



<AutomationML/>

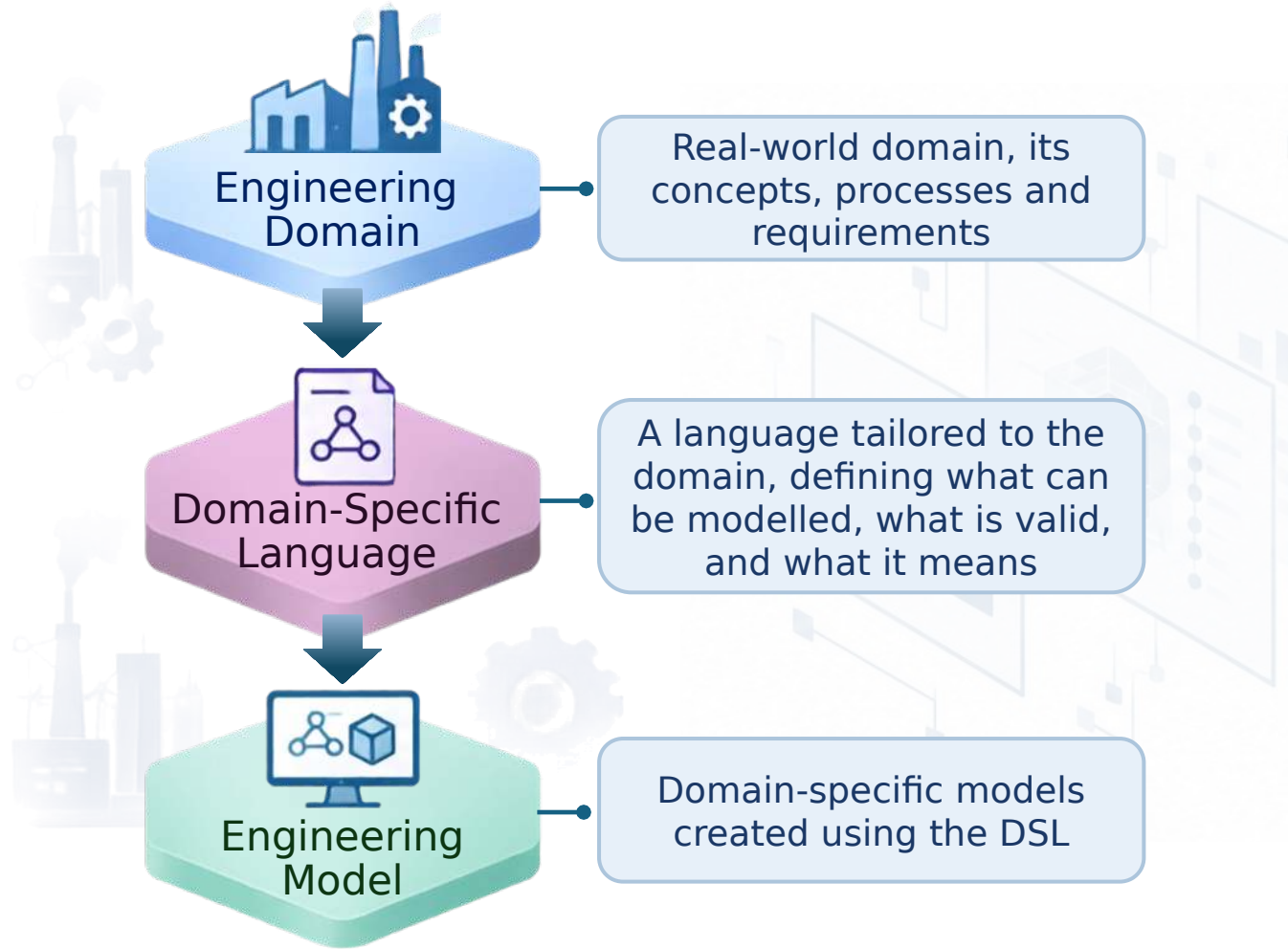
The Glue for Seamless  
Automation Engineering

# **AutomationML as an Exchange Format for Tool-Independent Domain-Specific Modeling Language Specifications**

---

Tirth Kandarp Joshi, Simon Eschlberger, Katharina Polanec, David  
Hoffmann

# Domain-Specific Language (DSL)



## A DSL defines



**What can be modelled?**

- Domain concepts
- Relationship between concepts



**What is valid?**

- Constraints
- Modeling rules

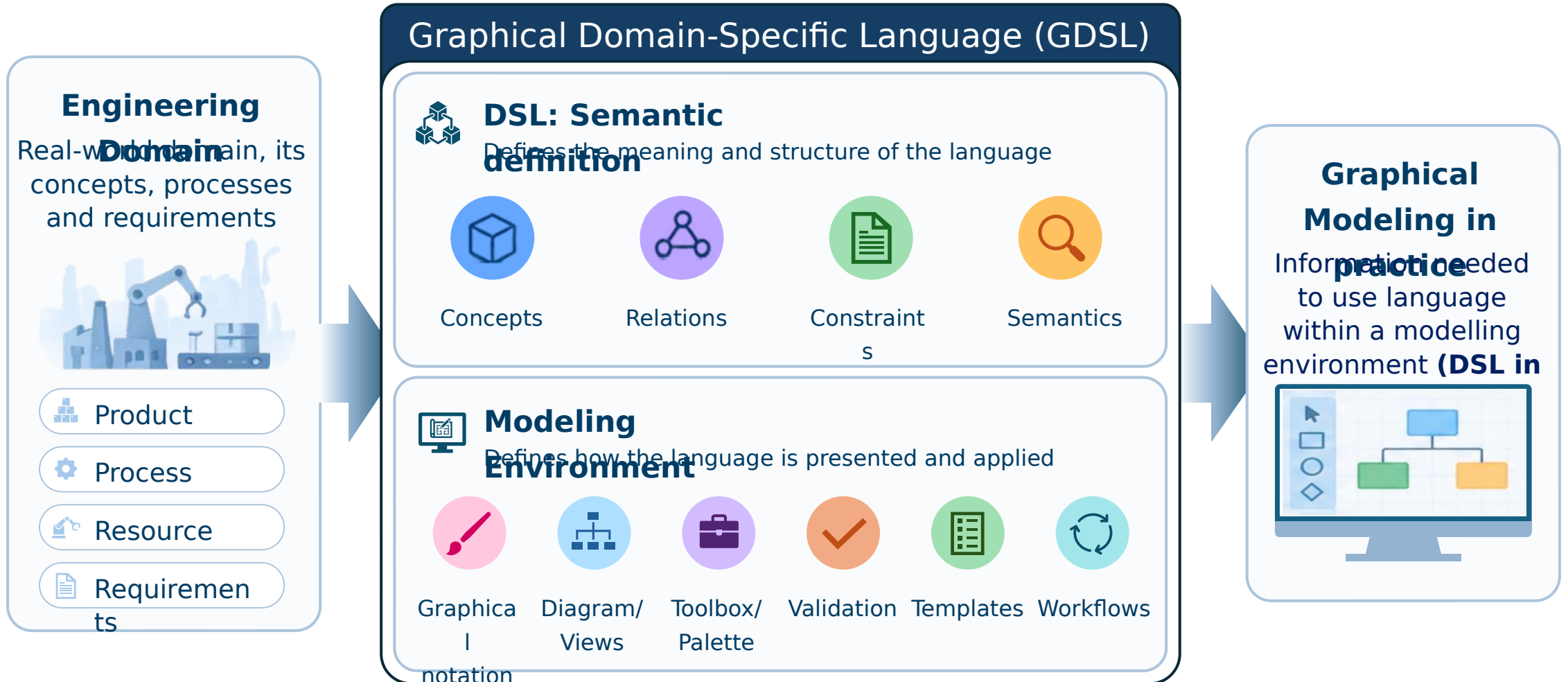


**What does it mean?**

- Domain-specific semantics

A DSL provides domain-specific concepts, relationships and rules for creating models within a particular domain.

# Graphical Domain-Specific Language (GDSDL)

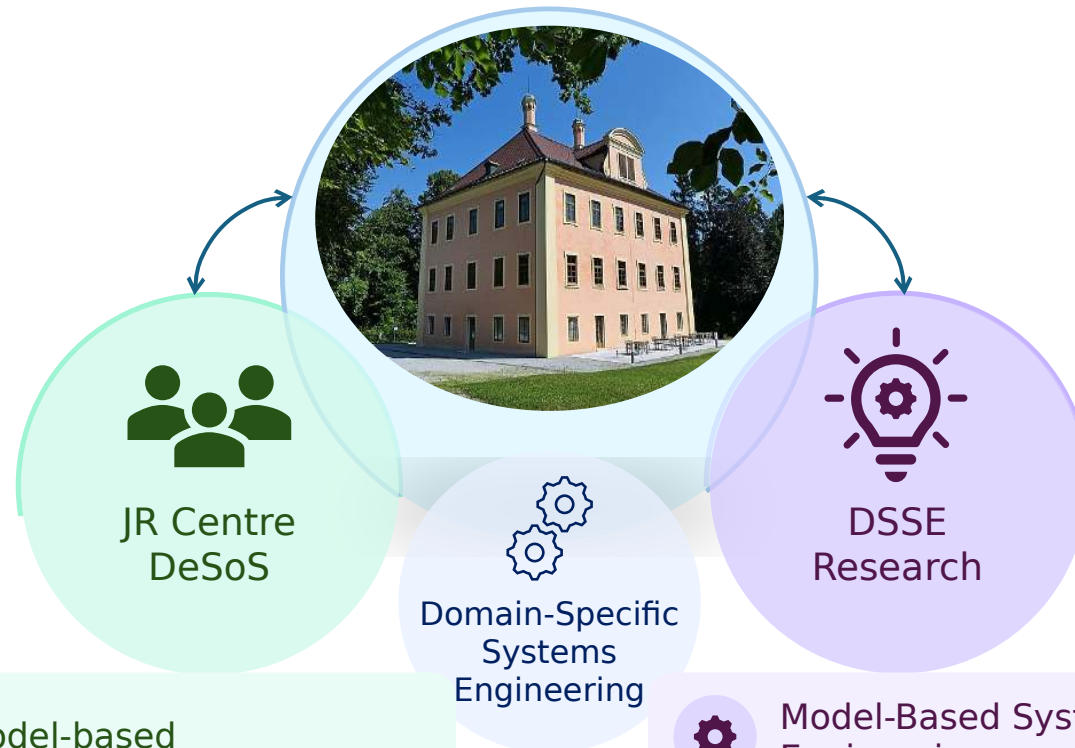


A GDSDL specifies not only what can be modeled, but also how the language is presented and applied.

# FH Salzburg: Domain-Specific Systems Engineering



FH Salzburg



JR Centre  
DeSoS

DSSE  
Research

Domain-Specific  
Systems  
Engineering



Model-based  
development



Complex & distributed  
CPS



System-of-systems  
modeling



Automotive – Smart Grid  
Cybersecurity – Industry  
4.0



Model-Based Systems  
Engineering



Domain-Specific Systems  
Engineering



DSLs & Modeling Toolboxes



Multiple Engineering  
Domains



Systems Engineering  
Community

**TdSE**



TdSE 2025 -  
Salzburg  
1<sup>st</sup> TdSE in Austria



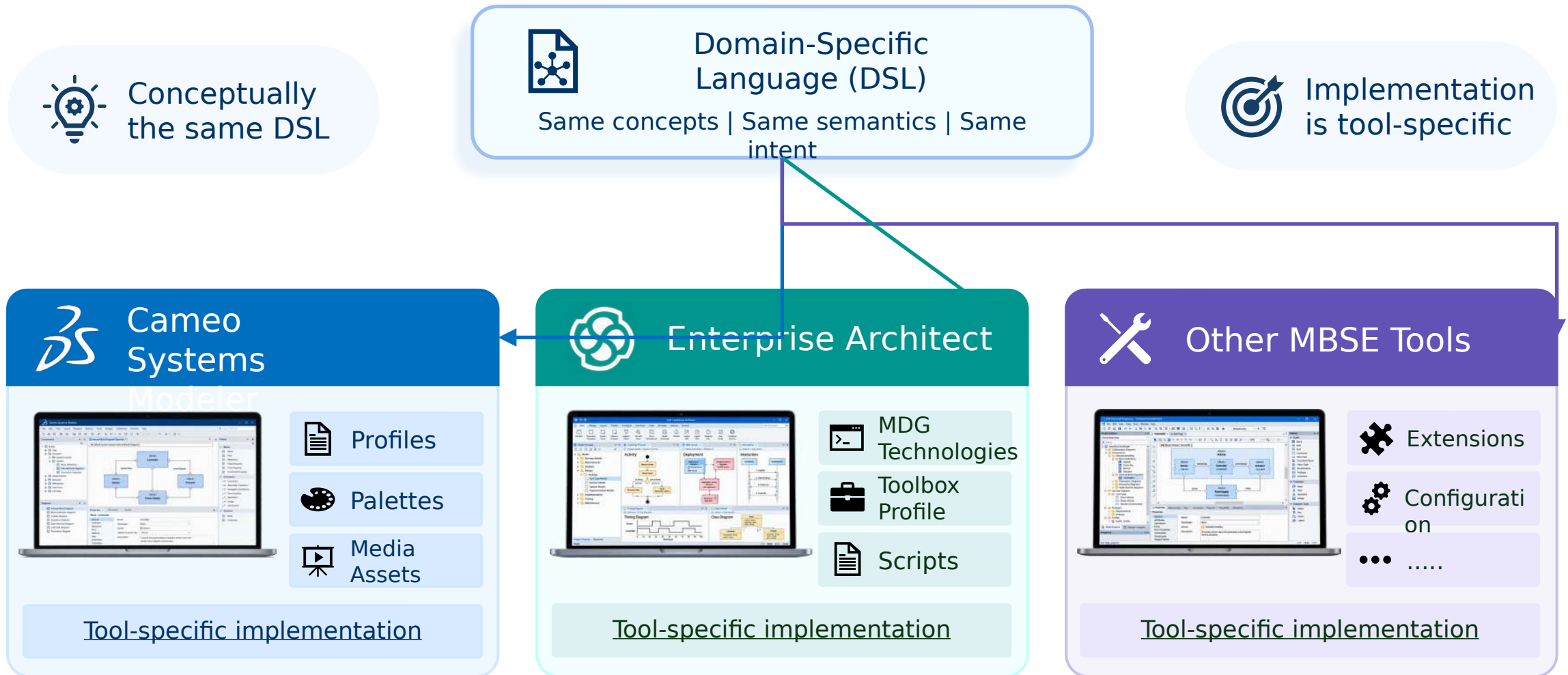
378 participants



SE-NextGen  
Experience

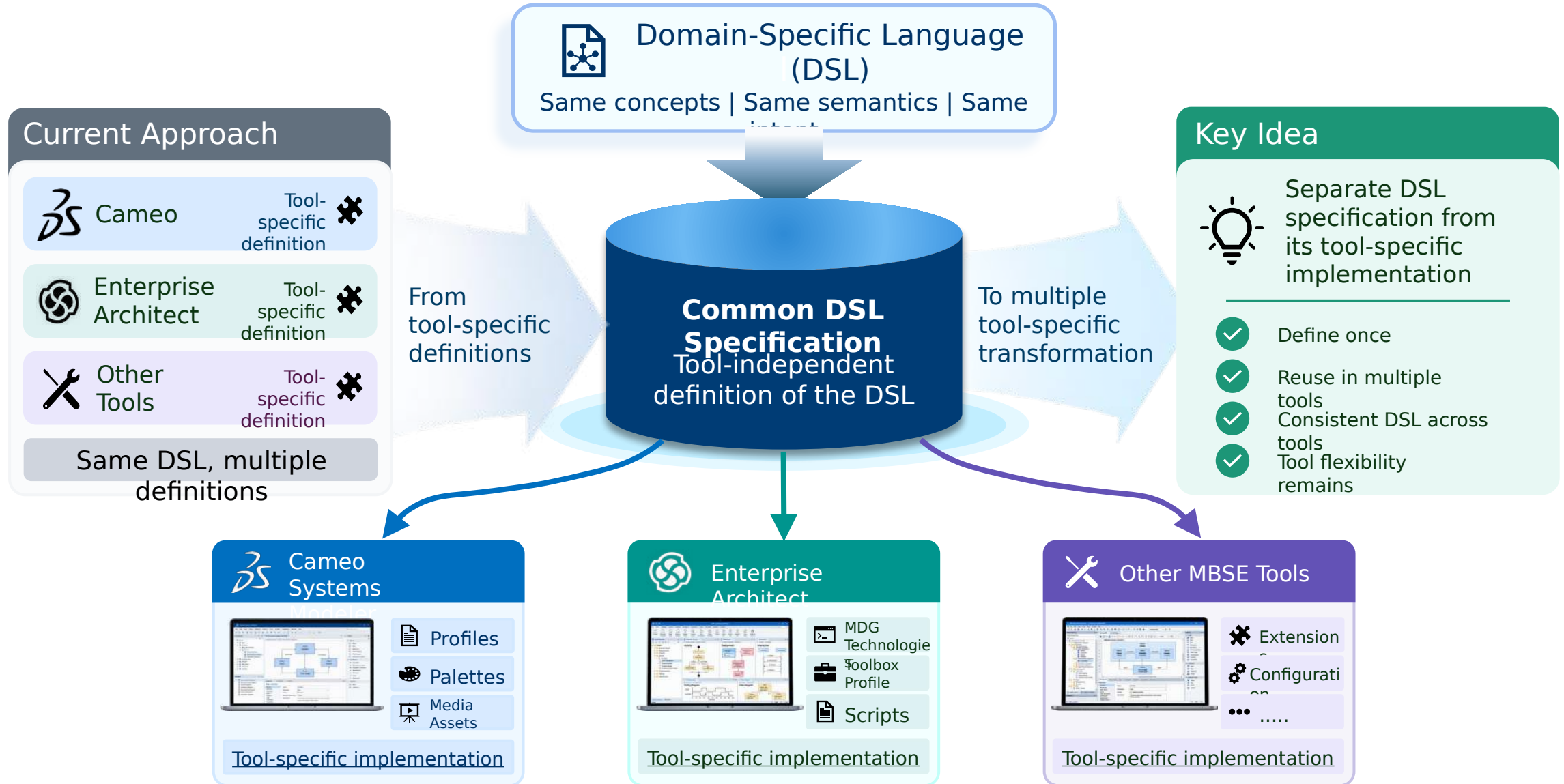


# Problem Observed

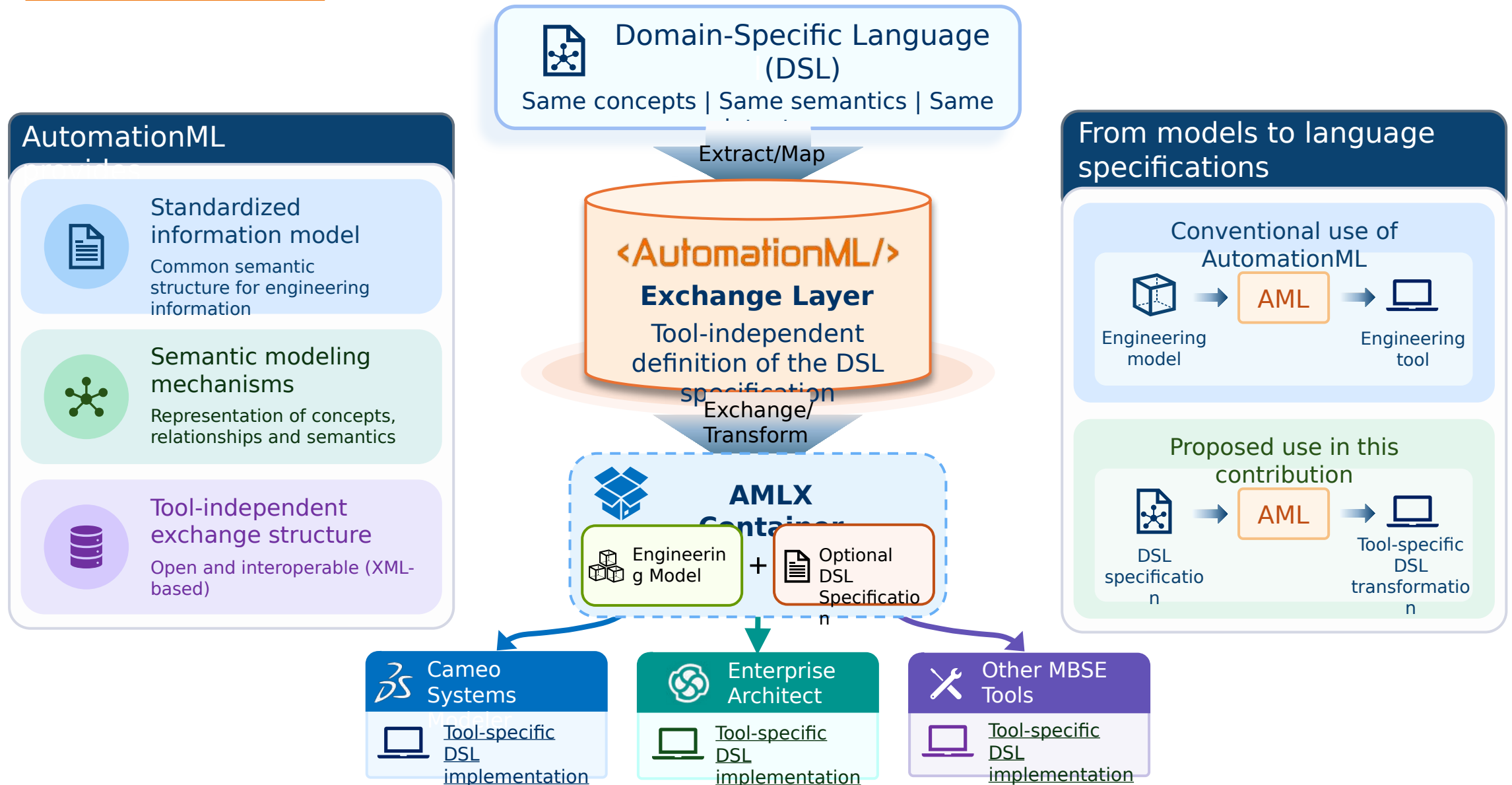


The DSL may be conceptually the same, but its implementation is tool-specific.

# Proposed Idea



# AutomationML as an Exchange Format for DSL Specifications

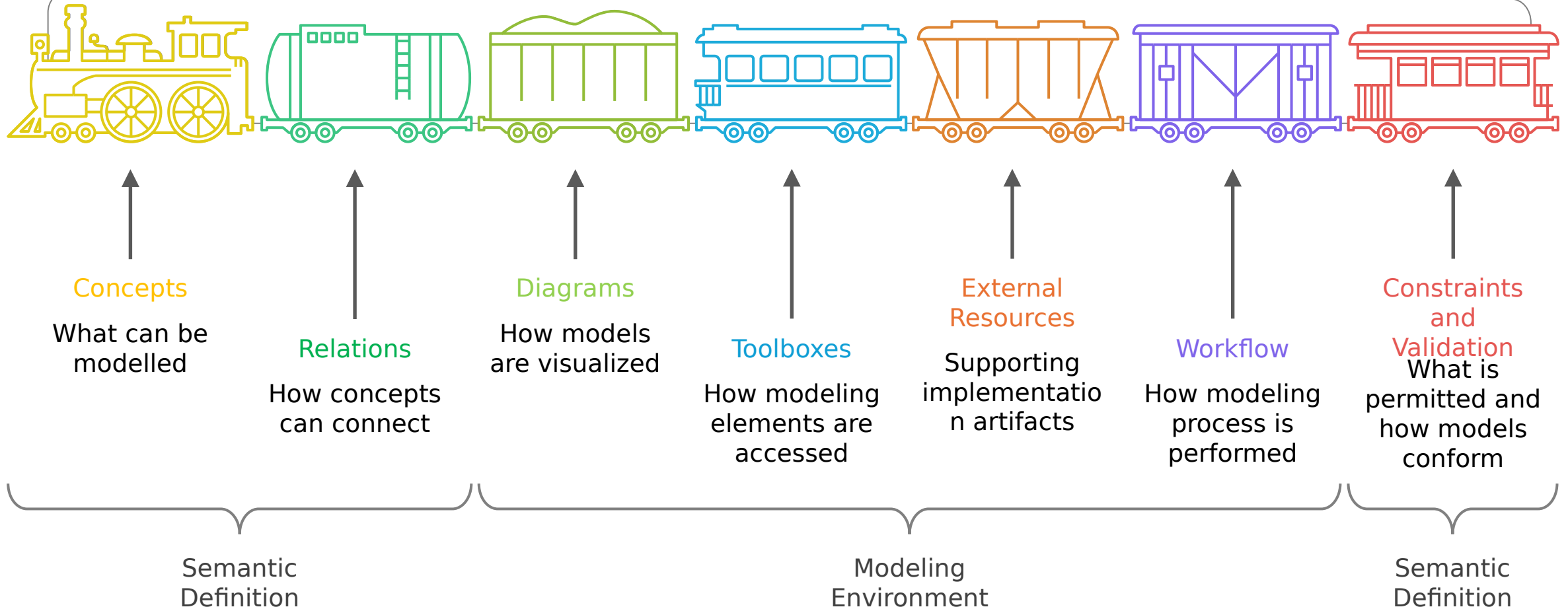


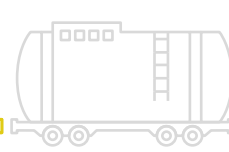
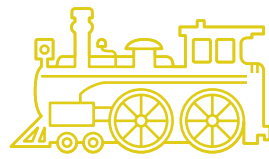
# AML Representation as an Extension, Not a Replacement



A unified AML representation that contains both the engineering instance model and DSL specification as complementary information

# AML DSL EXPRESS





# Concepts

Relations

Diagrams

Toolboxes

External  
Resources

Workflow

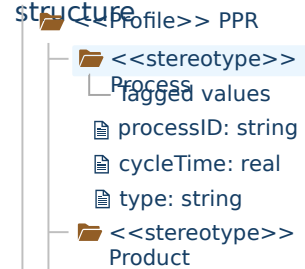
Constraints  
and Validation

## How EA/Cameo represent concepts?



Enterprise Architect  
(MDG)

Model  
structure



UML/SysML profile +  
Stereotype applied to a  
metaclass



Tagged Values define  
domain-specific  
properties

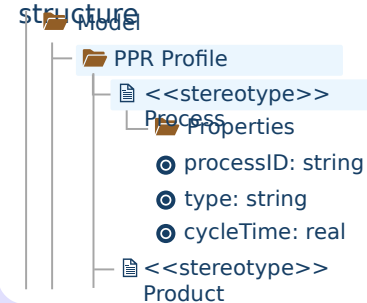


A DSL concept requires both domain-specific identity and a  
reusable structural definition.



Cameo Systems  
Modeler

Model  
structure



UML/SysML profile +  
Stereotype applied to a  
metaclass



Stereotype Properties  
define domain-specific  
properties

## What the neutral DSL specification needs?



Concept meaning  
/identity



Reusable structural  
definition



Domain-specific  
resources



Associated  
Relations/interfaces

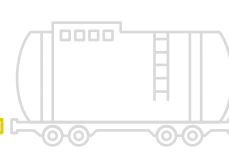
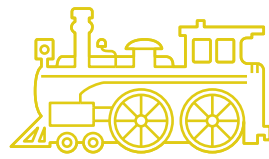


Constraints and validation

What concept means

+

How concept is  
structurally defined



# Concepts

Relations

Diagrams

Toolboxes

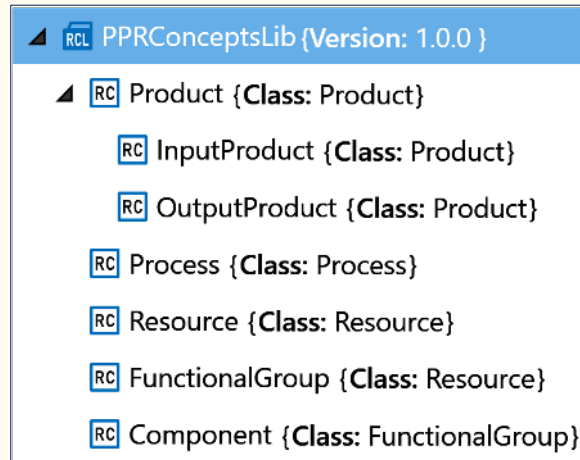
External  
Resources

Workflow

Constraints  
and Validation

## PPRConceptsLib → RoleClassLib

Abstract semantic identity



Domain meaning/  
classification



Semantic  
Hierarchy: Product  
→ Input / Output  
Product



Independent of  
implementation



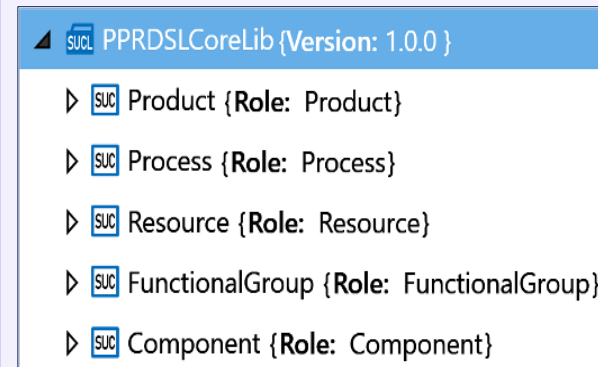
PPRConceptsLi  
b



WHAT  
the concept  
means

## PPRDSLCoreLib → SystemUnitClassLib

Reusable structural definition



Reusable DSL  
concept  
definition



Structural  
definition of  
concept with  
associated RC



Contains:  
Interfaces,  
constraints,  
attributes,



PPRDSLCoreLi  
b

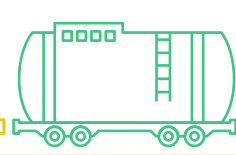
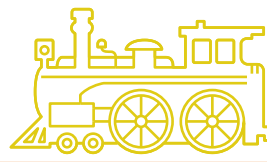


HOW concept is  
structurally  
defined

Role Classes from  
PPRConceptsLib

defines the semantics for

System Unit Classes from  
PPRDSLCoreLib



Concepts

# Relations

Diagrams

Toolboxes

External  
Resources

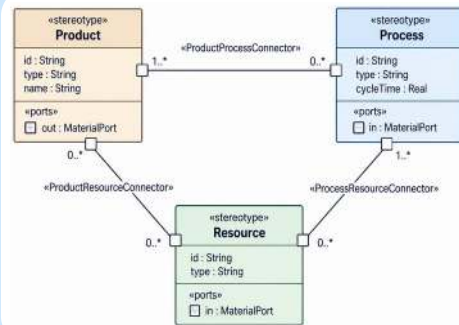
Workflow

Constraints  
and Validation

## How EA/Cameo represent relations?



Enterprise Architect  
(MDG)



Associations /  
Connectors / Ports



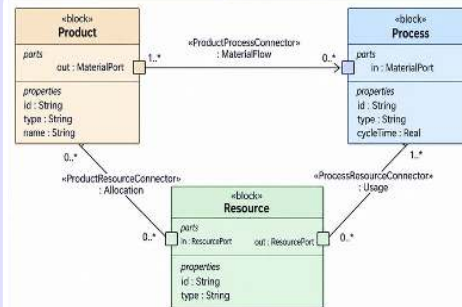
Stereotyped connectors  
and metamodel  
constraints can restrict  
valid connections



A DSL relation requires both a valid connection port and an actual connection link between concepts.



Cameo Systems  
Modeler



Associations /  
Connectors / Ports



UML/SysML Profiles  
constrain the permitted  
modelling structures

## What the neutral DSL specification needs?



Type of relations that  
exists



Where the relation is  
exposed



How the relation is  
exposed



Permitted multiplicity

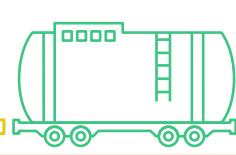
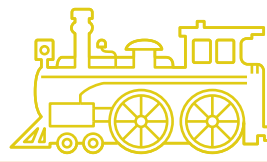


How concepts can be connected

What is connection  
point



How and where is  
connection made



Concepts

# Relations

Diagrams

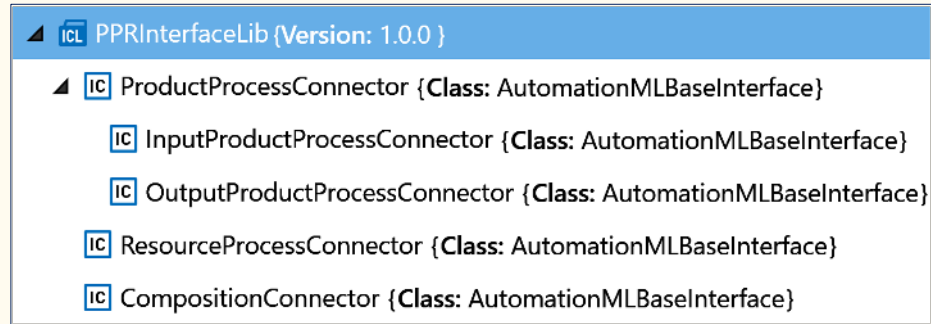
Toolboxes

External  
Resources

Workflow

Constraints  
and Validation

PPRInterfaceLib →  
InterfaceClassLib  
Reusable connector type



Reusable  
interface  
definitions



Defines allowed  
connection  
semantics



Hierarchical  
specialization



PPRInterfaceLi  
b



WHAT  
connector type  
exists

Interface Classes from  
PPRInterfaceLib





Concepts

## Relations

Diagrams

Toolboxes

External  
Resources

Workflow

Constraints  
and Validation

### PPRDSLCoreLib → SystemInitClassLib

```
PPRDSLCoreLib {Version: 1.0.0 }
├─ SUC Product {Role: Product}
│   └─ ProductCompositionConnector{Class: CompositionConnector } 0..n
├─ SUC Process {Role: Process}
│   └─ InputProductProcessConnector{Class: InputProductProcessConnector } 1..n
│   └─ OutputProductProcessConnector{Class: OutputProductProcessConnector } 1..n
│   └─ ResourceProcessConnector{Class: ResourceProcessConnector } 1..n
├─ SUC Resource {Role: Resource}
│   └─ ResourceProcessConnector{Class: ResourceProcessConnector } 1..n
│   └─ ResourceFunctionalGroupDecompositionConnector{Class: CompositionConnector } 0..n
├─ SUC FunctionalGroup {Role: FunctionalGroup}
│   └─ ResourceFunctionalGroupDecompositionConnector{Class: CompositionConnector } 0..n
│   └─ FunctionalGroupComponentDecompositionConnector{Class: CompositionConnector } 0..n
├─ SUC Component {Role: Component}
│   └─ FunctionalGroupComponentDecompositionConnector{Class: CompositionConnector } 1..n
```

### External Interface representation



Port occurrence on DSL  
concept



Concept-specific connection  
point + cardinality



References the InterfaceClass  
via RefBaseClassPath



PPRDSLCoreLib/  
ExternalInterface

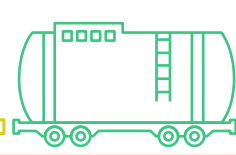
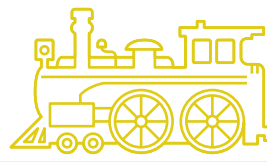


WHERE the connector is  
exposed

Interface Classes from  
PPRInterfaceLib

defines the semantics for

External Interfaces from  
PPRDSLCoreLib



Concepts

Relations

# Diagram

Toolboxes

External  
Resources

Workflow

Constraints  
and Validation

## S

### How EA/Cameo represent diagrams?



Enterprise Architect  
(MDG)

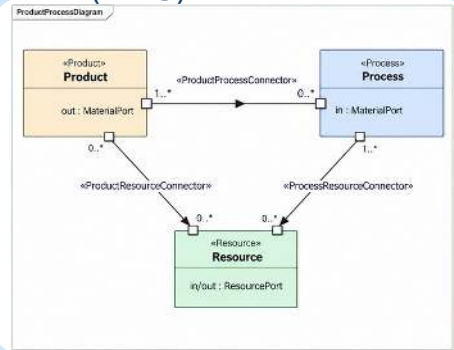


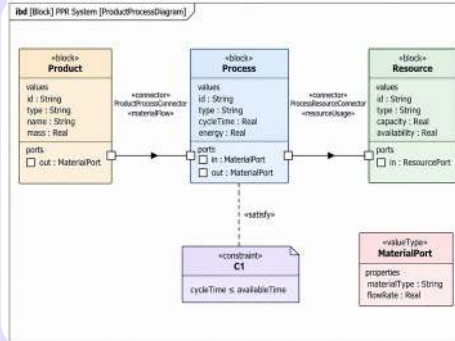
Diagram Profile /  
Custom Diagram Type



Associated  
Metamodel/profile  
constraints



Cameo Systems  
Modeler



Custom Diagram Type  
in Custom Profile



Based on standard  
diagram type with used  
Profiles

### What the neutral DSL specification needs?



Diagram identity



Modelling scope



Diagram-specific  
characteristics



Diagram constraints

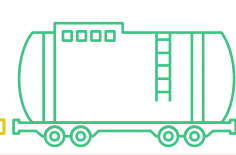
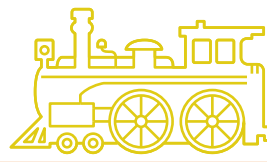


Presentation/configuration  
information

What may be  
modelled?

+

How is it  
presented/configured?



Concepts

Relations

# Diagram

Toolboxes

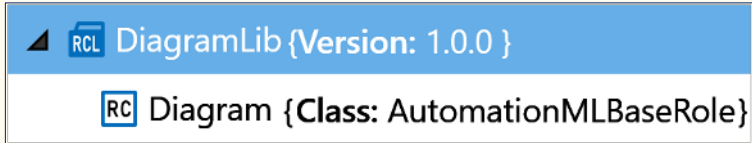
External  
Resources

Workflow

Constraints  
and Validation

S

## DiagramLib → RoleClassLib Semantic Identity of a Diagram



Generic Diagram  
Role



Identifies a meaning  
/type of a Diagram



Acts as the semantic  
role for all specialized  
diagrams in DSL

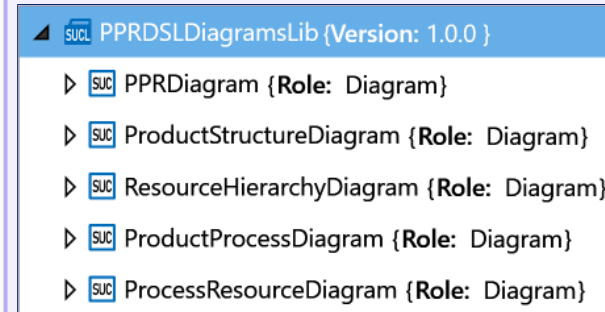


DiagramLib



WHAT  
the diagram  
means

## PPRDSLDiagramsLib → SystemUnitClassLib Reusable DSL diagram definitions



Instantiable DSL  
diagram classes



Structural  
definition of DSL  
diagram with  
associated RC



Contains: diagram  
constraints,  
external resources



PPRDSLDiagramsLi  
b



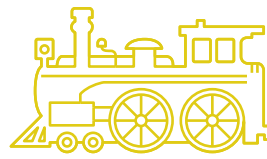
HOW diagrams  
are structurally  
defined

Role Class from  
DiagramLib

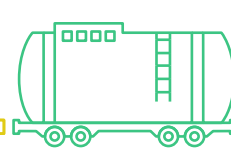
defines the semantics for



System Unit Classes from  
PPRDSLDiagramsLib



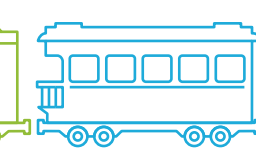
Concepts



Relations



Diagrams



# Toolboxe



External  
Resources



Workflow



Constraints  
and Validation

S

## How EA/Cameo represent toolboxes?



Enterprise Architect  
(MDG)



Toolbox Profile  
associated with a  
Diagram Profile



Provides modelling  
elements available in the  
modelling context



Cameo Systems  
Modeler



Custom Diagram  
Toolbar / Palette  
associated with a  
Custom Diagram Type



Provides diagram-specific  
modelling tools

## What the neutral DSL specification needs?



Toolbox identity



Modelling elements/tools



Associated Diagram type



Toolbox constraints

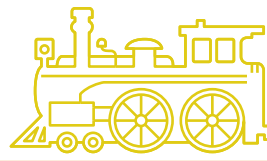


Presentation/configuration  
information

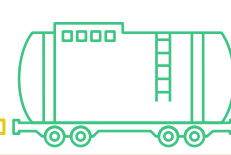
What elements are  
accessible?



How is it  
presented/configured?



Concepts



Relations



Diagrams

# Toolbox



External  
Resources



Workflow

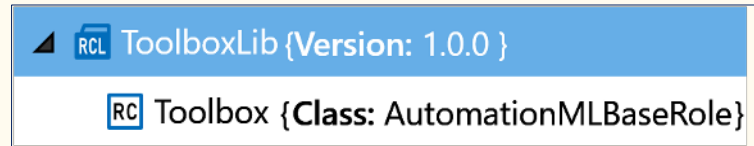


Constraints  
and Validation

S

## ToolboxLib → RoleClassLib

Semantic Identity of a Toolbox



Generic Diagram  
Role



Identifies a meaning  
/type of a Toolbox



Acts as the semantic  
role for all specialized  
toolboxes in DSL



ToolboxLib



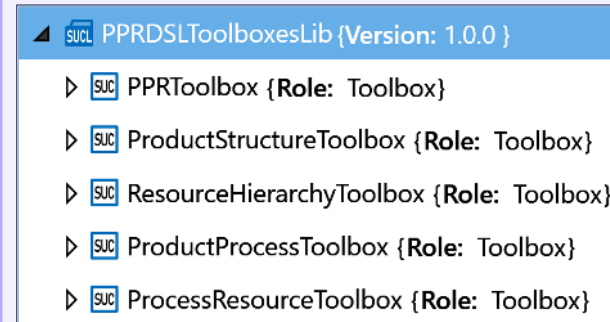
WHAT  
the toolbox  
means

Role Class from  
ToolboxLib

defines the semantics for

## PPRDSLToolboxesLib → SystemUnitClassLib

Reusable DSL toolbox definitions



Instantiable DSL  
toolboxes classes



Structural  
definition of DSL  
toolbox with  
associated RC



Contains: toolbox  
constraints,  
external resources

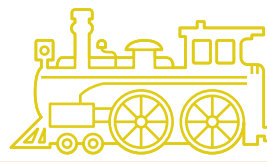


PPRDSLToolboxesLib

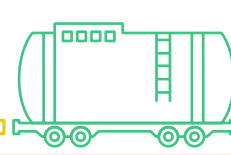


HOW toolboxes  
are structurally  
defined

System Unit Classes from  
PPRDSLToolboxesLib



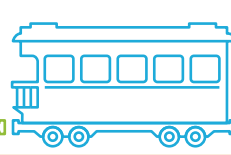
Concepts



Relations



Diagrams



Toolboxes



# External Resource



Workflow



Constraints and Validation

## How EA/Cameo represent external resources?



Enterprise Architect (MDG)



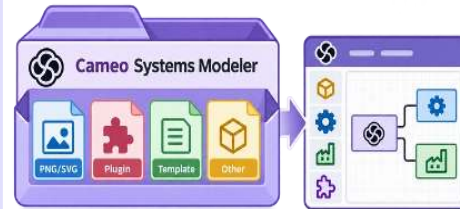
Shape Scripts, images, icons/logos packaged with an MDG Technology



Tool-specific presentation remains part of the tool environment



Cameo Systems Modeler



Media assets such as .png / .svg and supporting plugin/template resources



Tool-specific resources support diagram and modelling presentation



The important point is that these are supporting artifacts, not the semantic definition itself

S

## What the neutral DSL specification needs?



Resource identity/classification



Resource reference



Resource type



Resource location/URI

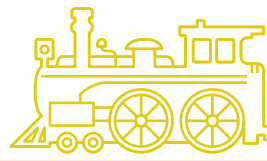


Artifact in which resource is to be embedded

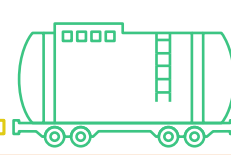
What may be modelled?

+

How is it presented/configured?



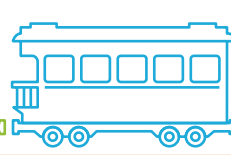
Concepts



Relations



Diagrams



Toolboxes



# External Resource



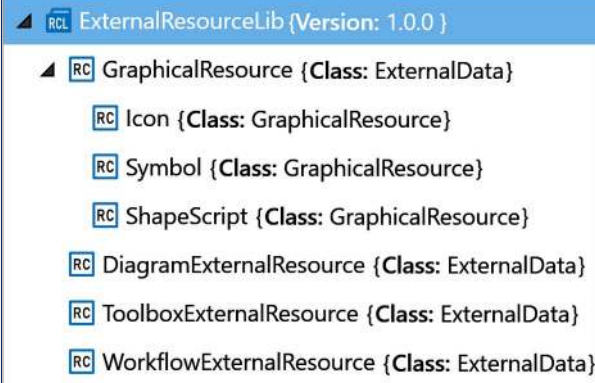
Workflow



Constraints and Validation

S

ExternalResourceLib →  
RoleClassLib  
Semantic Identity of an External Resource



Generic External resource Role



Identifies a meaning /type of a External Resource



ExternalResourceLib

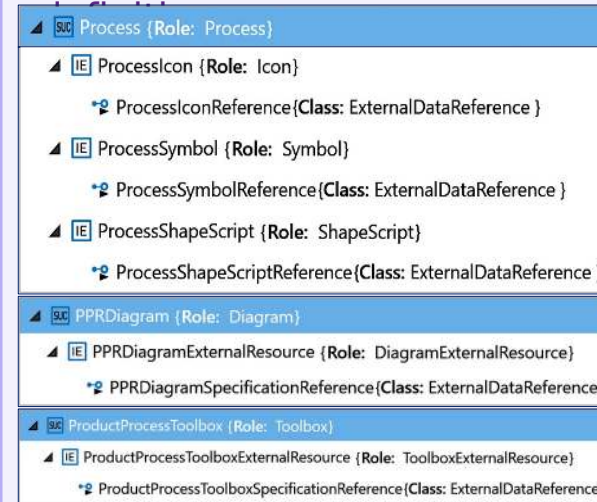


WHAT  
external resource  
is (+ type)

Role Classes from  
ExternalResourceLib

defines the semantics for

PPRDSLSUCs → InternalElement +  
ExternalInterface  
Reusable DSE External Resource



IE: represented  
associated  
resource



EI: references the  
external resource  
data file



Contains: refURI +  
MIMETYPE

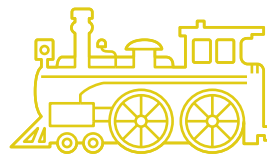


IEs + EIs in SUCs

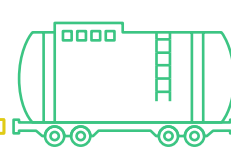


HOW external  
resources are  
structurally embedded

Internal Elements in PPRDSLSUCs



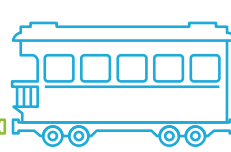
Concepts



Relations



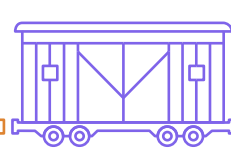
Diagrams



Toolboxes



External  
Resources



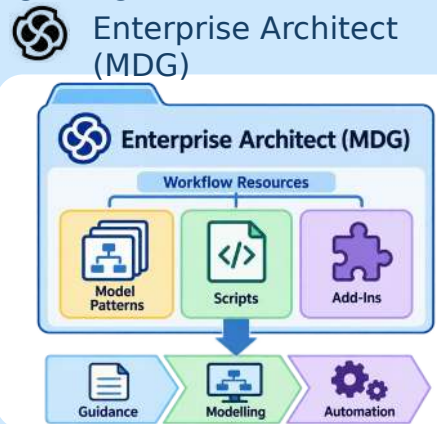
Workflow



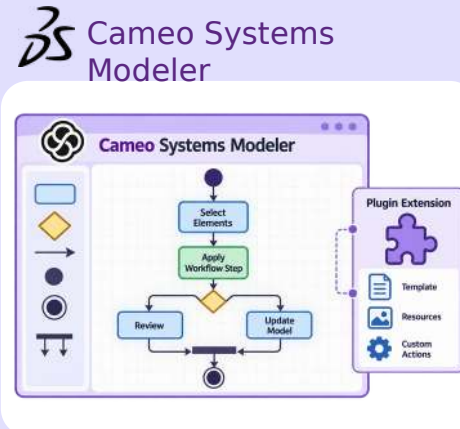
Constraints  
and Validation

W

## How EA/Cameo represent workflow?

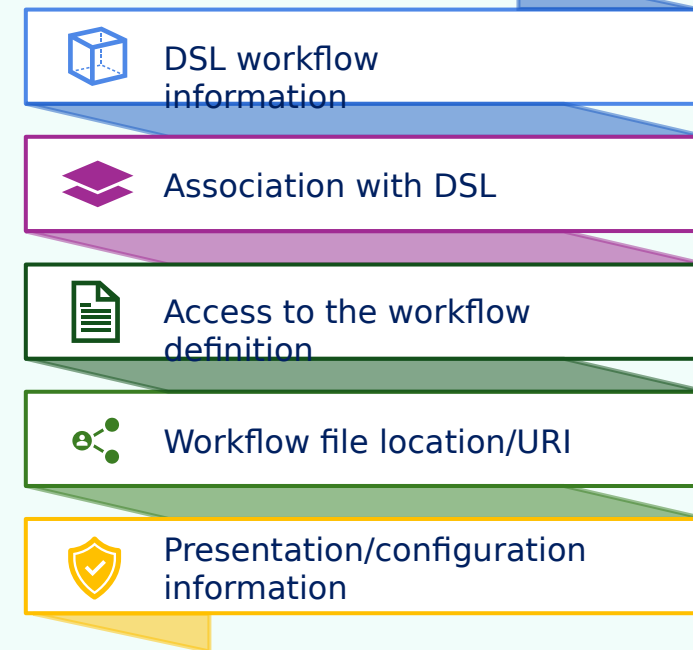


- Model Patterns / workflow resources / scripts or Add-In-based processes
- Can combine modelling guidance with tool-specific automation



- UML Activity-based modelling or plugin-based extensions
- Behavioural workflow can be represented through extensions

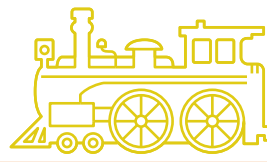
## What the neutral DSL specification needs?



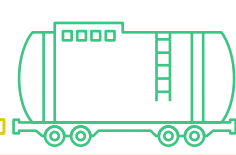
What may be modelled?

+

How is it presented/configured?



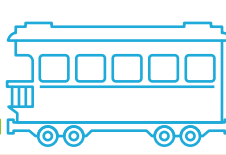
Concepts



Relations



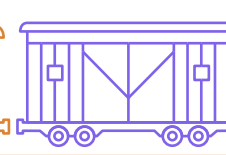
Diagrams



Toolboxes



External  
Resources



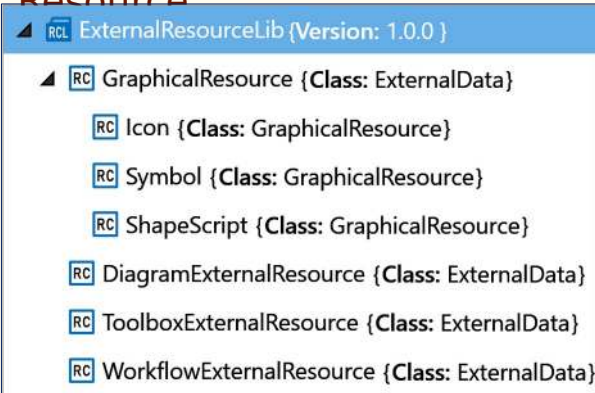
Workflow



Constraints  
and Validation

W

## ExternalResourceLib → RoleClassLib Semantic Identity of an External Resource



Generic External  
resource Role



Identifies a meaning  
/type of a External  
Resource



ExternalResourceLib

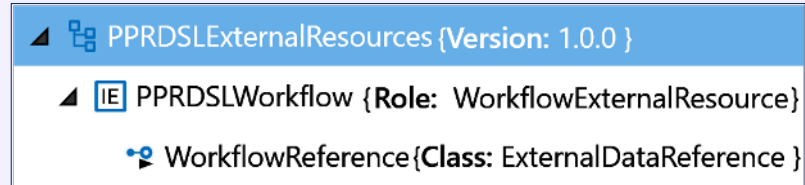


WHAT  
external resource  
is (+ type)

Role Classes from  
ExternalResourceLib

defines the semantics for

## PPRDSLExternalResources → InstanceHierarchy DSL Workflow definition



IH: DSL-wide  
external  
information



IE: represents  
workflow with the  
DSL



EI: references the  
external workflow  
data file

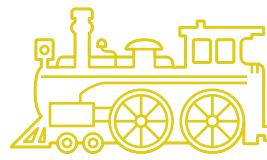


PPRDSLExterna  
lResources

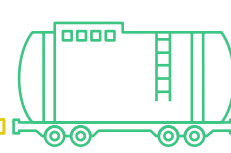


HOW external  
resources are  
structurally embedded

InternalElement from  
PPRDSLExternalResources



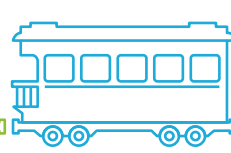
Concepts



Relations



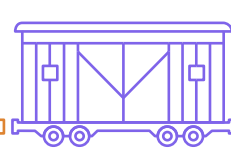
Diagrams



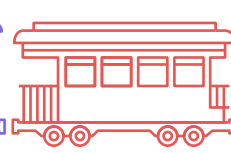
Toolboxes



External  
Resources



Workflow



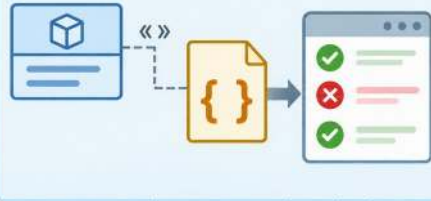
# Constraints and Validation

## How EA/Cameo represent constraint and validation?



Enterprise Architect  
(MDG)

Enterprise Architect (MDG)



Constraints with mechanisms such as OCL Conformance/Meta-Constraints

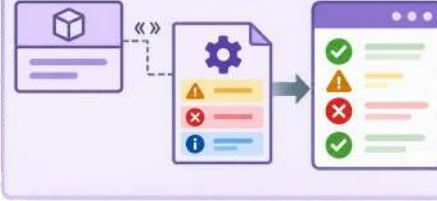


Tool-specific validation mechanisms can execute or evaluate rules



Cameo Systems  
Modeler

Cameo Systems Modeler



Constraints / validation rules with specification, severity and error information



Validation mechanisms evaluate model conditions and report violations

## What the neutral DSL specification needs?



Constraint context



Constraint condition



Machine readable semantics



Error Severity

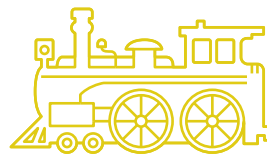


Error Message to be displayed

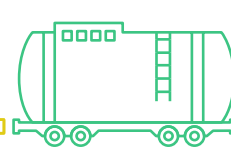
What must be true?



What should be reported if it is violated?



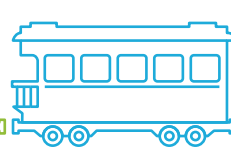
Concepts



Relations



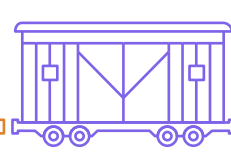
Diagrams



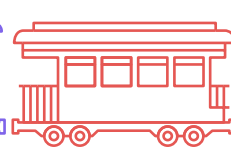
Toolboxes



External  
Resources

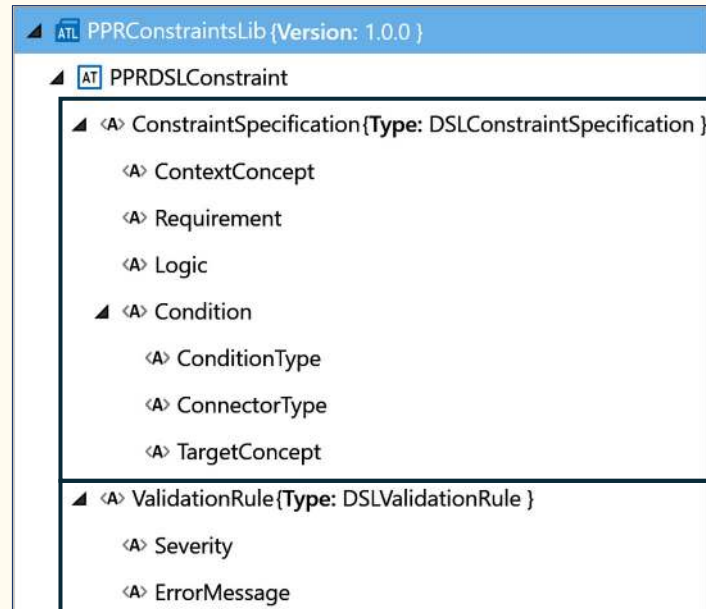


Workflow



# Constraints and Validation

PPRDSLConstraint →  
AttributeType  
Reusable constraint definition



Machine-  
readable  
semantic  
s

Validation  
informatio  
n



Applicable to the  
DSL Concepts



Human-readable  
rule + machine-  
readable semantics



Context,  
requirement and  
logical condition

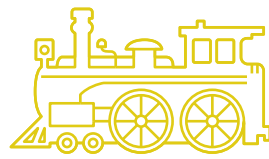


Error severity level  
and Message

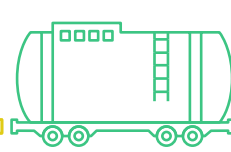
Attributes from PPRDSLConstraint  
AT

defines the semantics and  
structure for

Attributes instantiated in PPRDSLCoreLib  
SUCs



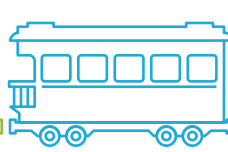
Concepts



Relations



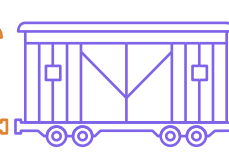
Diagrams



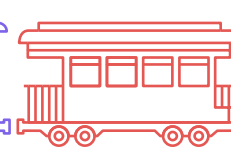
Toolboxes



External  
Resources

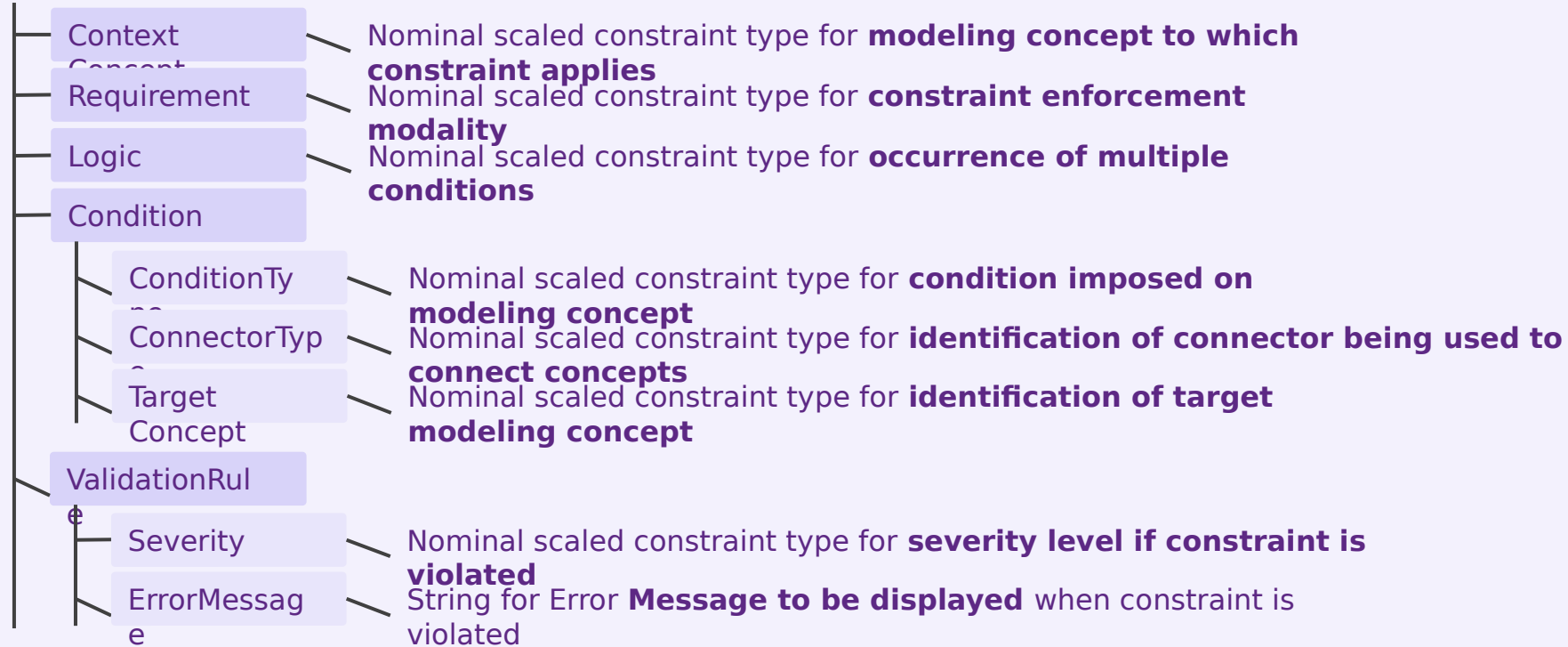


Workflow

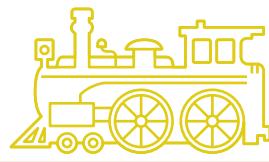


# Constraints and Validation

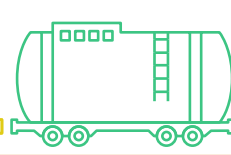
## PPRDSLConstraint



AML DSL representation only contains machine-readable constraint metadata, not an executable validation engine.



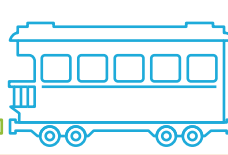
Concepts



Relations



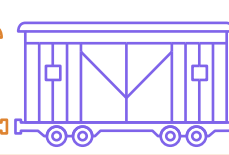
Diagrams



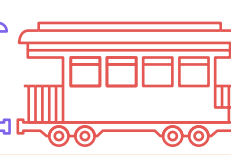
Toolboxes



External  
Resources



Workflow



# Constraints and Validation

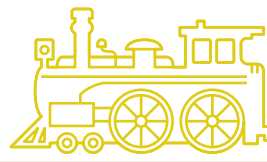
**Constraint: Each Process shall produce Output Product as its output entities**

| Name                           | Value                                  | D | Data  |
|--------------------------------|----------------------------------------|---|-------|
| ProcessOutputProductConstraint |                                        |   | Null  |
| ConstraintSpecification        |                                        |   | Null  |
| ContextConcept                 | Process                                |   | xs:st |
| Requirement                    | Required                               |   | xs:st |
| Logic                          | AllOf                                  |   | xs:st |
| Condition                      |                                        |   | xs:st |
| ConditionType                  | Participation                          |   | xs:st |
| ConnectorType                  | OutputProductProcessConne              |   | xs:st |
| TargetConcept                  | OutputProduct                          |   | xs:st |
| ValidationRule                 |                                        |   | Null  |
| Severity                       | error                                  |   | xs:st |
| ErrorMessage                   | Process must produce Output Product(s) |   | xs:st |

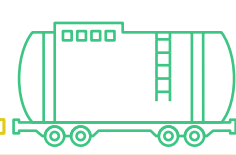
**Constraint: A Resource can be decomposed into Functional Group**

|                               |                                                           |  |       |
|-------------------------------|-----------------------------------------------------------|--|-------|
| ResourceFunctionalGroupDecomp |                                                           |  | Null  |
| ConstraintSpecification       |                                                           |  | Null  |
| ContextConcept                | Resource                                                  |  | xs:st |
| Requirement                   | Permitted                                                 |  | xs:st |
| Logic                         |                                                           |  | xs:st |
| Condition                     |                                                           |  | xs:st |
| ConditionType                 | Decomposition                                             |  | xs:st |
| ConnectorType                 | CompositionConnector                                      |  | xs:st |
| TargetConcept                 | FunctionalGroup                                           |  | xs:st |
| ValidationRule                |                                                           |  | Null  |
| Severity                      | info                                                      |  | xs:st |
| ErrorMessage                  | Resource may be decomposed into Functional Group elements |  | xs:st |

AML DSL representation only contains machine-readable constraint metadata, not an executable validation engine.



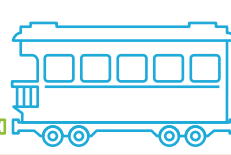
Concepts



Relations



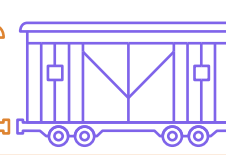
Diagrams



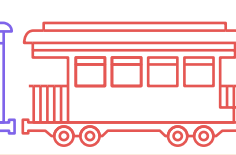
Toolboxes



External  
Resources



Workflow

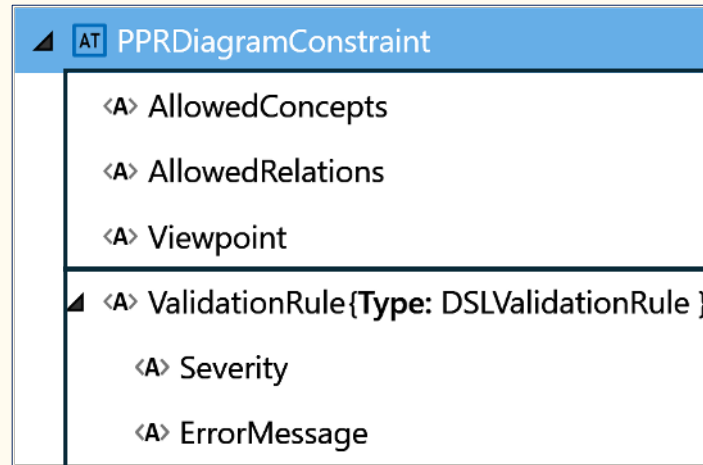


# Constraints and Validation

PPRDiagramConstraint →

AttributeType

Reusable diagram constraint definition



Machine-readable  
semantic  
s  
Validation  
informatio  
n



Applicable to the  
DSL Diagrams



Human-readable  
rule + machine-  
readable semantics

Example: Resource Hierarchy Decomposition  
Diagram Toolbox Constraint

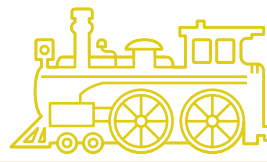
**Constraint: Only Resource, Functional Group, Component and Composition relations are permitted in Resource Hierarchy diagram**

| Name                                           | Value                                                                                                            | Defi | U | DataTyp   |
|------------------------------------------------|------------------------------------------------------------------------------------------------------------------|------|---|-----------|
| ResourceHierarchyDecompositionDiagramConstrain |                                                                                                                  |      |   | Null      |
| AllowedConcepts                                | Resource, FunctionalGroup, Component                                                                             |      |   | xs:string |
| AllowedRelations                               | Parent/CompositionRelations                                                                                      |      |   | xs:string |
| Viewpoint                                      | ResourceCentric                                                                                                  |      |   | xs:string |
| ValidationRule                                 |                                                                                                                  |      |   | Null      |
| Severity                                       | error                                                                                                            |      |   | xs:string |
| ErrorMessage                                   | Only Resource, Functional Group, Component and Composition relations are permitted in Resource Hierarchy diagram |      |   | xs:string |

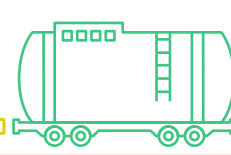
Attributes from  
PPRDiagramConstraint AT

defines the semantics and  
structure for

Attributes instantiated in PPRDSLDiagramsLib  
SUCs



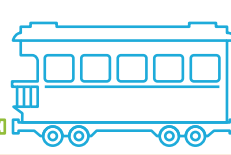
Concepts



Relations



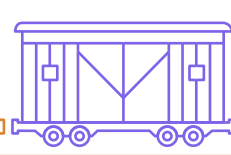
Diagrams



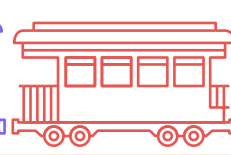
Toolboxes



External  
Resources



Workflow

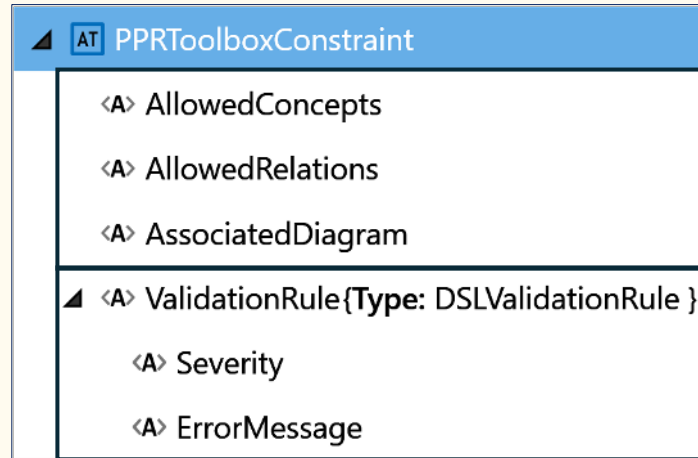


# Constraints and Validation

PPRToolboxConstraint →

AttributeType

Reusable diagram constraint definition



Machine-readable  
semantic  
s  
Validation  
informatio  
n



Applicable to the  
DSL Toolboxes



Human-readable  
rule + machine-  
readable semantics

Example: Process Resource Toolbox  
Constraint

**Constraint: Only Process and Resource concepts and Resource Process Connector is permitted in Process Resource Toolbox**

| Name                             | Value                                                                                                    | Defa | Ur | DataType  |
|----------------------------------|----------------------------------------------------------------------------------------------------------|------|----|-----------|
| ProcessResourceToolboxConstraint |                                                                                                          |      |    | Null      |
| AllowedConcepts                  | Process, Resource                                                                                        |      |    | xs:string |
| AllowedRelations                 | ProcessResourceRelations                                                                                 |      |    | xs:string |
| AssociatedDiagram                | ProcessResourceDiagram                                                                                   |      |    | xs:string |
| ValidationRule                   |                                                                                                          |      |    | Null      |
| Severity                         | error                                                                                                    |      |    | xs:string |
| ErrorMessage                     | Only Process and Resource concepts and ResourceProcessConnector is permitted in Process Resource Toolbox |      |    | xs:string |

Attributes from PPRToolboxConstraint  
AT

defines the semantics and  
structure for

Attributes instantiated in PPRDSLToolboxesLib  
SUCs

# AMLEditor Multiselect



## Problem

Some Attribute or constraints require multiple selectable values

- Single value is not sufficient
- Need to select several values at the same time



## Example

### Allowed Concepts

- ☐ Product
- ☒ Process
- ☒ Resource



## Required editor capability



### Native multi-selection

Select and deselect multiple values easily



### Enumeration-based multi-value selection

Use of defined value lists



### List-valued attribute editing

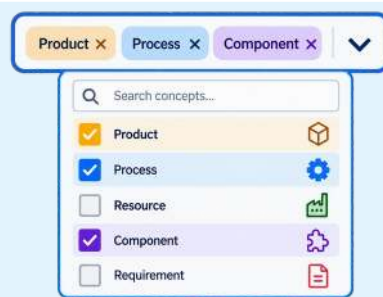
Support attributes with multiple entries



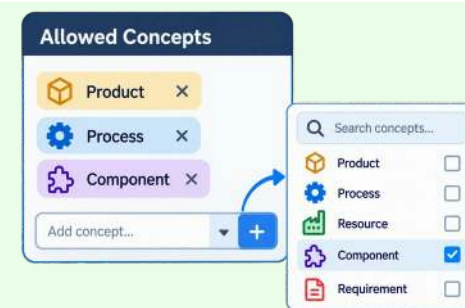
## Possible Approaches



Native multi-select



Multi-value enumeration



List-valued editor



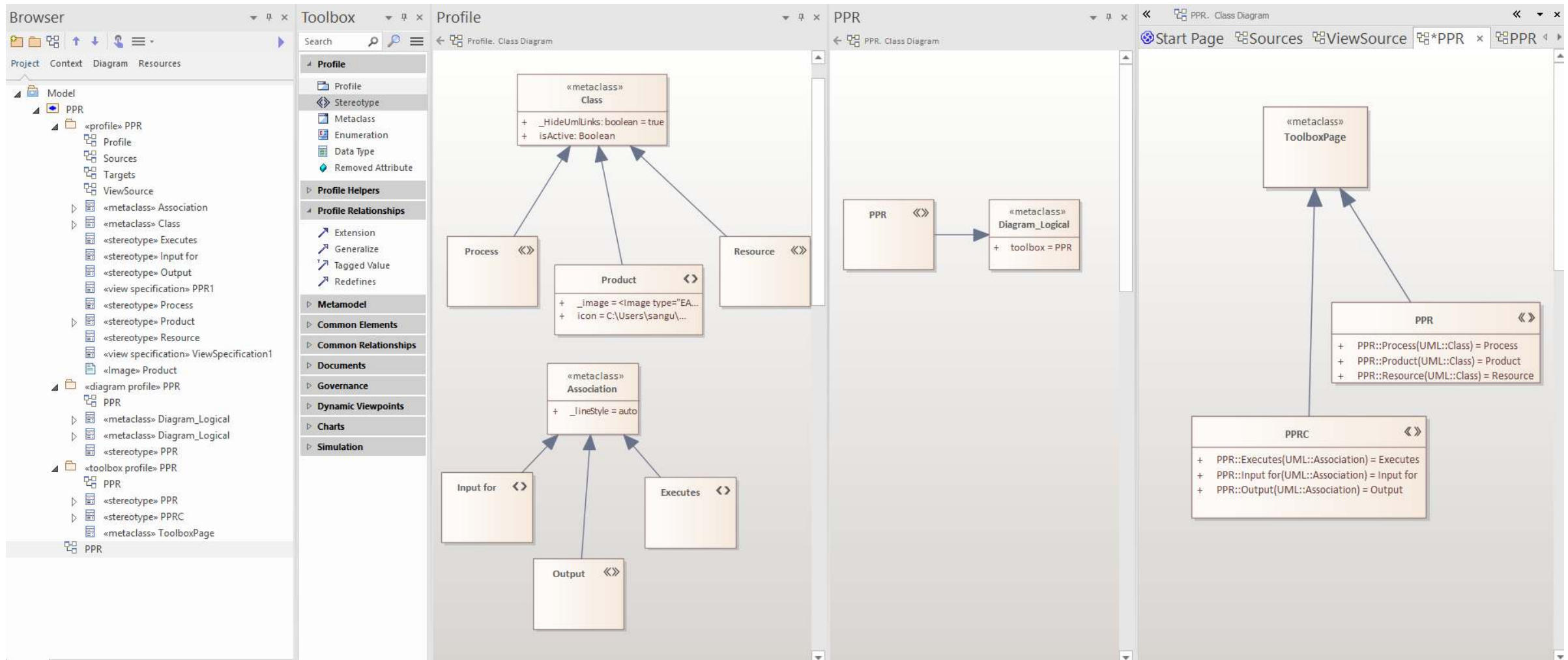
The requirement is an editor usability feature, not something only specific to the DSL representation

# Tool-Independent DSL Specification in AutomationML

The screenshot displays five panels of the AutomationML IDE, each showing a different aspect of the DSL specification:

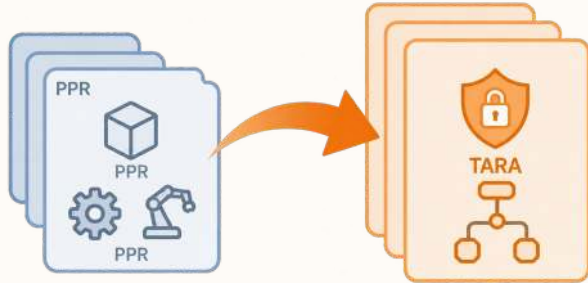
- InstanceHierarchy:** Shows a tree structure of instances. Annotations include `PPRMetamodel {Version: 1.0.0}`, `Product {Class: Product Role: Product}`, `Process {Class: Process Role: Process}`, `Resource {Class: Resource Role: Resource}`, `FunctionalGroup {Class: FunctionalGroup Role: FunctionalGroup}`, `Component {Class: Component Role: Component}`, `PPRDSLExternalResources {Version: 1.0.0}`, and `PPRDSLWorkflow {Role: WorkflowExternalResource}`.  
**Metamodel and External resource**
- SystemUnitClassLib:** Shows a tree structure of system unit classes. Annotations include `PPRDSLCoreLib {Version: 1.0.0}`, `Product {Role: Product}`, `Process {Role: Process}`, `Resource {Role: Resource}`, `FunctionalGroup {Role: FunctionalGroup}`, `Component {Role: Component}`, `PPRDSLDiagramsLib {Version: 1.0.0}`, `PPRDiagram {Role: Diagram}`, `ProductStructureDiagram {Role: Diagram}`, `ResourceHierarchyDiagram {Role: Diagram}`, `ProductProcessDiagram {Role: Diagram}`, `ProcessResourceDiagram {Role: Diagram}`, `PPRDSLToolboxesLib {Version: 1.0.0}`, `PPRToolbox {Role: Toolbox}`, `ProductStructureToolbox {Role: Toolbox}`, `ResourceHierarchyToolbox {Role: Toolbox}`, `ProductProcessToolbox {Role: Toolbox}`, and `ProcessResourceToolbox {Role: Toolbox}`.  
**Reusable DSL Structures Integrating Semantics, Relations & Constraints**
- RoleClassLib:** Shows a tree structure of role classes. Annotations include `PPRConceptsLib {Version: 1.0.0}`, `Product {Class: Product}`, `Process {Class: Process}`, `Resource {Class: Resource}`, `FunctionalGroup {Class: Resource}`, `Component {Class: FunctionalGroup}`, `DiagramLib {Version: 1.0.0}`, `Diagram {Class: AutomationMLBaseRole}`, `ToolboxLib {Version: 1.0.0}`, `Toolbox {Class: AutomationMLBaseRole}`, `ExternalResourceLib {Version: 1.0.0}`, and `GraphicalResource {Class: ExternalData}`.  
**Semantic Concepts & Classification**
- InterfaceClassLib:** Shows a tree structure of interface classes. Annotations include `AutomationMLInterfaceClassLib {Version: 2.11.0}`, `PPRInterfaceLib {Version: 1.0.0}`, `ProductProcessConnector {Class: AutomationMLBaseInterface}`, `ResourceProcessConnector {Class: AutomationMLBaseInterface}`, and `CompositionConnector {Class: AutomationMLBaseInterface}`.  
**Relation & Connector Types**
- AttributeTypeLib:** Shows a tree structure of attribute types. Annotations include `AutomationMLBaseAttributeTypeLib {Version: 2.11.0}`, `PPRConstraintsLib {Version: 1.0.0}`, `PPRDSLConstraint`, `PPRDiagramConstraint`, `PPRToolboxConstraint`, `DSLConstraintSpecification`, and `DSLValidationRule`.  
**Constraints & Validation Rules**

# DSL specification in Enterprise Architect



# Outlook

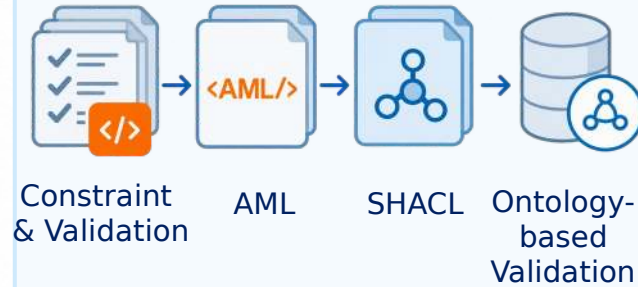
- 1 Broader DSL Applications  
TARA / Cybersecurity  
DSLs  
AML DSL Specs → TARA DSL



- 2 Formal Validation

FD4AML/SHACL

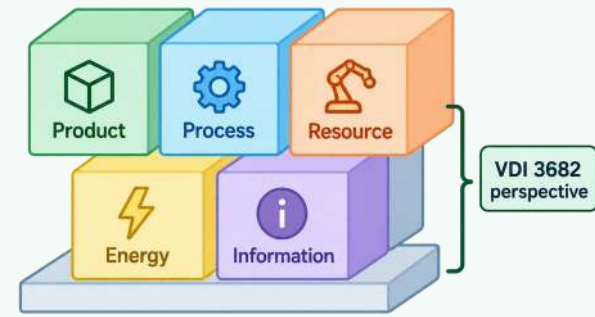
Constraint & Validation →  
AutomationML → SHACL →  
Ontology-based Validation



- 3 Extension towards VDI 3682

PPR → VDI 3682  
Perspective

Product + Process + Resource  
→ Energy + Information



Explore  
new domains

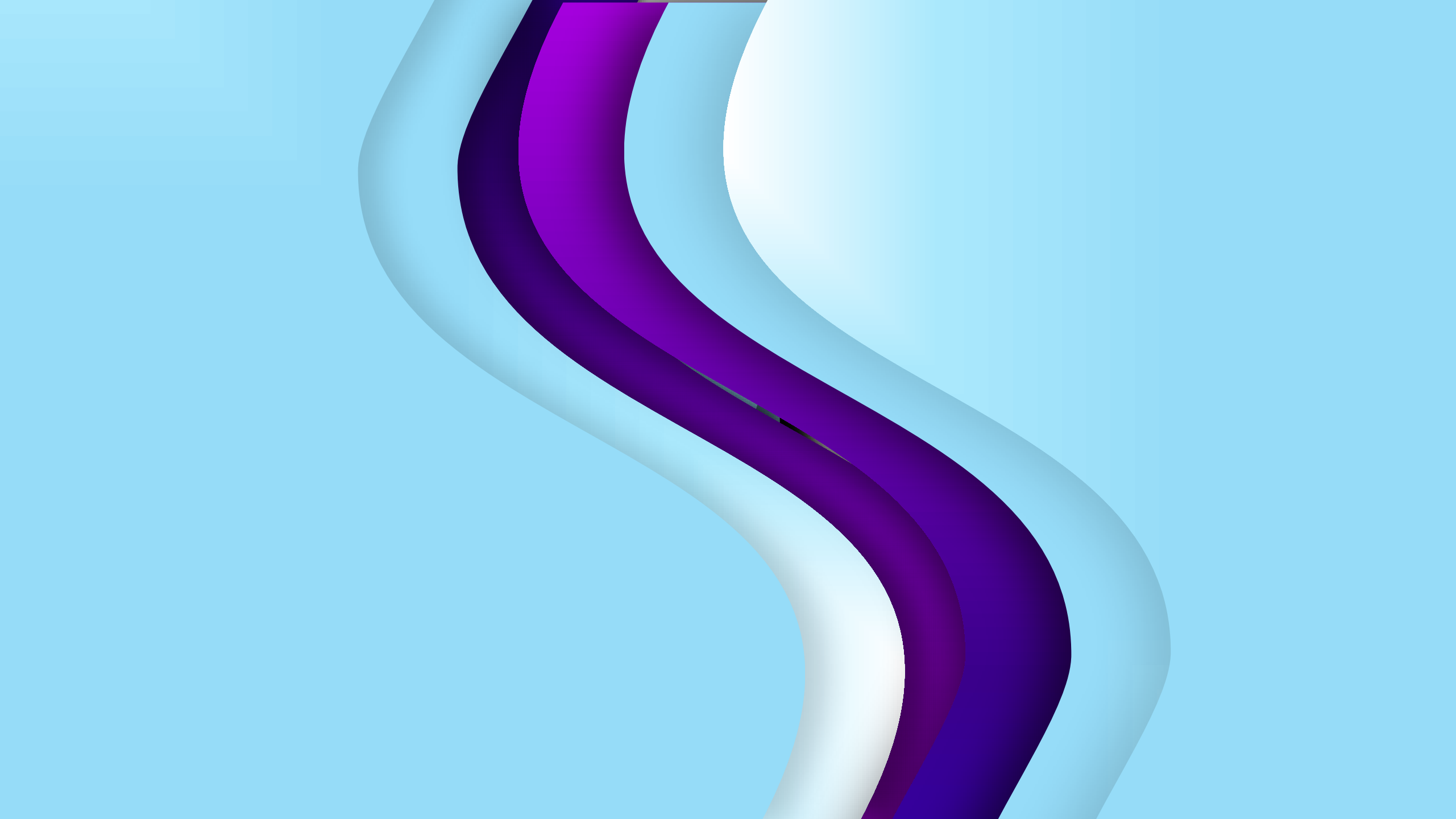
Enable  
formal assurance

Expand  
modeling scope



Towards reusable, formally constrained and tool-independent DSL specification in AutomationML

CONTRIBUTIONS CONNECT  
POSSIBILITIES





**THANK YOU**