8.5 MultiLangString

MultiLangString, as its name suggests, provides a means to describe things using different languages.

Superclass

Element

Associatiions

value:LangString[1..*] (composition) – contains the descriptions which bear the same meaning but in different languages

Semantics

MultiLangString provides a means to describing things using different languages. It contains a list of LangString, which the user can specify their languages and the descriptions in the languages.

Constraints

Issue: SACM22-14

feature

For each of the LangString in the value property, their +lang must be unique.

8.6 ModelElement (abstract)

ModelElement is the base element for the majority of modeling elements.

Superclass

SACMElement

Associations

name:LangString[1] (composition) – the name of the ModelElement.

implementationConstraint: ImplementationConstraint [0..*] (composition) – a collection of implementation constraints.

description: Description[0..1] (composition) – the description of the ModelElement.

note:Note[0..*] (composition) – a collection of notes for the ModelElement.

taggedValue: TaggedValue [0..*] (composition) – a collection of TaggedValues, TaggedValues can be used to describe additional features of a ModelElement

Semantics

All the individual and identifiable elements of a SACM model correspond to a ModelElement.

Constraints

ImplementationConstraints should only be specified if +isAbstract is true OCL: self.implmentationConstraint->size() > 0 implies self.isAbstract = true

8.7 UtilityElement (abstract)

UtilityElement is the base element for a number of auxiliary elements which can be added to ModelElements.

Superclass

SACMElement

Associations

content:MultiLangString[0..1] (composition) – a MultiLangString to describe the content of the UtilityElement in (possibly) multiple languages

Semantics

ExpressionElements are used to model (potentially structured) expressions in SACM.

10.10 Expression

The Expression class is used to model both abstract and concrete phrases in SACM. Abstract Expressions are denoted by the inherited isAbstract:Boolean attribute being set true. A concrete expression (denoted by isAbstract:Boolean being false) is one that has a literal string value and references only concrete ExpressionElements.

Superclass

ExpressionElement

Associations

element: ExpressionElement [0..*] – an optional reference to other ExpressionElements forming part of the structured Expression.

Semantics

Issue: SACM22-14

Expressions are used to model phrases and sentences. These are defined using the value property. Alternatively, the expression can also be defined (using the value property) as a production rule involving other ExpressionElements. In this case, the value must use a suitable (string) form for denoting the position of involved ExpressionElements (e.g. “$<ExpressionElement.name>$”) within the production rule, and expressing production rule operators (e.g. Extended Backus-Naur Form operators).

Constraints

Issue: SACM22-14

Where an Expression has associated ExpressionElements (the +element property), these should be referenced by name within the +value property.

Where the +value property references ExpressionElement by name, these ExpressionElements should be associated (using the +element property) with Expression.A concrete expression should have references to only concrete ExpressionElements

OCL: self.isAbstract = false implies self.element->forall(expr|expr.isAbstract = false).

10.11 Term

Term is used to model both abstract and concrete terms in SACM. Abstract Terms can be considered placeholders for concrete terms and are denoted by the inherited isAbstract:Boolean attribute being set true. A concrete term is denoted by isAbstract:Boolean being false.

Superclass

ExpressionElement

Attributes

externalReference: String[0..1] – an attribute recording an external reference (e.g., URI) to the object referred to by the Term

Associations

origin:Base::ModelElement[0..1] – a reference which points to the origin of the Term.

Semantics

Term class is used to model both abstract and concrete terms in SACM. Abstract Terms can be considered placeholders for concrete terms and are denoted by the inherited isAbstract:Boolean attribute being set true. A concrete term is denoted by isAbstract:Boolean being false.