Move to Artifact Class

Semantics

ArtifactElement corresponds to the base class for specifying all the identifiable units of data modelled and managed in a structured assurance case effort.

SACMComment

9 ~~8.11~~ 8 Note

This class specifies a generic note that may be associated with ~~a ModelElement~~ an Element. For example a note may include a number of explanatory comments.

Superclass BaseElement

~~SACMCoreElement~~

~~UtilityElement~~

Attributes **body:ExpressionLangString[0..1]{redefines elementName}**

~~body:ExpressionLangString[0..1]~~

AssociationEnds

annotatedElement:Element[0..*] – Elements to which the SACMComment applies

Semantics

SACMComments ~~Notes~~ are used to specify additional (typically optional) generic, unstructured, untyped information about a ModelElement. An example of this kind of information could be a comment about a ModelElement.

NamedValue attributes

9 ~~8.12~~ 9 ~~TaggedValue~~

This class represents ~~a simple key/value pair~~ that can be attached to any element in SACM. This is a simple extension mechanism to allow users to add attributes to each element beyond those already specified in SACM.

Superclass

BaseElement , **Foundation::Namespace**

~~UtilityElement~~

Attributes

value:~~String~~ [0..*] {ordered, nonunique}

~~Associations~~

~~key:MultiLangString[1] (composition) the key of the TaggedValue.~~

Associations

member:NamedValue [0..*] {ordered} - NamedValues owned by this NamedValue

ownedNamedValue [0..*] {subsets ownedMember} - NamedValues owned by this NamedValue

Semantics

NamedValue ~~TaggedValues~~ can be used to specify attributes, and their corresponding values, for ModelElements.

9.10 Group

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

Supertype

~~SACMElement~~ BaseElement , SACMModel

ownedSACMElement [0..*] {subsets ownedMember, subsets member} - SACMElements that are owned by this Group

Association End

~~element: SACMElement [0..*] {ordered} – a collection of SACMElements that comprise a group~~

member:SACMElement [0..*] {redefines element, ordered} - a collection of SACMElements that compromise a group

9.11 ~~9.10~~ IRI

IRI is a international resource identifier is a String and is formated accourding to RFC 3987.

9.12 ~~9.11~~ SACMModel

An abstract subtype of ModelElement which allows one to define a model and/or a pattern.

Supertype

~~String~~

Foundation::String

ModelElement , Argument::Assertion

Attributes

isPattern : Boolean [0..1] = false - if true elements in SACMModel are part of a pattern.

isModel : Boolean [0..1] = false - if true elements in SACMModel are part of a model.

{ordered}

Association

+subModel:SACMModel [0..*] - subModel is a recursive containment of elements in the model.

+elements:SACMElement [0..*] - elements are items in the model.

13 SACM Argument Metamodel

13.1 General

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

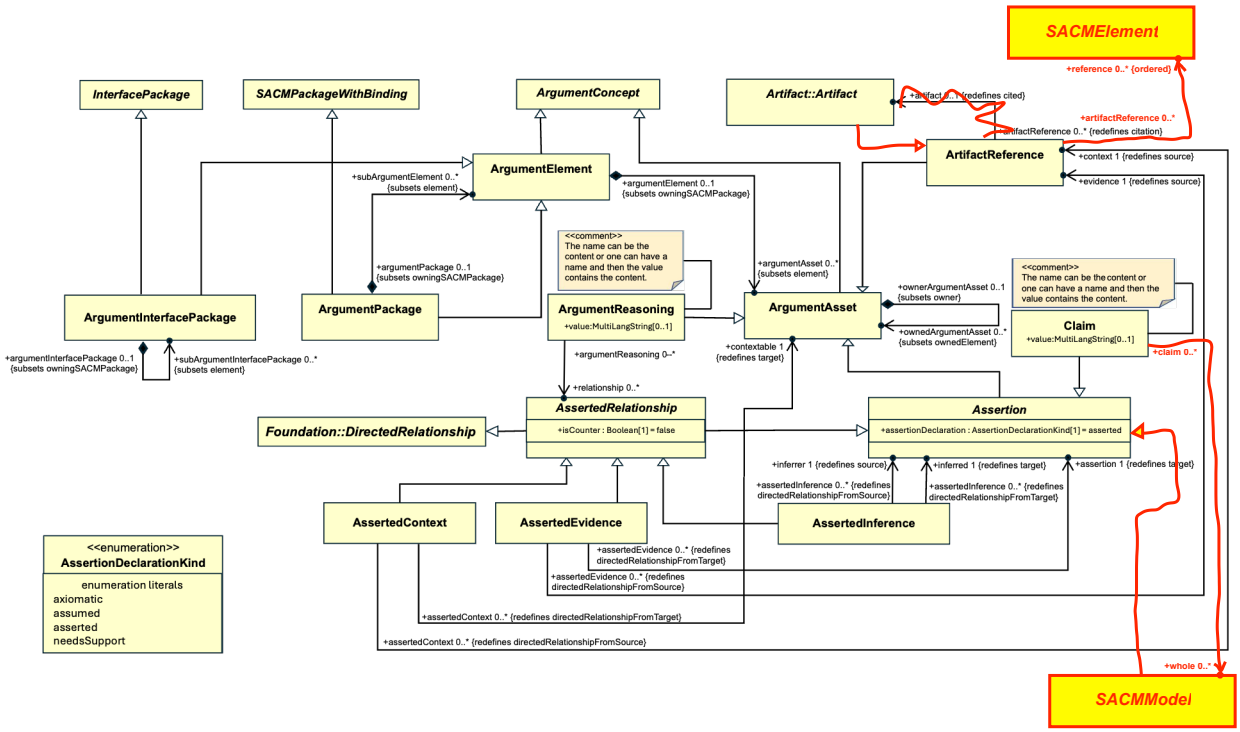

Figure 13.1 – Argument Package Diagram

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 ArgumentInterfacePackage. Within an ArgumentInterfacePackage, users create citations of the argument elements they select to disclose to external parties. Users are able to integrate ArgumentPackages through the use of ArgumentBindingPackage. An ArgumentBindingPackage binds ArgumentPackages together by including the declared ArgumentInterfacePackages 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).

Constraints
MustBeInArgumentPackage
owner.oclKindOf(ArgumentElement)

Superclass

Artifact
~~ArgumentAsset~~

Associations Ends reference:SACMElement [0..*] {ordered} - artifactReference to the SACMElement

~~artifact:Artifact[0..1] {redefine cited}~~ an Artifact.
~~referencedArtifactElement:Base::ArtifactElement[0..*]~~ – reference to ~~a collection of ArtifactElements.~~

Semantics

reference SACMElement(s) to

referencing

It is necessary to be able to ~~cite artifacts that~~ provide supporting evidence, context, or additional description within an argument structure. ArtifactReferences allow there to be an objectified ~~citation~~ of this information within the structured argument~~, thereby allowing the relationship between this artifact and the argument to also be explicitly declare~~d.

13 9
~~12~~ 11.~~10~~ Assertion (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.

Superclass

ArgumentAsset

Attributes

AssertionDeclarationKind[0..1]
assertionDeclaration:~~AssertionDeclaration[1]~~ = asserted – the declaration indicating the state of the Assertion.

~~**Associations**~~

~~metaClaim:Claim[0..*] - references Claims concerning (i.e., about) the Assertion (e.g., regarding the confidence in the Assertion)~~

Semantics

Structured arguments are declared by stating claims, citing evidence and contextual information, and asserting how these elements relate to each other.

13 10
~~12~~ 11.~~11~~ Claim

Claims are used to record the propositions of any structured argument contained in an ArgumentPackage. Propositions are instances of statements that could be true or false, but cannot be true and false simultaneously.

Superclass

Attributes
value:MultiLangString[0..1]

Association Ends
whole:SACMModel [0..*] - the whole structure that this Claim is about

Assertion

The core of any argument is a series of claims (premises) that are asserted to provide sufficient reasoning to support a (higher-level) claim (a conclusion). The name can be the content or one can have a name and then the value contains the content.

Semantics

~~The core of any argument is a series of claims (premises) that are asserted to provide sufficient reasoning to support a~~ (higher- level) claim (a conclusion).

A Claim that is intentionally declared without any supporting evidence or argumentation can be declared as being assumed (i.e., assertionDeclared = assumed). It is an assumption. However, it should be noted that a Claim that is not 'assumed' (i.e., assertionDeclaration = asserted) is not being declared as false. However, there is the expectation of the provision of a supporting argument structure (e.g., it may represent part of an incomplete structure).

A Claim that is intentionally declared as requiring further evidence or argumentation can be denoted by setting +assertionDeclaration to "needsSupport".

A Claim that is being declared as axiomatically true can be denoted by setting +assertionDeclaration to "axiomatic".

A Claim that is defeated by counter evidence or counter argument can be denoted by setting +assertionDeclaration to "defeated".

A Claim which cites another claim and supported by the cited claim can be denoted by setting +assertionDeclaration to "asCited".

The name can be the content or one can have a name and then the value contains the content.

AssuranceCasePackageBinding. It contains references, using isCitation=True to each ArtifactAsset needed and defines relationships among ArtifactAssets from different ArtifactPackages.

Constraints

ArtifactBindingPackages

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

ArtifactInterfacePackage

ArtifactPackageInterface is a kind of ArtifactPackage that defines an interface that may be exchanged between users. An ArtfefactPackage may define one or more ArtifactPackageInterfaces.

ArtifactInterfacePackages

Superclass

, **Package::InterfacePackage**

ArtifactElement

ArtifactPackage

subArtifactInterfacePackage [0..1] {subsets owningSACMPackage} - ArtifactInterfacePackages that are contained in this ArtifactInterfacePackage

ArtifactInterfacePackage

Associations

+asset : ArtifactAsset[0..*]

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. An ArtifactPackageInterface resides inside the ArtifactPackage to which it refers. It refers to ArtifactAssets using isCitation=True that reside within the same ArtifactPackage as itself.

Constraints

All Elements that are cited in a BindingPackage must be contained in either the owner of that BindingPackage or a (recursively) sibling Package of that BindingPackage

inv CitedElementsAreScoped: element->forAll(ele.cited->notEmpty() implies e.cited->closure(glg.owningPackage).asSet()->one(plp=self.owningPackage))

Elements that are not either a Diagram or a BaseElement must be a citation.
inv MustBeCited: element->forAll(elnot e->oclKindOf(SACMDiagram) and not e->oclKindOf(BaseElement) implies e.isCitation=true)

ArtifactPackageInterfaces are only allowed to contain Artifacts with +isC 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

ArtifactConcept

Base::ArtifactElement

Association

Association

ownedArtifactAsset:ArtifactAsset[0..*] {subsets ownedElement} - ArtifactAssets owned by this ArtifactAsset.

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

Artifact represents the distinguishable units of data used in a structured assurance case.

Superclass

ArtifactAsset , **ArtifactReference**