

11 SACM ~~Argumentation~~ Metamodel

11.1 General

This chapter presents the normative specification for the SACM ~~Argumentation~~ Package. It begins with an overview of the metamodel structure followed by a description of each element.

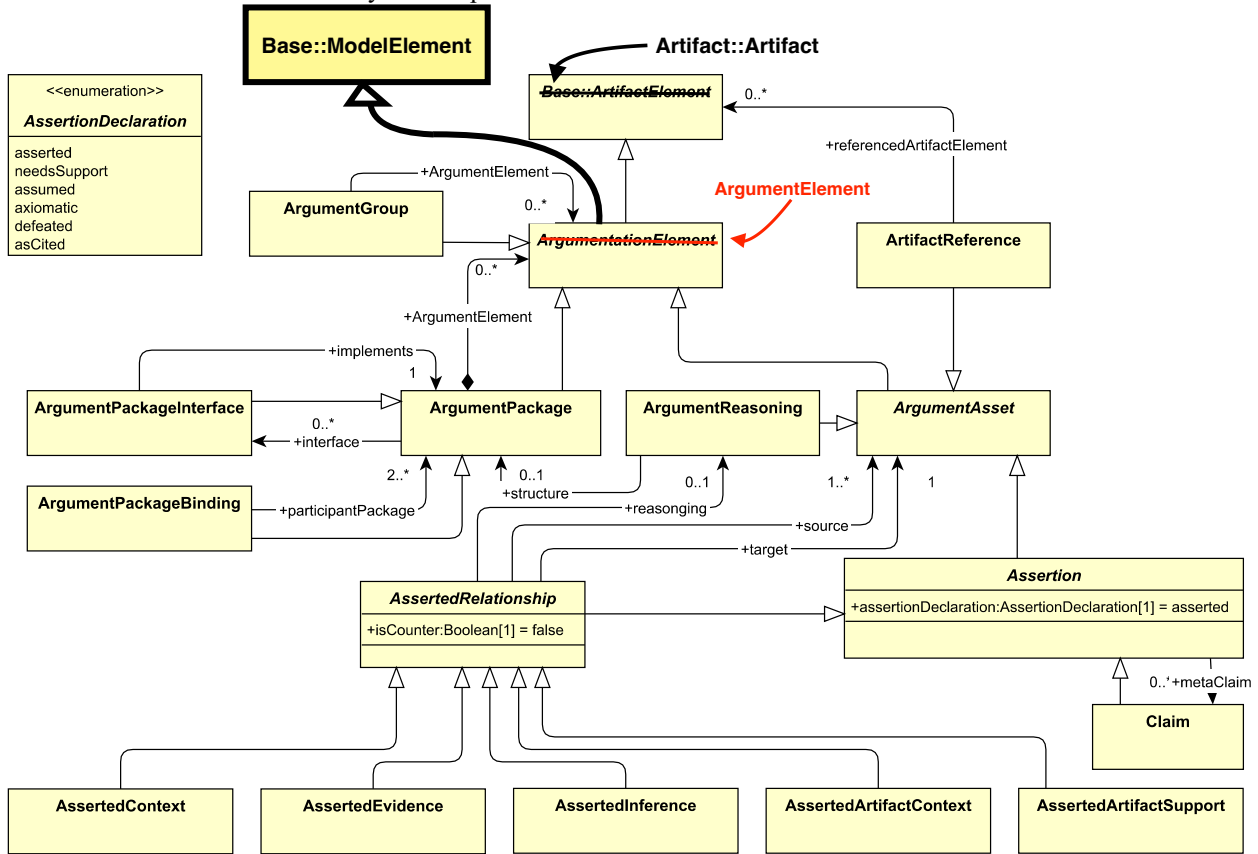


Figure 11.1 - ~~Argumentation~~ Package Diagram

This portion of the SACM model describes and defines the concepts required to model structured arguments. Arguments are represented in SACM through explicitly representing the Claims and citation of artifacts (e.g., as evidence) (ArtifactReference), and the ‘links’ between these elements – e.g., how one or more Claims are asserted to infer another Claim, or how one or more artifacts (referenced by ArtifactReference) are asserted as providing evidence for a Claim (AssertedEvidence). In addition to these core elements, in SACM it is possible to provide additional description of the ArgumentReasoning associated with inferential and evidential relationships, represent counter-arguments and counter-evidence (through isCounter: Boolean), and represent how artifacts provide the context in which arguments should be interpreted (through AssertedContext).

The packaging of structured arguments into ‘modular’ argument packages is enabled through ArgumentPackages. Users are able to declare interfaces for their packages through the use of ArgumentPackageInterface. Within an ArgumentPackageInterface, users create citations of the ~~argumentation~~ elements they select to disclose to external parties. Users are able to integrate ArgumentPackages through the use of ArgumentPackageBinding. An ArgumentPackageBinding binds ArgumentPackages together by including the declared ArgumentPackageInterfaces for the ArgumentPackages, it may contain additional argument structures to provide the rationale of the binding. It is also possible within a package to cite elements contained within other argument packages (through ArtifactReference).

11.2 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).

11.3 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.

11.4 ArgumentPackage Class

ArgumentPackage is the containing element for a structured argument represented using the SACM Argumentation Metamodel.

Superclass

ArgumentationElement

Associations

argumentationElement: ArgumentationElement[0..*] (composition) – a collection of ArgumentationElements forming a structured argument

Semantics

ArgumentPackages contain structured arguments. These arguments are composed of ArgumentAssets. ArgumentPackages elements can also be nested.

Constraints

If an ArgumentPackage has nested ArgumentPackages, then it is only allowed to contain ArgumentPackages.

11.5 ArgumentPackageBinding

ArgumentElement within the ArgumentPackage can be bound together by means of ArgumentPackageBinding. An ArgumentPackage can provide ArgumentPackageInterfaces, which export ArgumentationElements to be used by other ArgumentPackages. ArgumentPackageInterfaces contain citations to ArgumentationElements (e.g. an ArgumentPackageInterface may contain an 'asCited' Claim, with its 'citedElement' pointing to the Claim inside the

ArgumentElements

ArgumentPackage). An ArgumentPackageBinding binds the participant packages (i.e., ArgumentPackageInterfaces) by means of structured arguments, with ArgumentationElements citing the contents inside the participant packages.

Superclass

ArgumentPackage

Associations

participantPackage:ArgumentPackage[2..*] - 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 needsSupport (i.e. currently has no supporting argument). An ArgumentPackageBinding can be used to record the mapping by means of containing a structured argument linkingArgumentElements that cite the claims in question.

ArgumentPackageBinding is a sub type of ArgumentPackage, it is used to record the argument that connects the arguments of two or more ArgumentPackages.

An ArgumentPackageBinding resides within an AssuranceCasePackageBinding. It contains references, using isCitation=True to each ArgumentationElement used in its argument structure that relates ArgumentationElements from different ArgumentPackages.

Constraints

The participantPackages should be only ArgumentPackages

OCL:

self.argumentElement

self.participantPackage->forall(pp|pp.ocIsTypeOf(Argument::ArgumentPackageInterface)) and self.argumentationElement->forall(e|e.isCitation = true and e.citedElement <> null)

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

11.6 ArgumentPackageInterface

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

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 ArgumentElement based citations) 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

ArgumentElements

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.

An ArgumentPackageInterface resides inside the ArgumentPackage to which it refers. It refers to ~~ArgumentationElements~~ using isCitation=True that reside within the same ArgumentPackage as itself. Similar relationships exist for an ArtifactPackageInterface and for a TerminologyPackageInterface.

Constraints

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

11.7 ArgumentAsset (abstract)

ArgumentAsset is the abstract base element for the elements of any structured argument represented in SACM.

Superclass

ArgumentElement

~~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.

11.8 AssertionDeclaration (Enumeration)

AssertionDeclaration provides a list of declarations which can be used to declare the state of an Assertion.

Superclass

N/A

Enumeration Literals

asserted – the default enumeration literal, indicating that an Assertion is asserted.

needsSupport – a flag indicating that further ~~argumentation~~ **argument** has yet to be provided to support the Assertion.

assumed – a flag indicating that the Assertion being made is declared by the author as being assumed to be true rather than being supported by further ~~argumentation~~ **argument**.

axiomatic – a flag indicating that the Assertion being made by the author is axiomatically true, so that no further ~~argumentation~~ **argument** is needed.

defeated – a flag indicating that the Assertion is defeated by counter-evidence and/or ~~argumentation~~ **argument**.

asCited – a flag indicating that because the Assertion is cited, the AssertionDeclaration should be transitively derived from the value of the AssertionDeclaration of the cited Assertion.

Semantics

AssertionDeclaration provides a list of declarations which indicate the state of an Assertion.

11.9 ArtifactReference

ArtifactReference enables the citation of an artifact as information that relates to the structured argument.