OCL:
self.assuranceCasePackage->forall(acp|acp.oclIsTypeOf(AssuranceCasePackageInterface)) and
self.argumentPackage->forall(ap|ap.oclIsTypeOf(Argumentation::ArgumentPackageInterface)) and
self.artifactPackage->forall(ap|ap.oclIsTypeOf(Artifact::ArtifactPackageInterface)) and
self.terminologyPackage->forall(tp|tp.oclIsTypeOf(Terminology::TerminologyPackageInterface))

Superclass

Base::ArtifactElement

Associations

assuranceCasePackage: AssuranceCasePackage [0..*] (composition) – a collection of optional sub-packages

interface: AssuranceCasePackageInterface [0..*] – a number of optional assurance case package interfaces that the current package may implement

artifactPackage: ArtifactPackage [0..*] (composition) – a number of optional artifact sub-packages

terminologyPackage: TerminologyPackage [0..*] (composition) – a number of optional terminology sub-packages

argumentPackage:Argument::ArgumentPackage[0..*] (composition) – a number of optional argument packages.

Semantics

AssuranceCasePackage is the root class for creating structured assurance cases.

9.4 AssuranceCasePackageInterface

AssuranceCasePackageInterface is a kind of AssuranceCasePackage that defines an interface that may be exchanged between users. An AssuranceCasePackage may declare one or more ArtifactPackageInterfaces.

Superclass

AssuranceCasePackage

Associations
implements:AssuranceCasePackage[1] – the AssuranceCasePackage that the AssuranceCasePackageInterface declares.

Semantics

AssuranceCasePackageInterface enables the declaration of the elements of an AssuranceCasePackage that might be referred to (cited) in another AssuranceCasePackage, thus the elements can be used for assurance in the scope of the latter AssuranceCasePackage.

Constraints

AssuranceCasePackageInterface are only allowed to contain the following: ArgumentPackageInterfaces, ArtifactPackageInterfaces, and TerminologyPackages.

AssuranceCasePackageInterface,

9.5 ArgumentPackage

ArgumentPackage is a container for the structured argument aspect of the assurance case. It contains the structure of assertions which comprise the structured argument.

Superclass

ArgumentationElement

Associations

argumentPackage: ArgumentPackage [0..*] – an optional set of sub-ArgumentPackages, allowing for recursive containment argumentAsset: ArgumentAsset [0..*] an optional set of ArgumentAssets

. These declarations are provided by containing AssuranceCasePackageInterface(s)/ ArgumentPackageInterface(s)/ArtifactPackageInterface(s)/TerminologyPackageInterface(s) to the packages contained by the AssuranceCasePackage (for which the interface provided).

Semantics

ArgumentPackage is the base class for specifying the results of the argumentation efforts for a structured assurance case (i.e., an AssuranceCase).

9.6 TerminologyPackage

TerminologyPackage is a container element for terminology that may be exchanged. Terminology can define terms, expressions or categories, used elsewhere in the assurance case.

Superclass

TerminologyElement

Associations

terminologyAsset: TerminologyAsset [0..*] – an optional set of terminology assets (expressions, terms and categories)

terminologyPackage: TerminologyPackage [0..*] – an optional set of contained TerminologyPackage elements, allowing for recursive containment.

Semantics

9.5 AssuranceCasePackageBinding

9.4 AssuranceCasePackageBinding

Sub-packages within the AssuranceCasePackage can be bound together by means of AssuranceCasePackageBindings. AssuranceCasePackageBindings bind the participant packages by means of ArgumentPackageBindings/TerminologyPackageBindings/ArtifactPackageBindings elements that bind the contained packages of the participant packages.

Superclass

AssuranceCasePackage

Associations

+participantPackage:AssuranceCasePackage[2..*] – references to AssuranceCasePackages which the AssuranceCasePackageBinding binds together.

Semantics

AssuranceCasePackageBinding binds peer AssuranceCasePackages together to indicate the relationship between these AssuranceCasePackages. The bindings between AssuranceCasePackages consist of the bindings of the packages (i.e. ArgumentPackageBindings, ArtifactPackageBindings and TerminologyPackageBindings) contained in the AssuranceCasePackages, together with an optional ArgumentationPackage that asserts the relationship between +participantPackage.

Constraints

The participantPackages should be either AssuranceCasePackage or AssuranceCasePackageInterfaces

OCL:
self.participantPackage->forall(pp|pp.oclIsTypeOf(AssuranceCase::AssuranceCasePackage) or pp.oclIsTypeOf(AssuranceCase::AssuranceCasePackageInterface))

TerminologyPackage is the base class for specifying all the terminology needs and constraints (via TerminologyAssets) for a structured assurance case (i.e., an AssuranceCase).

9.7 ArtifactPackage

ArtifactPackage is a container element for the assets that are used as evidence or cited in support of a structured argument. These assets form the evidential basis for the assurance case.

Superclass

ArtifactElement

Associations

ArtifactAsset [0..*] – an optional set of ArtifactAsset elements, such as citations, artifacts, resources, activities, etc.

artifactPackage: ArtifactPackage [0..*] – an optional set of contained ArtifactPackage elements, allowing for recursive containment.

Semantics

ArtifactPackage is the base class for specifying and structuring the ArtifactAssets of a structured assurance case (i.e., an AssuranceCase).

10.5 TerminologyPackageInterface
10.6 TerminologyPackageBinding

10.3 TerminologyGroup

TerminologyGroup can be used to associate a number of TerminologyElements to a common group (e.g. representing a common type or purpose, or being of interest to a particular stakeholder).

Superclass

TerminologyElement

Associations

terminologyElement[0..*] – an optional collection of TerminologyElements that are organised within the TerminologyGroup.

Semantics

TerminologyGroup can be used to associate a number of TerminologyElements to a common group (e.g. representing a common type or purpose, or being of interest to a particular stakeholder). The name and the description of the TerminologyGroup should provide the semantic for understanding the TerminologyGroup. TerminologyGroups serve no structural purpose in the formation of the argument network, nor are they meant as a structural packaging mechanism (this should be done using TerminologyPackages).

10.4 TerminologyPackage

The TerminologyPackage is the container element for SACM terminology assets.

Superclass

TerminologyElement

Associations

TerminologyElement:TerminologyElement[0..*] (composition) – TerminologyElements contained in the TerminologyPackage, it can be either TerminologyPackage (and its sub-types) or TerminologyAssets (or its sub-types).

Semantics

TerminologyPackage contains the TerminologyElements that can be used within the naming and description of SACM arguments and artifacts. TerminologyPackages can be nested.

10.7 ~~10.5~~ TerminologyAsset (abstract)

The TerminologyAsset Class is the abstract class for the different types of terminology elements represented in SACM.

Superclass

TerminologyElement

Semantics

TerminologyAssets represent all of the elements required to model and categorize expressions in SACM (expressions and terminology categories).

10.8 ~~10.6~~ Category

The Category class describes categories of ExpressionElements (Terms and Expressions) and can be used to group these elements within TerminologyPackages.

Superclass

TerminologyAsset

Semantics

Terms and ExpressionElements can be said to belong to Categories. Categories can group Terms, Expressions, or a mixture of both. For example, a Category could be used to describe the terminology associated with a specific assurance standard, project, or system.

10.9 ~~10.7~~ ExpressionElement (abstract)

The ExpressionElement class is the abstract class for the elements in SACM that are necessary for modeling expressions.

Superclass

TerminologyAsset

10.5 TerminologyPackageInterface

TerminologyPackageInterface is a kind of TerminologyPackage that defines an interface that may be exchanged between users. An TerminologyPackage may declare one or more TerminologyPackageInterfaces.

Superclass

TerminologyElement

Associations

implements:TerminologyPackage[1] – the TerminologyPackage that the TerminologyPackageInterface declares.

Semantics

TerminologyPackageInterface enables the declaration of the elements of an TerminologyPackage that might be referred to (cited) in another TerminologyPackage, thus the elements can be used for assurance in the scope of the latter AssuranceCasePackage.

10.6 TerminologyPackageBinding

Elements within the TerminologyPackage can be bound together by means of TerminologyPackageBindings. TerminologyPackageBindings bind the participant packages by means of terminology elements that connect the cited elements of the participant packages.

Superclass

TerminologyPackage

Semantics

TerminologyPackageBinding binds TerminologyPackages together to indicate the relationship between two TerminologyPackages.

Constraints

1. The participantPackages should be either TerminologyPackage or TerminologyPackageInterface
   OCL:
   self.participantPackage->forall(pp|pp.oclIsKindOf(Terminology::TerminologyPackage))

packages (through an ArgumentPackageBinding). It is also possible within a package to cite elements contained within other argument packages (through ArtifactReference).

11.3 ArgumentGroup

ArgumentGroup can be used to associate a number of ArgumentElements to a common group (e.g. representing a common type or purpose, or being of interest to a particular stakeholder).

Superclass

ArgumentationElement

Associations

argumentationElement:ArgumentationElement[0..*] – an optional collection of ArgumentationElements organised within the ArgumentGroup.

Semantics

ArgumentGroup can be used to associate a number of ArgumentElements to a common group (e.g. representing a common type or purpose, or being of interest to a particular stakeholder). The name and the description of the ArgumentGroup should provide the semantic for understanding the ArgumentGroup. ArgumentGroups serve no structural purpose in the formation of the argument network, nor are they meant as a structural packaging mechanism (this should be done using ArgumentPackages).

11.4 ArgumentationElement (abstract)

An ArgumentationElement is the top level element of the hierarchy for argumentation elements. ArgumentationElement extends Base::ArtifactElement. Subsequently, all argument elements are considered artifacts.

Superclass

Base::ArtifactElement

Semantics

The ArgumentationElement is a common class for all elements within a structured argument.

11.5 ArgumentPackage Class

The ArgumentPackage Class is the container class for a structured argument represented using the SACM Argumentation Metamodel.

Superclass

ArgumentationElement

Associations

argumentAsset:ArgumentAsset[0..*]

The ArgumentAssets contained in a given instance of an ArgumentPackage.
argumentPackage:ArgumentationPackage[0..*]

The nested argumentPackage contained in a given instance of an ArgumentPackage
interface:ArgumentationPackage[0..*]

Reference to the declared interface for the ArgumentPackage.

Semantics

ArgumentPackages contain structured arguments. These arguments are composed of ArgumentAssets. ArgumentPackages elements can be nested, and can contain citations (references) to other ArgumentPackages.

For example, arguments can be established through the composition of Claims (propositions) and the AssertedInferences between those Claims.

11.6 ArgumentPackageBinding ~~Class~~

~~The ArgumentPackageBinding is a sub type of ArgumentPackage used to record the mapping (agreement) between two or more ArgumentPackages.~~

ArgumentElement within the ArgumentPackage can be bound together by means of ArgumentPackageBinding. ArgumentPackageBinding bind the participant packages by means of argument elements that connect the cited elements of the participant packages.

Superclass

ArgumentPackageBinding

ArgumentPackage

Associations

participantPackage:ArgumentPackageInterface[2..*] - the

~~The~~ ArgumentPackages being mapped together by the ArgumentPackageBinding.

Semantics

ArgumentPackageBindings can be used to map resolved dependencies between the Claims of two or more ArgumentPackages.

For example, one ArgumentPackage may contain a claim that ~~is toBeSupported~~ (i.e. currently has no supporting argument). An ArgumentPackageBinding can be used to record the mapping ~~(~~by means of containing a structured argument linking ~~ArgumentAssetCitations to the claims in question) between this claim and a supporting claim in another ArgumentPackage~~.

~~An ArgumentPackageInterface~~ is a sub type of ArgumentPackage ~~that can be used to create an explicit interface to an existing~~ ArgumentPackage.

Constraints

The 'root' ArgumentAssets contained by an ArgumentPackageBinding (i.e. the ArgumentAssets only associated as target of an AssertedRelationship) and 'leaf' ArgumentAssets (i.e. the ArgumentAssets only associated as source of an AssertedRelationship) must be ArgumentAssetCitations to Claims or ArtifactElementCitations contained within the ArgumentPackages associated by the participantPackage association.

ArgumentElements that cite the claims in question.

needsSupport

, it is used to record the argument that connects the arguments of two or more

s

11.7 ArgumentPackageInterface ~~Class~~

ArgumentPackageInterface is a kind of ArgumentPackage that defines an interface that may be exchanged between users. An ArgumentPackage may declare one or more ArgumentPackageInterfa

Superclass

ArgumentPackage

Associations

implements:ArgumentPackage[1] – a reference to the ArgumentPackage which the ArgumentPackageInterface declares.

Semantics

ArgumentPackageInterfaces can be used to declare (by means of containing ArgumentAssetCitations) the ArgumentAssets contained in an ArgumentPackage that form part of the explicit, declared, interface of the ArgumentPackage.

For example, whilst an ArgumentPackage may contain many Claims, it may be desirable to create an ArgumentPackageInterface that cites only a subset of those claims that are intended to be mapped / used (e.g. by means of an ArgumentPackageBinding) by other ArgumentPackages. There may be more than one ArgumentPackageInterface for a given ArgumentPackage that reveal different aspects of the ArgumentPackage for different audiences.

Constraints

ArgumentPackageInterfaces are only allowed ~~to contain ArgumentAssetCitations to ArgumentAssets within the ArgumentPackage with which this ArgumentPackageInterface is associated (by the interface association)~~

with isCitation=true and +citedElement refer to ArgumentAssets within the ArgumentPackage implementation referred to by implements.

11.8 ArgumentAsset Class (abstract)

The ArgumentAsset Class is the abstract class for the elements of any structured argument represented in SACM.

Superclass

ArgumentationElement

Semantics

ArgumentAssets represent the constituent building blocks of any structured argument contained in an ArgumentPackage.

For example, ArgumentAssets can represent the Claims made within a structured argument contained in an ArgumentPackage.

11.9 Assertion Class (abstract)

Assertions are used to record the propositions of Argumentation (including both the Claims about the subject of the argument and the structure of the Argumentation being asserted). Propositions can be true or false, but cannot be true and false simultaneously.

Associations

metaClaim:Claim[0..*]

references Claims concerning (i.e., about) the Assertion (e.g., regarding the confidence in the Assertion)

Semantics

Constraints

The participantPackages should be only ArgumentPackages

OCL: self.participantPackage->forall(pp|pp.oclIsTypeOf(Argument::ArgumentPackage))

The ArgumentElements contained by an ArgumentPackageBinding must be ArgumentElement citations to ArgumentElements contained within the ArgumentPackages associated by the participantPackage association.

designer could be the owner of the design specification, which would also relate to other artifacts: the requirements specification that satisfies, the architecture that implements, its verification report, etc. Associations between Artifacts and Activities /Events/Participants/ Resources/Techniques, and between Aritfacts and Activities /Events/Participants/ Resources/Techniques Participants can be recorded by means ArtifactAssetRelationships.

12.2 ArtifactPackage

ArgumentPackage is the containing element for artifacts involved in a structured assurance case.

Superclass

Base::ArtifactElement

Associations

artifactElement:Base::ArtifactElement[0..*] (composition) – a collection of ArtifactElements forming a artifact package in a structured assurance case.

Semantics

ArtifactPackages contain ArtifactElements that represent the artifact forming part of a structured assurance case. ArtifactPackages can also be nested.

12.3 ArtifactGroup

ArtifactGroup can be used to associate a number of ArtifactElements to a common group (e.g. representing a common type or purpose, or being of interest to a particular stakeholder).

Superclass

Base::ArtifactElement

Associations

artifactElement:ArtifactElement[0..*] – an optional collection of ArtifactElements organised within the ArtifactGroup.

Semantics

ArtifactGroup can be used to associate a number of ArtifactElements to a common group (e.g. representing a common type or purpose, or being of interest to a particular stakeholder). The name and the description of the ArtifactGroup should provide the semantic for understanding the ArtifactGroup. ArtifactGroups serve no structural purpose in the formation of the argument network, nor are they meant as a structural packaging mechanism (this should be done using ArtifactPackage).

12.4 ArtifactPackageBinding

The ArtifactPackageBinding is a sub type of ArtifactPackage used to record ArtifactAssetRelationships between the ArtifactAssets of two or more ArtifactPackages.

Superclass

ArtifactPackage

Associations

participantPackage:ArtifactPackage~~Interface~~[2..*] **- the**

~~The~~ ArtifactPackages containing the ArtifactAssets being related together by the ArtifactPackageBinding.

Semantics

ArtifactPackageBindings can be used to map dependencies between the cited ArtifactAssets of two or more ArtifactPackages. For example, a binding could be used to record a ‘derivedFrom’ ArtifactAssetRelationship between the ArtifactAsset of one package to the ArtifactAsset of another.

Contraints

ArtifactPackageBindings must only contain ArtifactAssetRelationships with source and target Artifacts, with isCitation = true citing ArtifactAssets contained within the ArtifactPackages associated by participantPackage.

12.5 ArtifactPackageInterface

ArtifactPackageInterface is a kind of ArtifactPackage that defines an interface that may be exchanged between users. ~~A typical use case might be for a component supplier to provide its customers with ArtifactPackageInterfaces that contain the relevant supplier's ArtifactElements for the customers' ArtifactPackages.~~ An ArtfefactPackage may ~~also declare that it implements or conforms to a particular~~ ArtifactPackageInterface.

define one or more

Superclass

ArtifactPackage

Associations

~~artifactAsset: ArtifactAsset [0..*] – an optional set of ArtifactAsset elements, such as citations, artifacts, resources, activities, etc.~~

~~artifactPackage: ArtifactPackage [0..*] - an optional set of contained ArtifactPackage elements, allowing for recursive containment.~~

implements:ArtifactPackage[1] - a reference to the ArtifactPackage which the ArtifactPackageInterface declares.

Semantics

ArtifactPackageInterface enables the declaration of the elements of an ArtifactPackage that might be referred to (cited) in another ArtifactPackage, ~~thus the elements can be used for assurance in the scope of the latter ArtifactPackage~~.

Constraints

ArtifactPackageInterfaces are only allowed to contain Artifacts with +isCitation=true citing ArtifactAssets within the ArtifactPackage with which this ArtifactPackageInterface is associated.

12.6 ArtifactAsset (abstract)

ArtifactAsset represents the artifact-specific pieces of information of an assurance case, in contrast to the argument-specific pieces of information.

Superclass

Base::ArtifactElement

Association

property:Property[0..*] (composition) – an optional collection of Propert(ies) which enable the specification of the characteristics of an ArtifactAsset.

Semantics

Information about artifacts is essential for any assurance case. The artifacts correspond, for instance, to the evidence provided in support of the arguments and claims of an assurance case. It is also important to have access to related pieces of information such as the provenance of an artifact, its lifecycle, and its properties. All this information might have to be consulted for developing confidence in the validity of an assurance case.

12.7 Artifact class

The Artifact class represents the distinguishable units of data used in an assurance case.

Superclass

ArtifactAsset

Attributes

version: String
The version of the Artifact

date: Date
The date on which the artifact was created.

Associations

artifactProperty::ArtifactProperty[0..*]
The ArtifactProperties of the Artifact

artifactEvent::ArtifactEvent[0..*]
The set of ArtifactEvents that represent the lifecycle of the Artifact

Semantics