handled.

Connections

Connector	Source	Target
Generalization Source -> Destination	Handle	Tactic
Generalization Source -> Destination	Tactic	Identifiable
Generalization Source -> Destination	Detect	Tactic
Aggregation Source -> Destination	MalfunctionInstanceRef	Tactic
Aggregation Source -> Destination	Tactic	AbstractSafetyRequirement
Generalization Source -> Destination	Avoid	Tactic

Package Software

Package Notes:

Package Configuration

Package Notes:

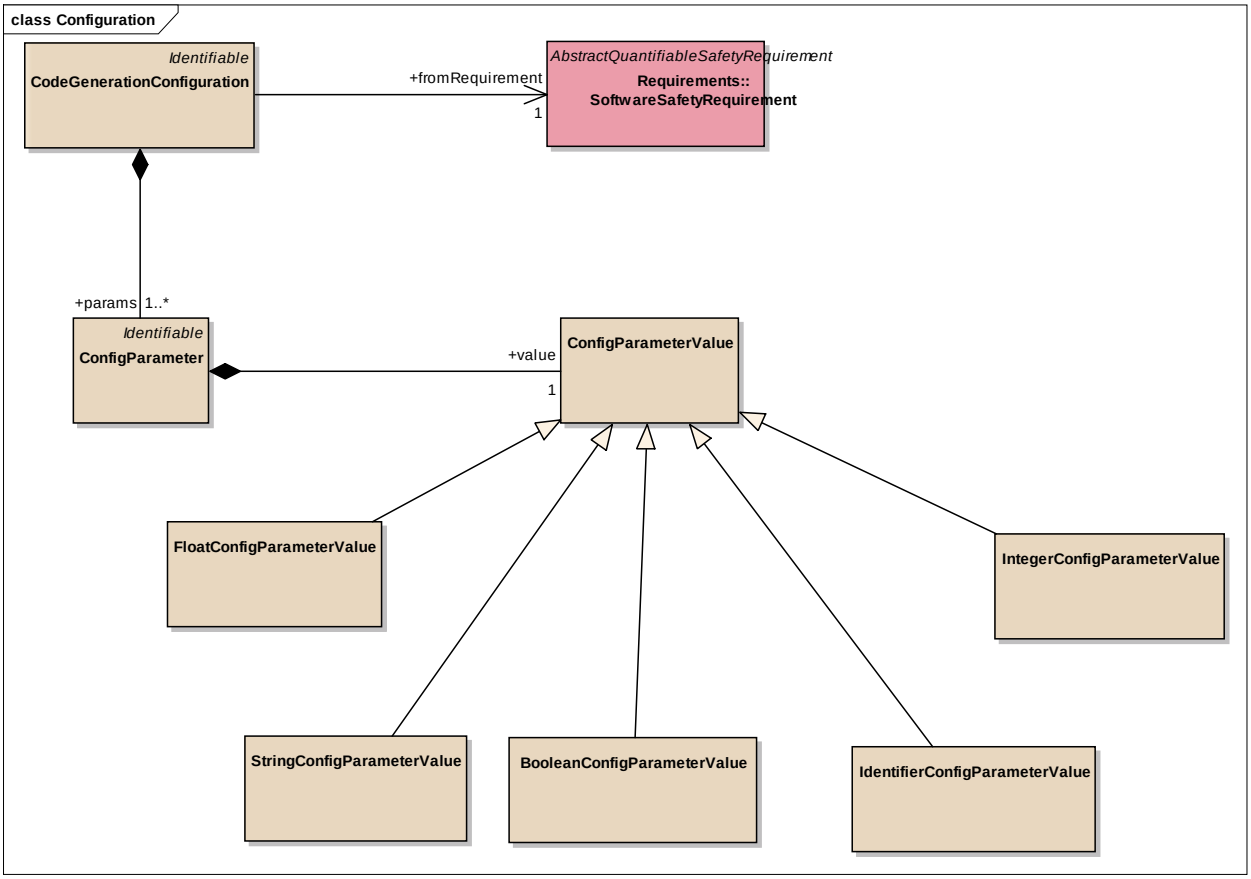


Figure 71: **Configuration** - *(Class diagram)*

Diagram Notes:

Class BooleanConfigParameterValue

Element Base Classes: **ConfigParameterValue**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	BooleanConfigParameterValue	ConfigParameterValue

Class CodeGenerationConfiguration

Element Base Classes: **Identifiable**

Element Notes:

An SSMConfiguration defines the parameters needed by specific code generators. It provides a set of name/value pairs which contain additional information not defined within the software safety mechanism.

Connections

Connector	Source	Target
Aggregation Source -> Destination	CodeGenerationConfiguration	ImplementationSafetyExtension
Aggregation Source -> Destination	ConfigParameter	CodeGenerationConfiguration
Association Source -> Destination	CodeGenerationConfiguration	SoftwareSafetyRequirement
Generalization Source -> Destination	CodeGenerationConfiguration	Identifiable
Aggregation Source -> Destination	CodeGenerationConfiguration	TechnicalSafetyRequirement

Class ConfigParameter

Element Base Classes: **Identifiable**

Element Notes:

A ConfigParameter contains the name of the parameter and the value for this parameter.

Connections

Connector	Source	Target
Generalization Source -> Destination	ConfigParameter	Identifiable
Aggregation Source -> Destination	ConfigParameter	CodeGenerationConfiguration
Aggregation Source -> Destination	ConfigParameterValue	ConfigParameter

Class ConfigParameterValue

Element Base Classes:

Element Notes:

A ConfigParameterValue is the abstract element allowing extensions to be made for code generator parameters. In this fashion new parameter value types can be added later on as needed.

Connections

Connector	Source	Target
Generalization Source -> Destination	IdentifierConfigParameterValue	ConfigParameterValue
Generalization Source -> Destination	ConstantComparisonParameter	ConfigParameterValue
Generalization Source -> Destination	IntegerConfigParameterValue	ConfigParameterValue
Aggregation Source -> Destination	ConfigParameterValue	ConfigParameter
Generalization Source -> Destination	FloatConfigParameterValue	ConfigParameterValue
Generalization	BooleanConfigParameterValue	ConfigParameterValue

Connector	Source	Target
Source -> Destination		
Generalization Source -> Destination	StringConfigParameterValue	ConfigParameterValue

Class FloatConfigParameterValue

Element Base Classes: **ConfigParameterValue**

Element Notes:

The FloatConfigParameterValue defines the float parameter value type.

Connections

Connector	Source	Target
Generalization Source -> Destination	FloatConfigParameterValue	ConfigParameterValue

Class IdentifierConfigParameterValue

Element Base Classes: **ConfigParameterValue**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	IdentifierConfigParameterValue	ConfigParameterValue

Class IntegerConfigParameterValue

Element Base Classes: **ConfigParameterValue**

Element Notes:

This element defines a specific parameter value type, in this case an integer value.

Connections

Connector	Source	Target
Generalization Source -> Destination	IntegerConfigParameterValue	ConfigParameterValue

Class StringConfigParameterValue

Element Base Classes: **ConfigParameterValue**

Element Notes:

Connections

Connector	Source	Target
Generalization Source -> Destination	StringConfigParameterValue	ConfigParameterValue

Package System

Package Notes:

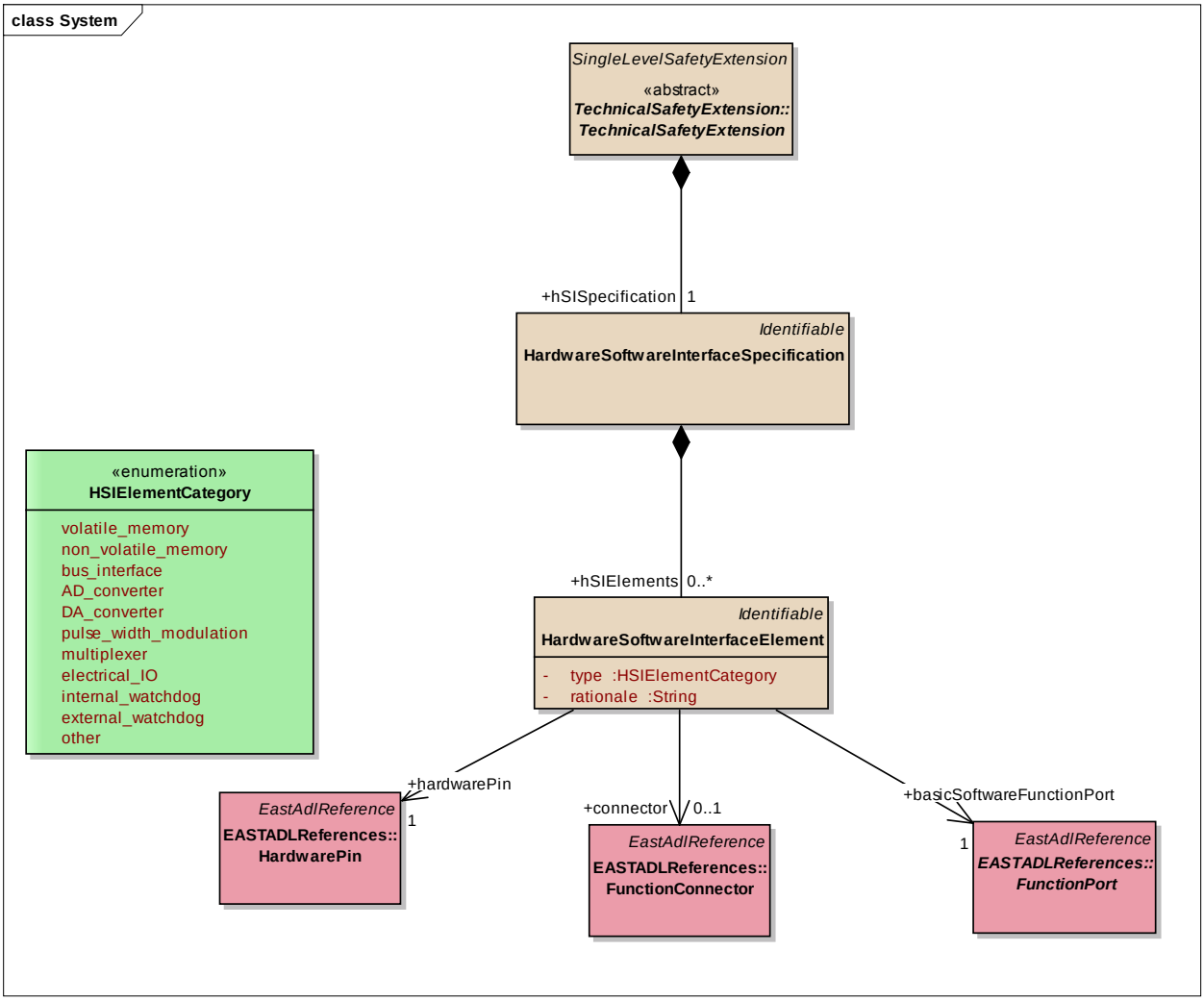


Figure 72: **System** - *(Class diagram)*

Diagram Notes:

Enumeration HSIElementCategory

Element Base Classes:

Element Notes:

Attributes

Attribute	Notes	Default
volatile_memory		
non_volatile_memory		
bus_interface		
AD_converter		
DA_converter		
pulse_width_modulation		
multiplexer		
electrical_IO		
internal_watchdog		
external_watchdog		
other		

Class HardwareSoftwareInterfaceElement

Element Base Classes: **Identifiable**

Element Notes:

An element within the HSI specification.

Constraint: (see EAST-ADL) the associated FunctionConnector has to be a specific connector that connects a BasicSoftwareFunctionType with a HardwareFunctionType.

Note: The FunctionConnector is an optional information of the HSI Element as it can also be derived from the associated HardwarePin and FunctionPort.

Constraint: (see EAST-ADL) the HardwarePin is contained in a HardwareComponentType that is associated with the HardwareFunctionType which port is connected by the forementioned connector.

Constraint: (see EAST-ADL) the FunctionPort has to be associated with a BasicSoftwareFunctionType

Connections

Connector	Source	Target
Association Source -> Destination	HardwareSoftwareInterfaceElement	HardwarePin
Association Source -> Destination	HardwareSoftwareInterfaceElement	FunctionPort
Aggregation Source -> Destination	HardwareSoftwareInterfaceElement	HardwareSoftwareInterfaceSpecification
Generalization Source -> Destination	HardwareSoftwareInterfaceElement	Identifiable
Association Source -> Destination	HardwareSoftwareInterfaceElement	FunctionConnector

Attributes

Attribute	Notes	Default
type HSIElementCategory		
rationale String		

Class HardwareSoftwareInterfaceSpecification

Element Base Classes: **Identifiable**

Element Notes:

The HardwareSoftwareInterface (HSI) specification specifies the hardware and software interaction.

The HSI specification includes the component's hardware devices that are controlled by software and hardware resources that support the execution of software.

The HSI specification shall includes the following characteristics:

a) the relevant operating modes of hardware devices and the relevant configuration parameters;

EXAMPLE 1 Operating modes of hardware devices such as: default, init, test or advanced modes.

EXAMPLE 2 Configuration parameters such as: gain control, band pass frequency or clock prescaler.

b) the hardware features that ensure the independence between elements and that support software partitioning;

c) shared and exclusive use of hardware resources;

EXAMPLE 3 Memory mapping, allocation of registers, timers, interrupts, I/O ports.

d) the access mechanism to hardware devices; and

EXAMPLE 4 Serial, parallel, slave, master/slave.

e) the timing constraints defined for each service involved in the technical safety concept.

Connections

Connector	Source	Target
Aggregation Source -> Destination	HardwareSoftwareInterfaceElement	HardwareSoftwareInterfaceSpecification
Generalization Source -> Destination	HardwareSoftwareInterfaceSpecification	Identifiable
Aggregation Source -> Destination	HardwareSoftwareInterfaceSpecification	TechnicalSafetyExtension

5 References

SAFE Website: www.safe-project.eu

SAFE_D2.1.a-ISO-Part_2.pdf (Management of functional safety)

SAFE_D2.1.a-ISO-Part_3.pdf (Concept Phase)

SAFE_D2.1.a-ISO-Part_4.pdf (Product development at the system level)

SAFE_D2.1.a-ISO-Part_5.pdf (Product development at the hardware level)

SAFE_D2.1.a-ISO-Part_6.pdf (Product development at the software level)

SAFE_D2.1.a-ISO-Part_7.pdf (Production and operation)

SAFE_D2.1.a-ISO-Part_8.pdf (Supporting Processes)

SAFE_D2.1.a-ISO-Part_9.pdf (Automotive Safety Integrity Level (ASIL)-oriented safety-oriented analysis