ARGEVIDE

Modular Assurance Cases in PREMIS

27.05.2025

1 Introduction to modular assurance cases

Modularisation is helpful in managing large and complex assurance cases. It also allows to distribute responsibilities and manage access rights on the level of single modules. Modularisation is required to enable reuse of arguments.

The breakdown of the argument into modules is usually made in claims. Any claim in the argument may be used as an interface between the modules.

The argument presented on the right refers to the overpressure hazard which is mitigated by opening a safety valve when a sensor signals high pressure.

Claims G1 and G2 refer to the system components, the sensor and the valve. They are good points to cut the argument and separate new modules. The scope of new modules is marked with red frames.

The goal is to develop modular argument presented in Figure 2. The notation used to represent modular arguments will be described in the following sections.

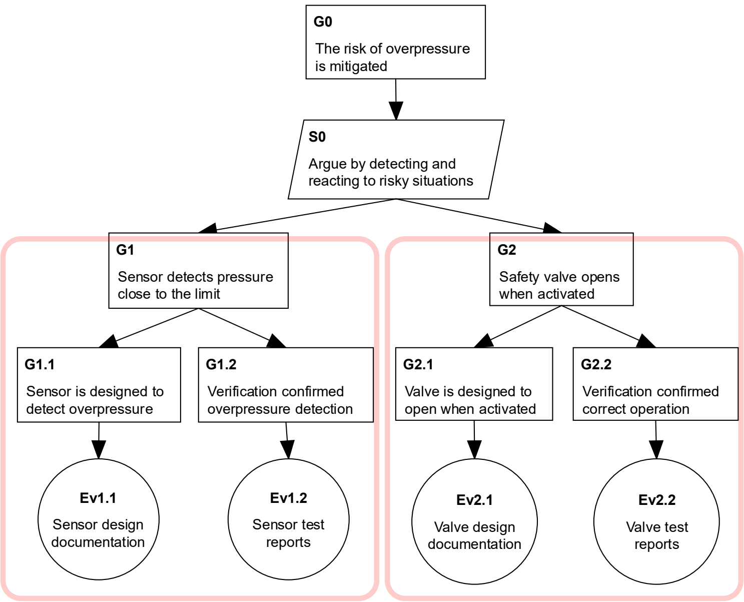

G0
The risk of overpressure is mitigated
S0
Argue by detecting and reacting to risky situations
G1
Sensor detects pressure close to the limit
G2
Safety valve opens when activated
G1.1
Sensor is designed to detect overpressure
G1.2
Verification confirmed overpressure detection
G2.1
Valve is designed to open when activated
G2.2
Verification confirmed correct operation
Ev1.1
Sensor design documentation
Ev1.2
Sensor test reports
Ev2.1
Valve design documentation
Ev2.2
Valve test reports

Figure 1. Assurance case for the overpressure hazard

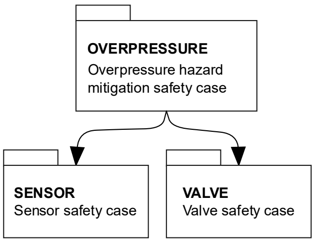


Figure 2. Architecture diagram

The architecture diagram in Figure 2 presents the result of the argument decomposition into three modules.

It contains the top argument module for the mitigation of the overpresure hazard, and two argument modules, one for each component.

The content of the tree modules is presented below:

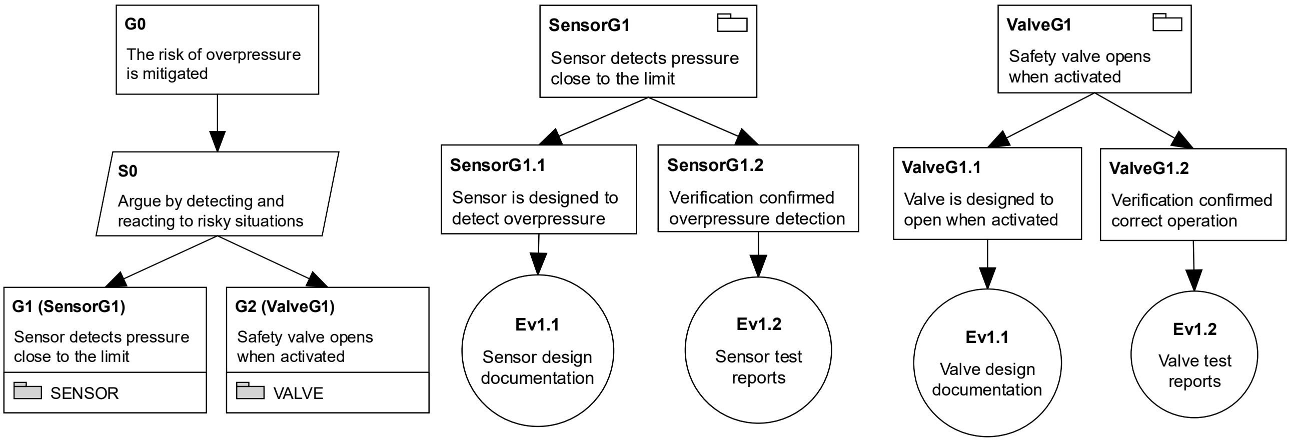

G0
The risk of overpressure is mitigated
S0
Argue by detecting and reacting to risky situations
G1 (SensorG1)
Sensor detects pressure close to the limit
SENSOR
G2 (ValveG1)
Safety valve opens when activated
VALVE
SensorG1
Sensor detects pressure close to the limit
SensorG1.1
Sensor is designed to detect overpressure
SensorG1.2
Verification confirmed overpressure detection
Ev1.1
Sensor design documentation
Ev1.2
Sensor test reports
ValveG1
Safety valve opens when activated
ValveG1.1
Valve is designed to open when activated
ValveG1.2
Verification confirmed correct operation
Ev1.1
Valve design documentation
Ev1.2
Valve test reports

Figure 3. Assurance case decomposed into three modules

2 Types of interfaces

Assurance case modules are connected through interfaces. PREMIS implements argument metamodel based on OMG SACM and this includes interfaces and bindings.

An **interface** is a container of argument elements (claims) made available for other argument modules for connection. There are two main types of interfaces:

- **Required interfaces** contain unsupported claims which require to be supported by claims in other argument modules.
- **Provided interfaces** contain claims which can provide support for other argument modules.

It is not allowed to mix required claims and provided claims in one interface. Separate interfaces are used for claims which require to be supported and for claims which provide support.

A **binding** is a connection between:

- Two interfaces, where one is a required interface and another one is a provided interface.
- Two elements (claims) in the interfaces, where one requires to be supported and another one provides support.

GSN argument and architecture diagrams generally ignore interfaces. They can present bindings if they are defined between the interfaces. Figure 4 presents three GSN diagrams of modules presented in Figure 3 extended with additional symbols for interfaces and bindings between claims in these interfaces. Additional interface for claim G0 has been added to present a situation of a claim which is not bound (yet).

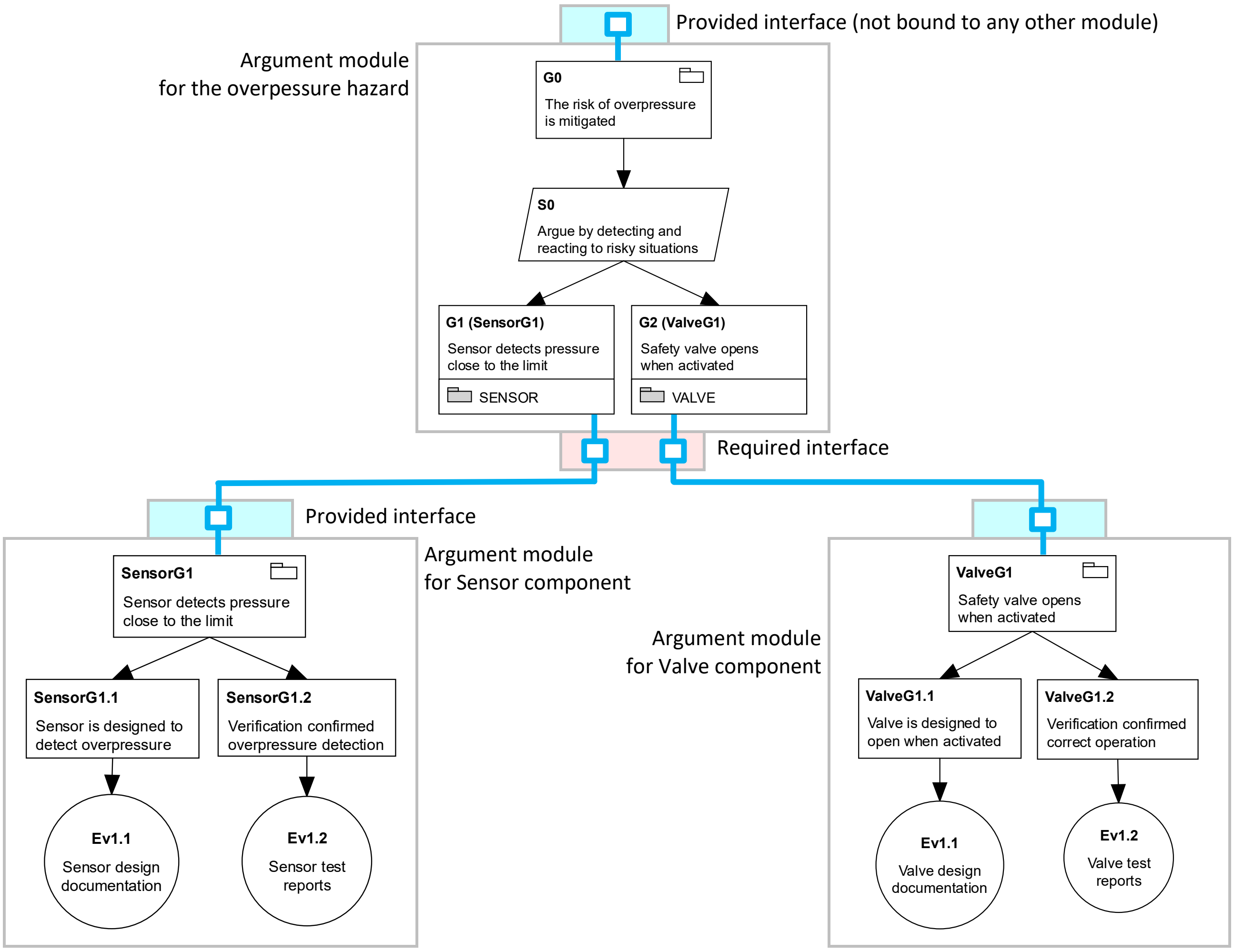

Provided interface (not bound to any other module)
Argument module
for the overpessure hazard
G0
The risk of overpressure is mitigated
S0
Argue by detecting and reacting to risky situations
G1 (SensorG1)
Sensor detects pressure close to the limit
SENSOR
G2 (ValveG1)
Safety valve opens when activated
VALVE
Required interface
Provided interface
Argument module
for Sensor component
SensorG1
Sensor detects pressure close to the limit
SensorG1.1
Sensor is designed to detect overpressure
SensorG1.2
Verification confirmed overpressure detection
Ev1.1
Sensor design documentation
Ev1.2
Sensor test reports
Argument module
for Valve component
ValveG1
Safety valve opens when activated
ValveG1.1
Valve is designed to open when activated
ValveG1.2
Verification confirmed correct operation
Ev1.1
Valve design documentation
Ev1.2
Valve test reports

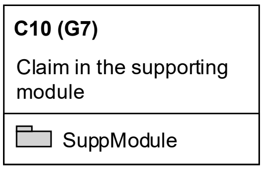
Figure 4. GSN diagrams of three modules extended with interfaces

Figure 4 presents the structure of the interfaces and the binding between elements in the interfaces. When you create a required interface to bind your argument with other supporting module you can chose one of three types of the binding:

- **Away element** – a citation of an element in another module will be created in your argument. This is simply a link to a claim in the supporting module.

  ‘Away element’ means that you directly use an element from another module in your argument module. You will not be able to edit the name and description of an away element as it is taken directly for another module.

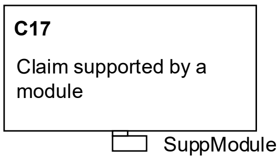
  If the module symbol is empty it means that there is no binding and the element waits to be linked to the supporting module.



- **Supported by another module** – the element in your argument is supported by another element in a supporting module. This type of connection is represented by a module icon decorator below the element.

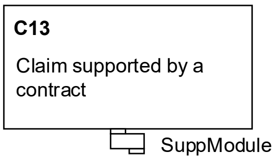
  When the element is supported by another module then its symbol is printed next to the module icon.

  Missing symbol means that there is no supporting module.



- **Supported by a contract** – element is supported by another element in a supporting module through an intermediary contract. The contract is a step of reasoning between the bound modules and it requires to be assessed by the user. The assessment requires verification whether the supporting claim is sufficient to justify the supported claim in its full context.

  The contract is represented by the icon below the element that requires to be supported.



The distinction of three types of bindings between argument modules is compliant with GSN Community Standard. Please refer to Modular Extension in GSN Standards for more information about the types of bindings.

☞ You can often ignore interfaces when you connect claims from different modules. PREMIS can create the interfaces automatically when necessary.

The type of the binding is to be chosen on the side of the supported module, in the required interface. Many interfaces may be connected to one provided interface and they can use different types of bindings.

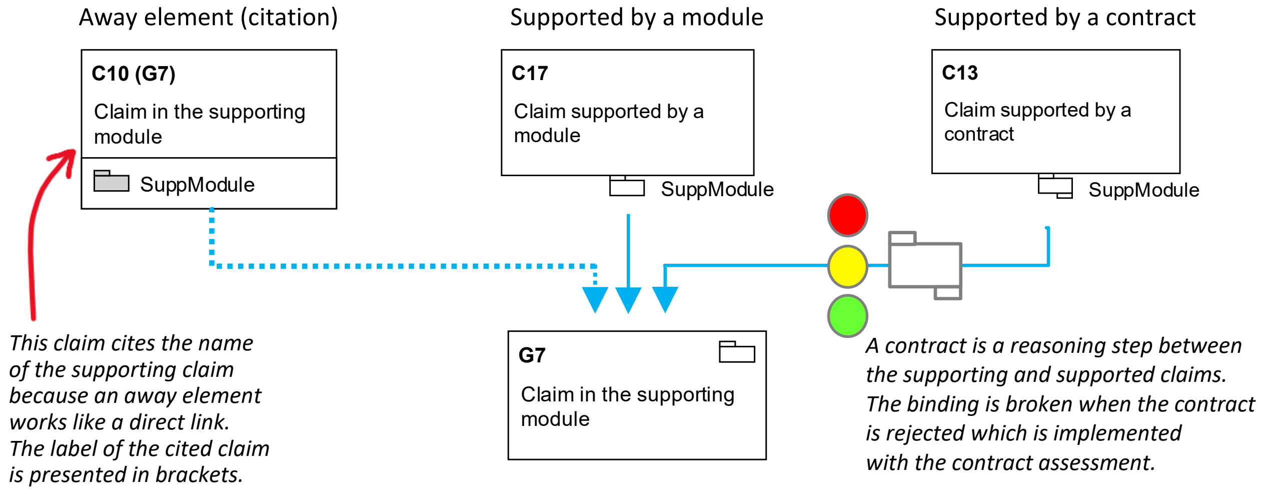

Away element (citation)
Supported by a module
Supported by a contract
C10 (G7)
Claim in the supporting module
SuppModule
C17
Claim supported by a module
SuppModule
C13
Claim supported by a contract
SuppModule
G7
Claim in the supporting module
This claim cites the name of the supporting claim because an away element works like a direct link. The label of the cited claim is presented in brackets.
A contract is a reasoning step between the supporting and supported claims. The binding is broken when the contract is rejected which is implemented with the contract assessment.

Figure 5. Binding implement relation many to one.

How to choose the type of the binding for the required interface?

In short:

- If you need a link to a claim in another module please select **away element**. The name and the description from the source argument will be presented. This binding uses the linking mechanism and when the provided claim is modified you will automatically see its current name and description. Please check names of element in Figure 5 to see the effect of linking.
- If you need to connect two claims you should choose the option **supported by another module**. The binding of this type creates an 'is supported' relation between two claims in different argument modules. This is the simplest way to connect two claims.
- The last option **supported by a contract** extends the binding with the assessment. You should select this option when you want to evaluate is the support provided through the interface is valid for a given required claim. The implementation of the binding is based on an automatically generated reasoning step presented in Figure 6.

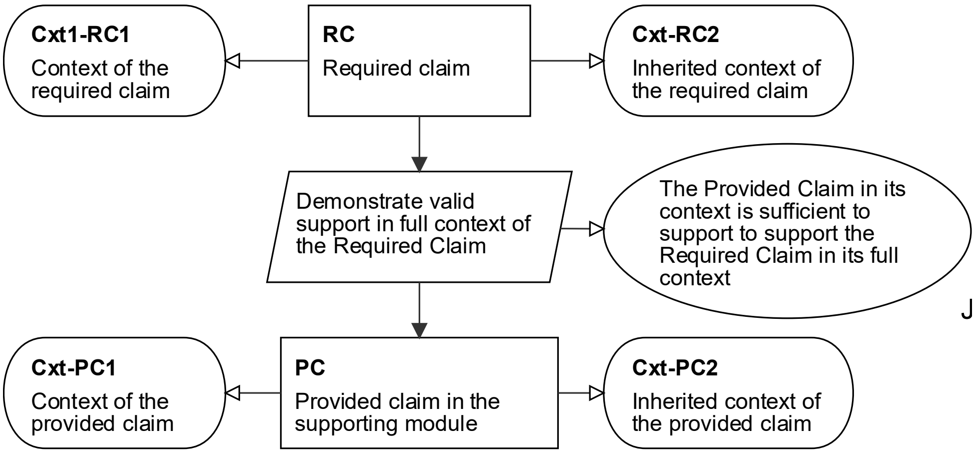

Cxt1-RC1
Context of the required claim
RC
Required claim
Cxt-RC2
Inherited context of the required claim
Demonstrate valid support in full context of the Required Claim
The Provided Claim in its context is sufficient to support to support the Required Claim in its full context
J
Cxt-PC1
Context of the provided claim
PC
Provided claim in the supporting module
Cxt-PC2
Inherited context of the provided claim

Figure 6. Assumed automatic structure of the contract (illustrative diagram)

3 Argument assessment for elements in the interfaces

Claims in required interfaces can be supported by other argument modules and this affects the assessment of these claims. The base rules for the assessment of elements in the interfaces are as follows:

1. The assessment of claims in provided interfaces is made available to all other bound modules.
   a) If the assessment publication mode is set to automatic (which is the default setting), any change of the assessment will be immediately published in the interface.
   b) If the assessment publication mode is set to manual then the user needs to manually run the assessment publication in the interface. Other modules will not get any information about the assessment change until it will be published by the user.

2. The use of the assessment in supported claims in required interfaces depends on the type of the binding:
   a) For away elements: as an away element is a direct link to another module, the assessment is automatically applied for the supported module. Provided interfaces with manual assessment publications are not accepted for away elements.
   b) For claims supported by another module: automatic and manual assessment flow from the supporting module may be applied.
   c) For claims supported by a contract: the assessment flow is controlled by the assessment of the contract. The assessment flow is possible when the contract is accepted. Otherwise, the assessment of the required claim is uncertain.

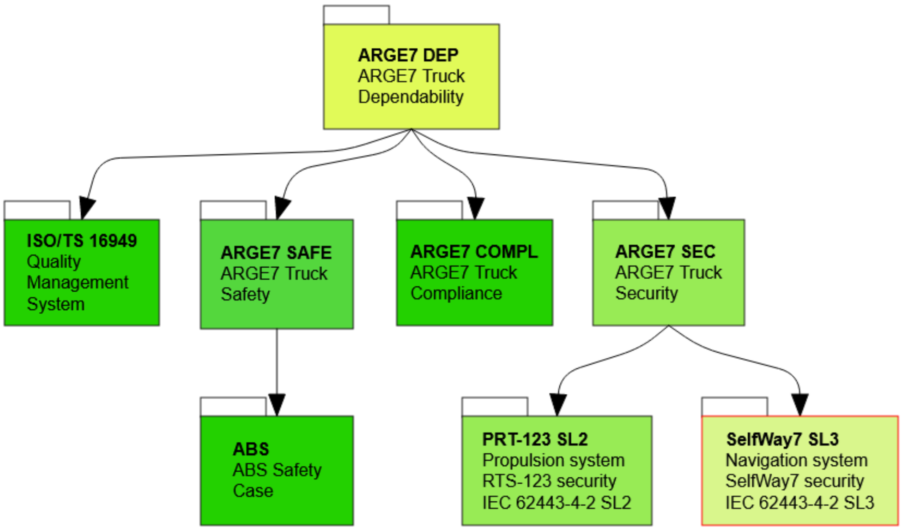

ARGE7 DEP
ARGE7 Truck Dependability
ISO/TS 16949
Quality Management System
ARGE7 SAFE
ARGE7 Truck Safety
ARGE7 COMPL
ARGE7 Truck Compliance
ARGE7 SEC
ARGE7 Truck Security
ABS
ABS Safety Case
PRT-123 SL2
Propulsion system RTS-123 security IEC 62443-4-2 SL2
SelfWay7 SL3
Navigation system SelfWay7 security IEC 62443-4-2 SL3

Figure 7. Assessment presentation in a module diagram

Appendix. PREMIS metamodel relation to OMG SACM

PREMIS metamodel for modular assurance cases is based on the concept of package interfaces and bindings defined in OMG SACM. PREMIS extends the SACM Assurance Case Packages metamodel with the information required for the implementation of types of interfaces and bindings described in this document. The metamodel extension is presented in Figure 8.

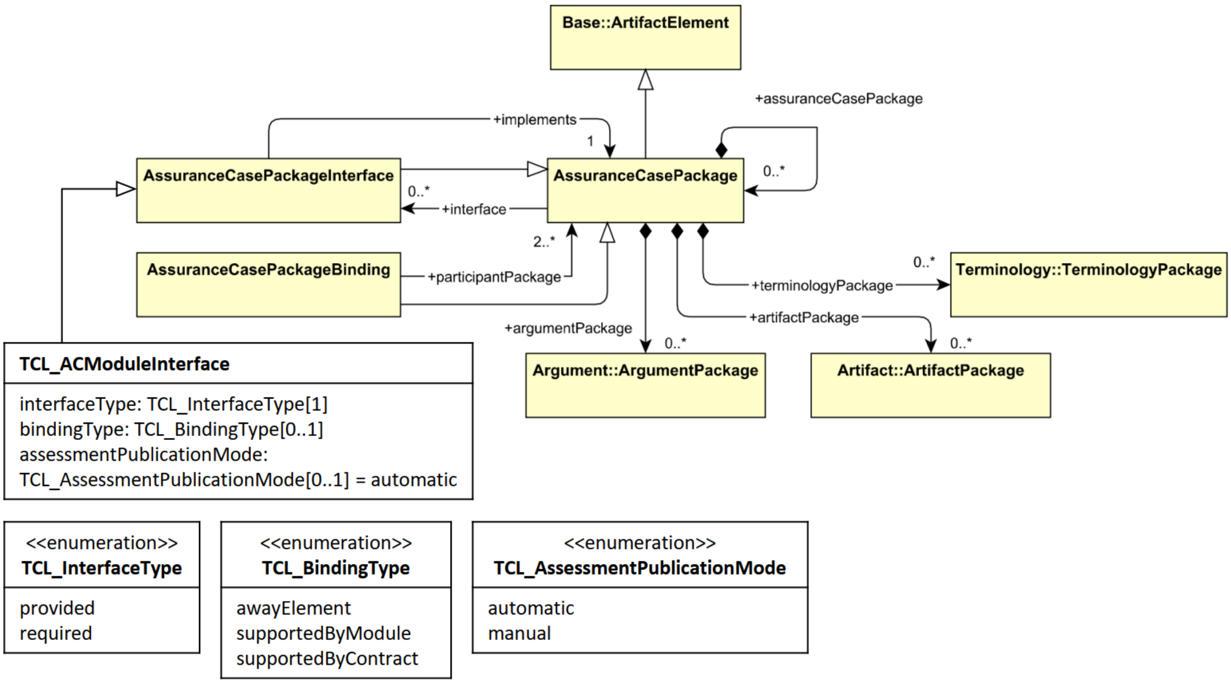


Figure 8. PREMIS extension of OMG SACM Assurance Case Packages metamodel

1. **AssuranceCasePackage** is an assurance case module that contains the argument structure (**ArgumentPackage**) and evidence (**ArtifactPackage**)
2. One or more interfaces can be declared for a module through the use of **TCL_ACPackageInterface**.
   a) Every interface is assigned a type: ‘provided’ or ‘required’. It is not allow in PREMIS to mix different types of connections in one argument interface.
   b) The binding type is declared for each required interface. Provided interfaces have null type of the binding.
3. An interface contains citations to claims in a given argument module. If the interface type is ‘required’ then only the unsupported claims can be used.
   a) A claim in the argument module can be supported either by another argument element inside the module or by a claim in another module through an interface but not by both.
4. **AssuranceCasePackageBinding** defines a binding between two interfaces, one of them is of ‘required’ type and the other one is ‘provided’.
5. **AssuranceCasePackageBinding** contains information about the binding between claims in the bound interfaces:
   a) Citations are used to connect away elements.
   b) Support relations are used to connect claims for the binding type ‘supported by another module’
   c) Argument structures presented in Figure 6 are used for the binding type ‘supported by a contract’
6. **AssessmentPublicationMode** attribute is used for provided interfaces to control the assessment flow between modules. The attribute is not used for required interfaces.

Diagram presented in Figure 8 ignores technical metadata to control the history of changes and versions of argument modules.