

Bringing Graphical Engineering Languages to AutomationML

Native process modeling in the AutomationML Editor

Hamied Nabizada

Funded by

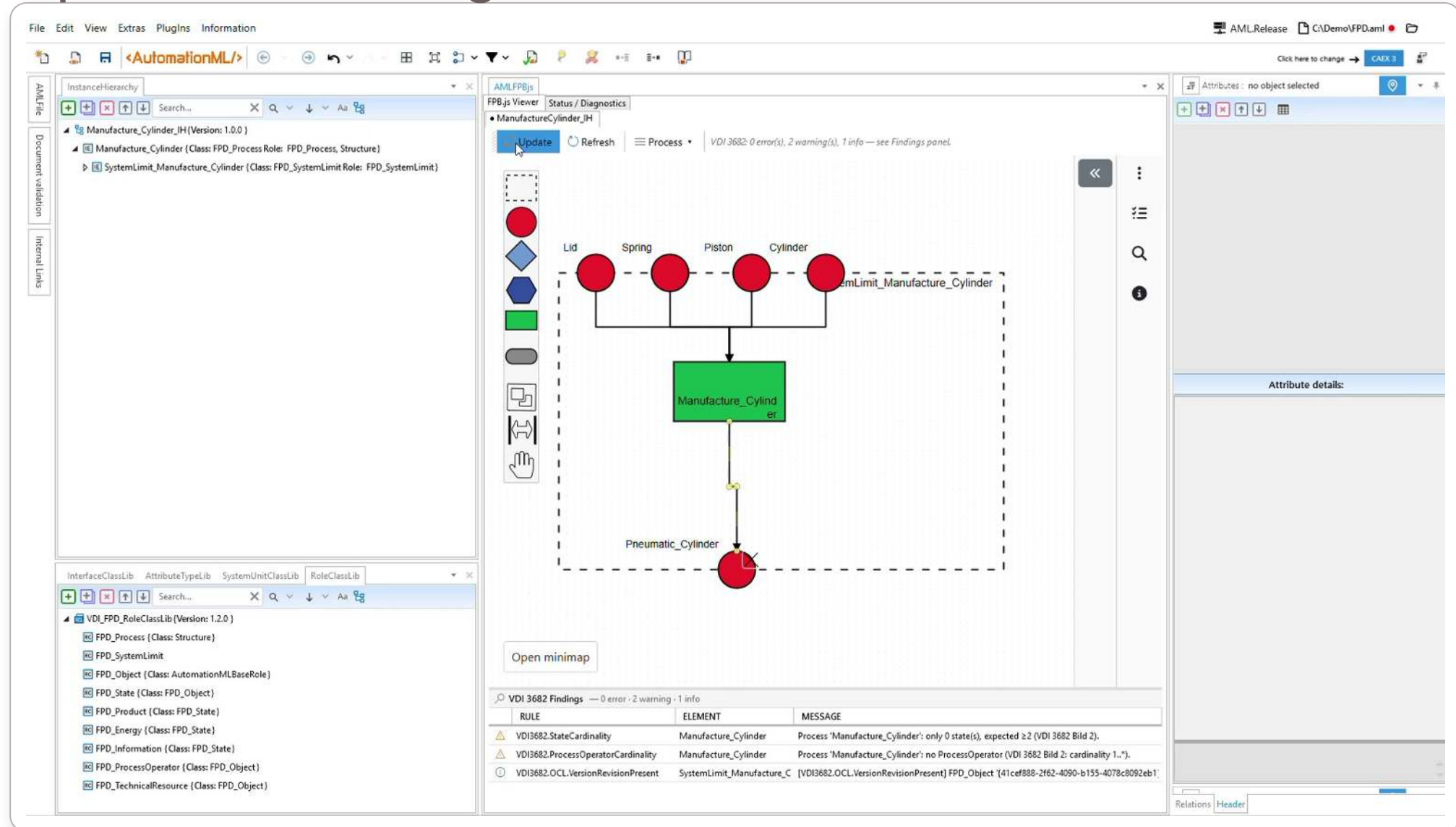


Digitalization and Technology
Research Center of the Bundeswehr



Funded by
the European Union
NextGenerationEU

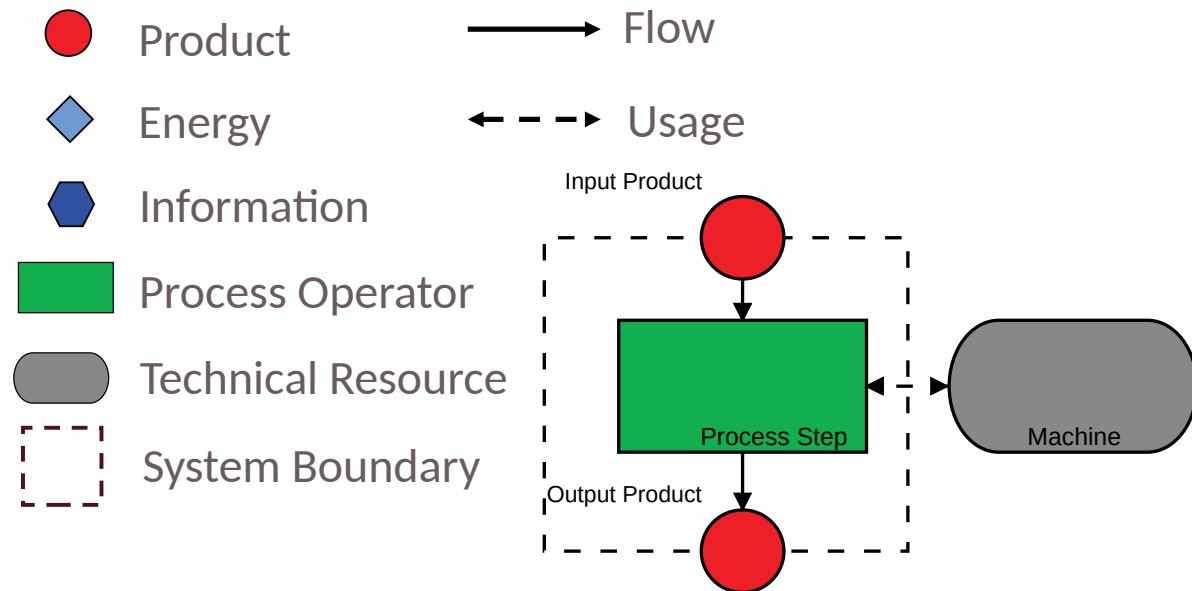
Native process modeling in AutomationML



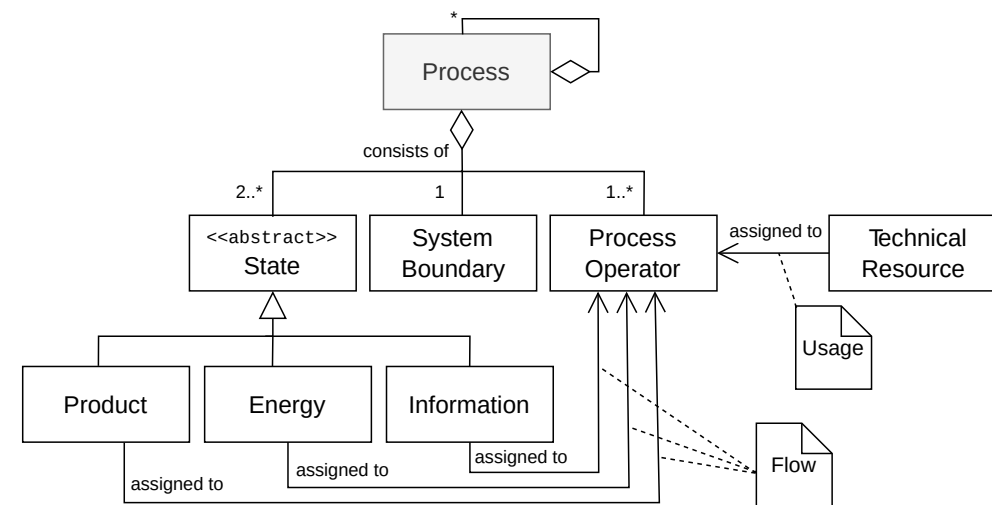
Process

What did we see? VDI/VDE 3682

Part 1: Graphical representation

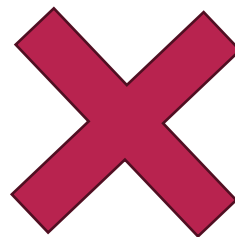


Part 2: Information model



Part 3: Exchange format

Announced in 2015,



but still not defined.

Process

The VDI wanted a Part 3: So we tried it in AutomationML

The Result

```
RC VDI_FPD_RoleClassLib Version: 1.2.0 {
  RC FPD_Process {Class: Structure}
  RC FPD_SystemLimit
  RC FPD_Object {Class: AutomationMLBaseRole}
  RC FPD_State {Class: FPD_Object}
  RC FPD_Product {Class: FPD_State}
  RC FPD_Energy {Class: FPD_State}
  RC FPD_Information {Class: FPD_State}
  RC FPD_ProcessOperator {Class: FPD_Object}
  RC FPD_TechnicalResource {Class: FPD_Object}
}
RC ISO_PT_RoleClassLib Version: 1.0.0 {
```

```
SUC VDI_FPD_SystemUnitClassLib Version: 1.2.0 {
  SUC FPD_Process {Role: FPD_Process, Structure}
  SUC FPD_SystemLimit {Role: FPD_SystemLimit}
  SUC FPD_Object {Role: FPD_Object}
  SUC FPD_State {Class: FPD_Object Role: FPD_State}
  SUC FPD_Product {Class: FPD_State Role: FPD_Product, Product}
  SUC FPD_Energy {Class: FPD_State Role: FPD_Energy}
  SUC FPD_Information {Class: FPD_State Role: FPD_Information}
  SUC FPD_ProcessOperator {Class: FPD_Object Role: FPD_ProcessOperator, Process}
  SUC FPD_TechnicalResource {Class: FPD_Object Role: FPD_TechnicalResource, Resource}
}
SUC ISO_PT_SystemUnitClassLib Version: 1.0.0 {
```

```
ICL VDI_FPD_InterfaceClassLib Version: 1.2.0 {
  ICL FPD_Port {Class: Port}
  ICL FPD_FlowIn {Class: FPD_Port}
  ICL FPD_FlowOut {Class: FPD_Port}
  ICL FPD_Usage {Class: FPD_Port}
  ICL FPD_ParallelFlowIn {Class: FPD_Port}
  ICL FPD_ParallelFlowOut {Class: FPD_Port}
  ICL FPD_AlternativeFlowIn {Class: FPD_Port}
  ICL FPD_AlternativeFlowOut {Class: FPD_Port}
}
ICL ISO_PT_InterfaceClassLib Version: 1.0.0 {
```

```
ATL VDI_FPD_AttributeTypeLib Version: 1.2.0 {
  ATL FPD_Identification
  <A> uniqueIdent
  <A> longName
  <A> shortName
  <A> versionNumber
  <A> revisionNumber
  ATL FPD_Characteristic
  > <A> Category {Type: FPD_Identification}
  > <A> DescriptiveElement
  > <A> RelationalElement
}
ATL ISO_PT_AttributeTypeLib Version: 1.0.0 {
ATL OMG_DD_AttributeTypeLib Version: 0.1.0 {
ATL AutomationML_ObjectReferences_AttributeT
```

```
ATL VDI_FPD_AttributeTypeLib Version: 1.2.0 {
ATL ISO_PT_AttributeTypeLib Version: 1.0.0 {
ATL OMG_DD_AttributeTypeLib Version: 0.1.0 {
  ATL DD_Bounds
  > <A> position {Type: DD_Point}
  <A> width
  <A> height
  ATL DD_Waypoint
  > <A> position {Type: DD_Point}
  ATL DD_Point
  <A> x
  <A> y
}
ATL AutomationML_ObjectReferences_AttributeT
```

5 libraries: 28 classes for the language, 3 for the geometry.

Plain CAEX. All of it public.

And a mapper between FPB.js and AML:
<https://aml.fpbjs.net>

Process

FPB.js and AML read each other



JSON → AML

AML → JSON

Input (FPB.js JSON) Clear

Output (AutomationML) Copy Download

```
"identification": {
  "$type": "fpb:Identification",
  "uniqueIdent": "caaf949f-23e2-41b4-8744-1fedc7169042",
  "longName": "",
  "shortName": "Check_Quality",
  "versionNumber": "",
  "revisionNumber": ""
},
"isAssignedTo": [
  "8c9934d3-3b22-4a5e-b577-db50de1d8d66"
],
"incoming": [
  "f5a9ec58-a3e7-4cda-b19b-cf64c6e6844c"
],
"outgoing": [
  "48745616-d9ff-4198-9281-4c7898a8ce7e",
  "d0191091-8828-48fd-bcad-9c315665aaf8",
  "cf450dc5-d77f-4fc2-9824-1cb0682a1904"
],
"name": "Check_Quality"
},
```

```
RefBaseRoleClassPath="AutomationMLBaseLibrariesAMLEd22_11_0@Automation
MLBaseRoleClassLib/AutomationMLBaseRole/Product" />
</InternalElement>
<InternalElement Name="Check_Quality" ID="{caaf949f-23e2-41b4-
8744-1fedc7169042}"
RefBaseSystemUnitPath="VDI_FPD_SystemUnitClassLib/FPD_ProcessOperator"
>
  <Attribute Name="Identification" AttributeDataType="xs:string"
RefAttributeType="VDI_FPD_AttributeTypeLib/FPD_Identification">
    <Attribute Name="uniqueIdent" AttributeDataType="xs:string">
      <Value>caaf949f-23e2-41b4-8744-1fedc7169042</Value>
    </Attribute>
    <Attribute Name="longName" AttributeDataType="xs:string" />
    <Attribute Name="shortName" AttributeDataType="xs:string">
      <Value>Check_Quality</Value>
    </Attribute>
    <Attribute Name="versionNumber"
AttributeDataType="xs:string" />
    <Attribute Name="revisionNumber"
AttributeDataType="xs:string" />
  </Attributes>
```

2

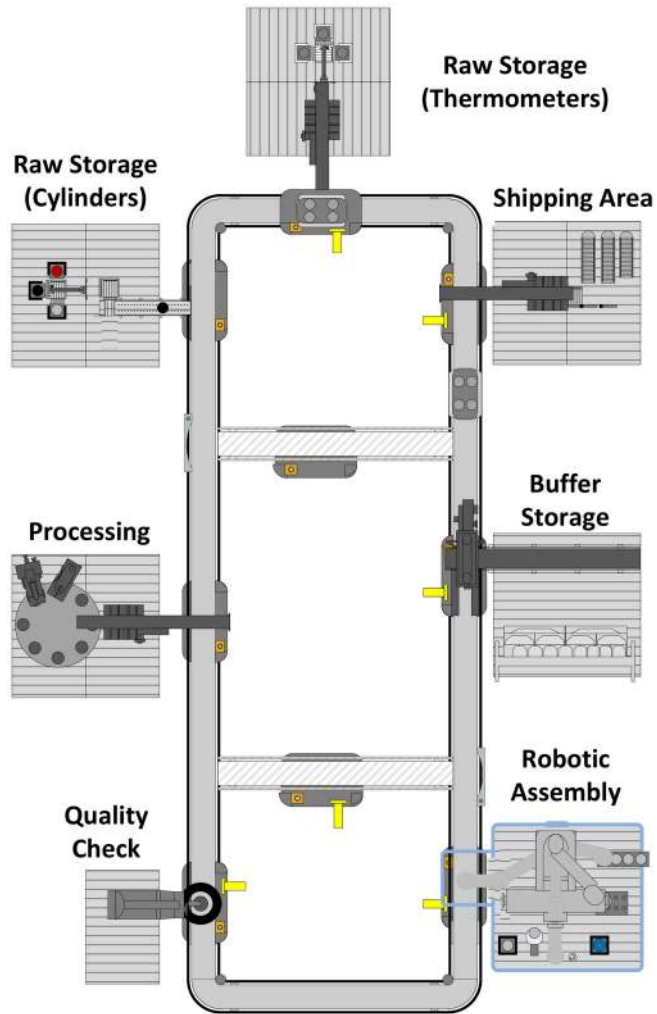
Convert

```
<InternalElement Name="Check_Quality" ID="{caaf949f-23e2-41b4-
8744-1fedc7169042}"
RefBaseSystemUnitPath="VDI_FPD_SystemUnitClassLib/FPD_ProcessOperator"
```

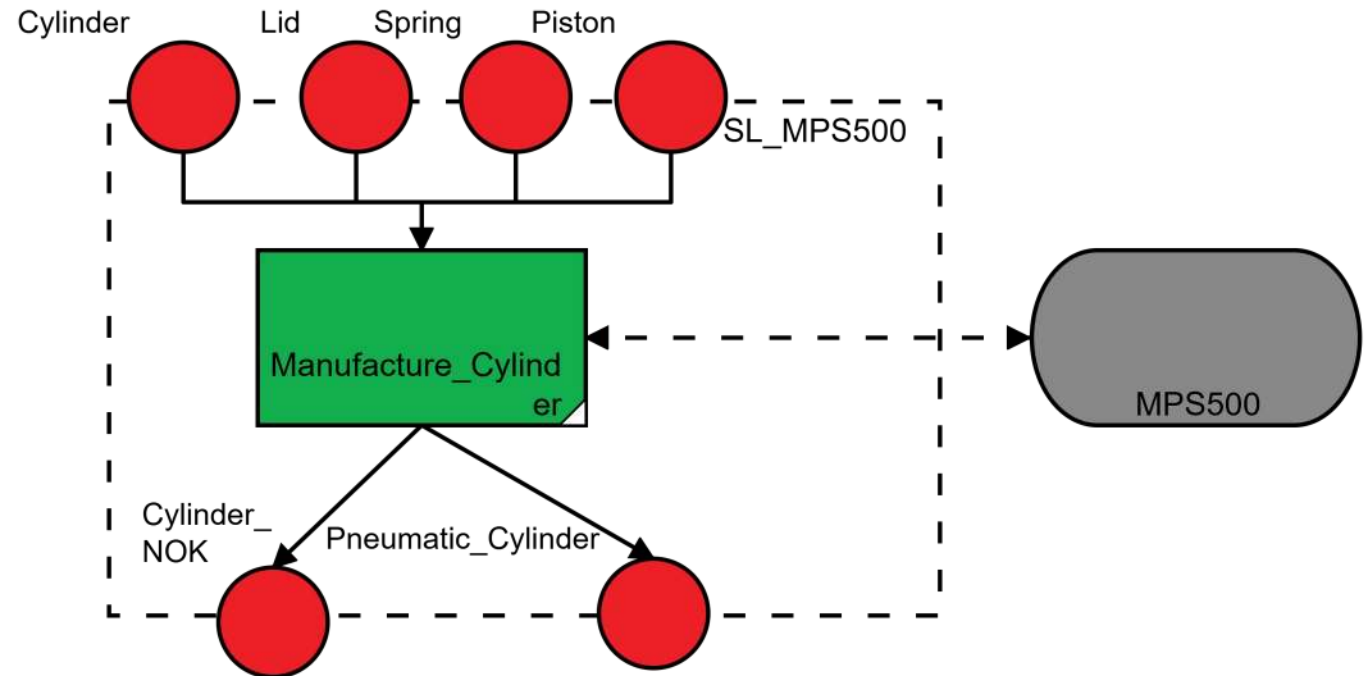
<https://aml.fpbjs.net>

Process

The MPS 500 and its process



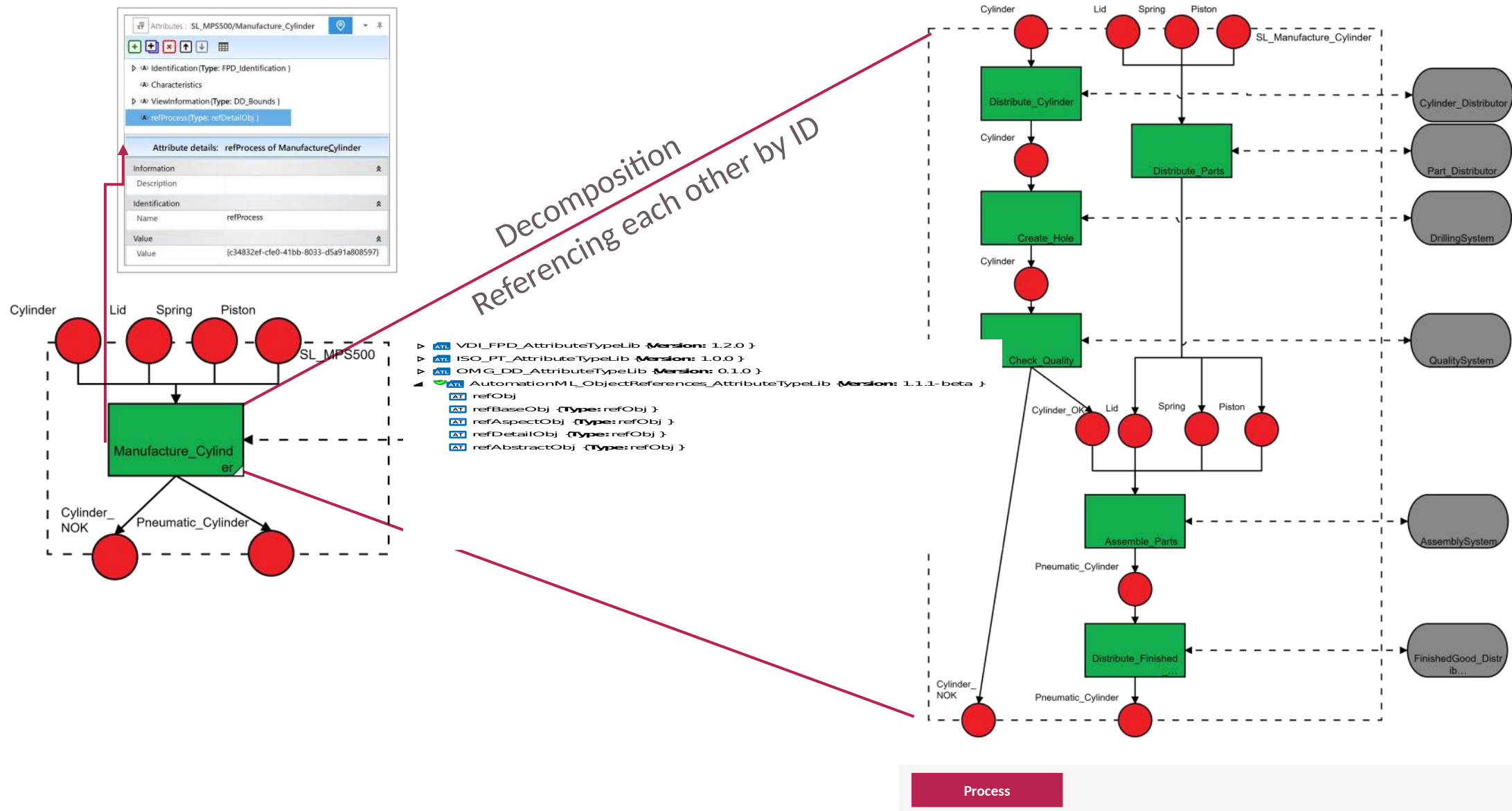
Festo MPS 500, seven stations. It makes a pneumatic cylinder from four parts.



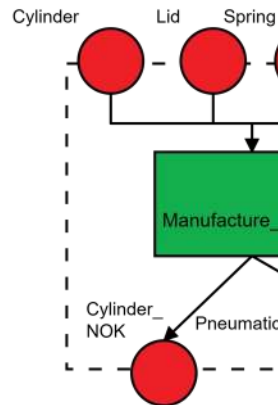
The same plant as a process: VDI/VDE 3682, one operator.

Process

One operator becomes a process



The plant is already in the file



- ▷ IH_FPD_ProcessView {Version: 1.0.0 }
- ▷ IH_PT_BehaviorView {Version: 1.0.0 }
- ▲ IH_MPS500_Plant {Version: 1.0.0 }
 - ▲ IE MPS500
 - ▷ RawStorageThermometers
 - ▷ RawStorageCylinders
 - ▷ ProcessingStation
 - ▷ QualityCheckStation
 - ▷ RoboticAssemblyStation
 - ▷ BufferStorage
 - ▷ ShippingArea
 - ▷ Controller

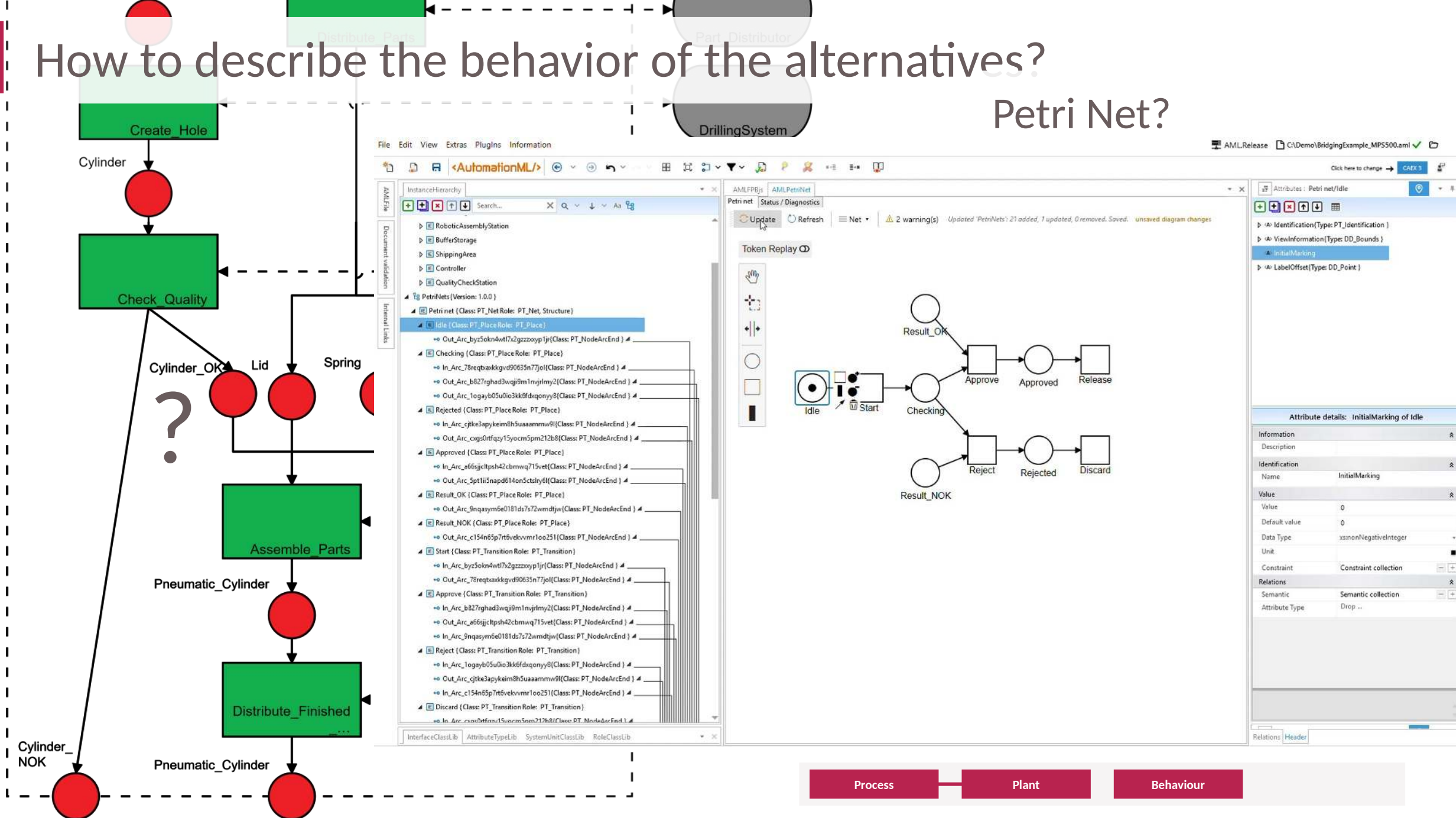
- ▷ IE Distribute_Cylinder {Class: FPD_ProcessOperator Role: FPD_ProcessOperator, Process }
- ▷ IE Spring {Class: FPD_Product Role: FPD_Product, Product }
- ▷ IE Lid {Class: FPD_Product Role: FPD_Product, Product }
- ▷ IE Cylinder {Class: FPD_Product Role: FPD_Product, Product }
- ▲ IE QualitySystem {Class: FPD_TechnicalResource Role: FPD_TechnicalResource, Resource }
 - FPD_Usage {Class: FPD_Usage }
 - PlantEquipmentLink {Class: AutomationMLBaseInterface }
- ▷ IE DrillingSystem {Class: FPD_TechnicalResource Role: FPD_TechnicalResource, Resource }
- ▷ IE Cylinder_Distributor {Class: FPD_TechnicalResource Role: FPD_TechnicalResource, Resource }
- ▷ IE Part_Distributor {Class: FPD_TechnicalResource Role: FPD_TechnicalResource, Resource }
- ▷ IE AssemblySystem {Class: FPD_TechnicalResource Role: FPD_TechnicalResource, Resource }
- ▷ IE FinishedGood_Distributor {Class: FPD_TechnicalResource Role: FPD_TechnicalResource, Resource }
- ▷ IE SystemLimit_Manufacture_Cylinder {Class: FPD_SystemLimit Role: FPD_SystemLimit }
- ▷ IE SL_MPS500 {Class: FPD_Process Role: FPD_Process, Structure }
- ▷ IH_PT_BehaviorView {Version: 1.0.0 }
- ▲ IH_MPS500_Plant {Version: 1.0.0 }
 - ▲ IE MPS500
 - ▷ RawStorageThermometers
 - ▷ RawStorageCylinders
 - ▷ ProcessingStation
 - ▲ IE QualityCheckStation
 - EquipmentLink {Class: AutomationMLBaseInterface }
 - Sensor_OK {Class: AutomationMLBaseInterface }
 - Sensor_NOK {Class: AutomationMLBaseInterface }
 - ▷ RoboticAssemblyStation
 - ▷ BufferStorage

Process

Plant

How to describe the behavior of the alternatives?

Petri Net?



The same trick for Petri nets

```
▷ VDI_FPD_RoleClassLib {Version: 1.2.0 }
└─ ISO_PT_RoleClassLib {Version: 1.0.0 }
   └─ PT_Net {Class: Structure }
   └─ PT_Element {Class: AutomationMLBaseRole }
   └─ PT_Place {Class: PT_Element }
   └─ PT_Transition {Class: PT_Element }
   └─ PT_Arc {Class: PT_Element }
```

```
└─ ISO_PT_InterfaceClassLib {Version: 1.0.0 }
   └─ PT_Port {Class: Port }
   └─ PT_NodeArcEnd {Class: PT_Port }
   └─ PT_ArcSource {Class: PT_Port }
   └─ PT_ArcTarget {Class: PT_Port }
   └─ VDI_FPD_InterfaceClassLib {Version: 1.2.0 }
```

```
▷ VDI_FPD_AttributeTypeLib {Version: 1.2.0 }
└─ ISO_PT_AttributeTypeLib {Version: 1.0.0 }
   └─ PT_Identification
      └─ ID
         └─ IDTYPE
            └─ OMG_DD_AttributeTypeLib {Version: 0.1.0 }
               └─ AutomationML_ObjectReferences_AttributeTypeLib {Version: 1.1.1-beta }
```

```
▷ VDI_FPD_AttributeTypeLib {Version: 1.2.0 }
└─ ISO_PT_AttributeTypeLib {Version: 1.0.0 }
   └─ OMG_DD_AttributeTypeLib {Version: 0.1.0 }
      └─ DD_Bounds
         └─ <A> position {Type: DD_Point }
            └─ <A> width
            └─ <A> height
      └─ DD_Waypoint
         └─ <A> position {Type: DD_Point }
      └─ DD_Point
         └─ <A> x
         └─ <A> y
      └─ AutomationML_ObjectReferences_AttributeTy
```

3 libraries: 15 classes for the language. The geometry library is the same as for the FPD.

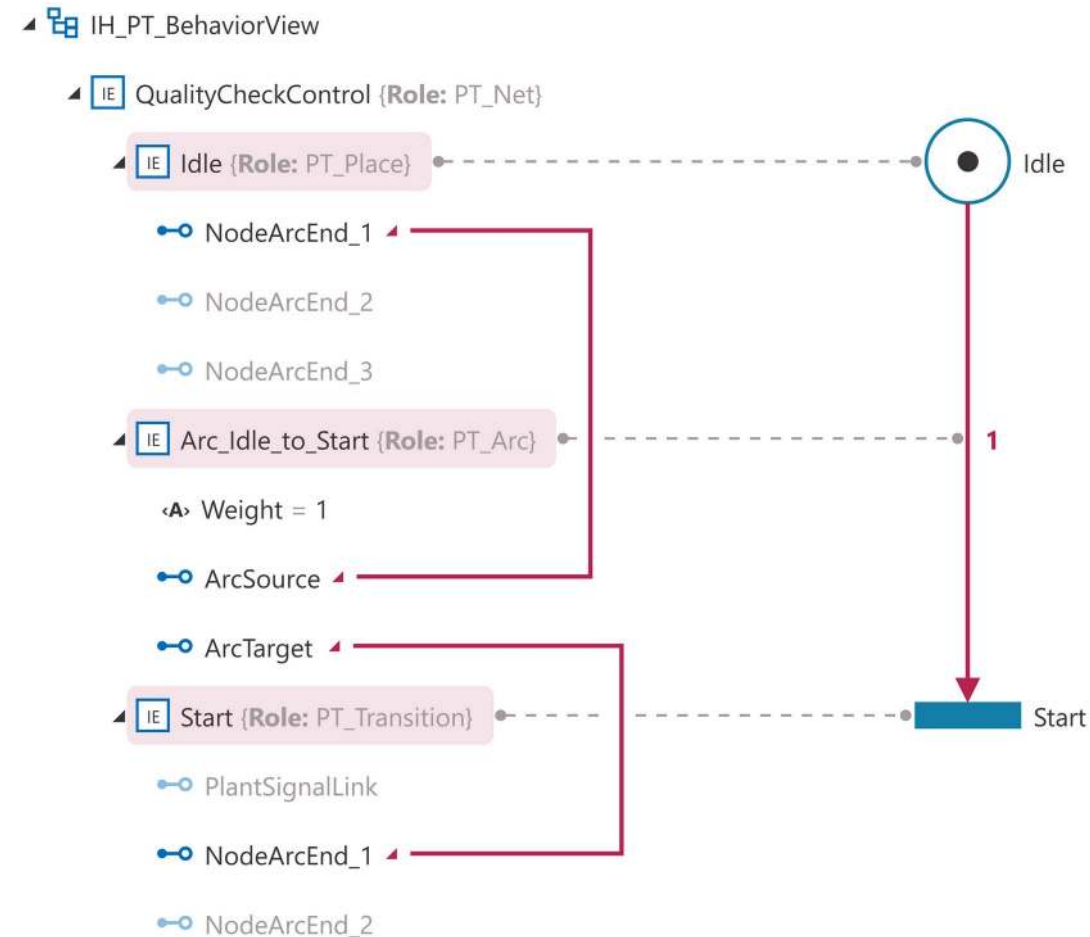
PNML stays the exchange format. AML is where the net meets the plant.

Process

Plant

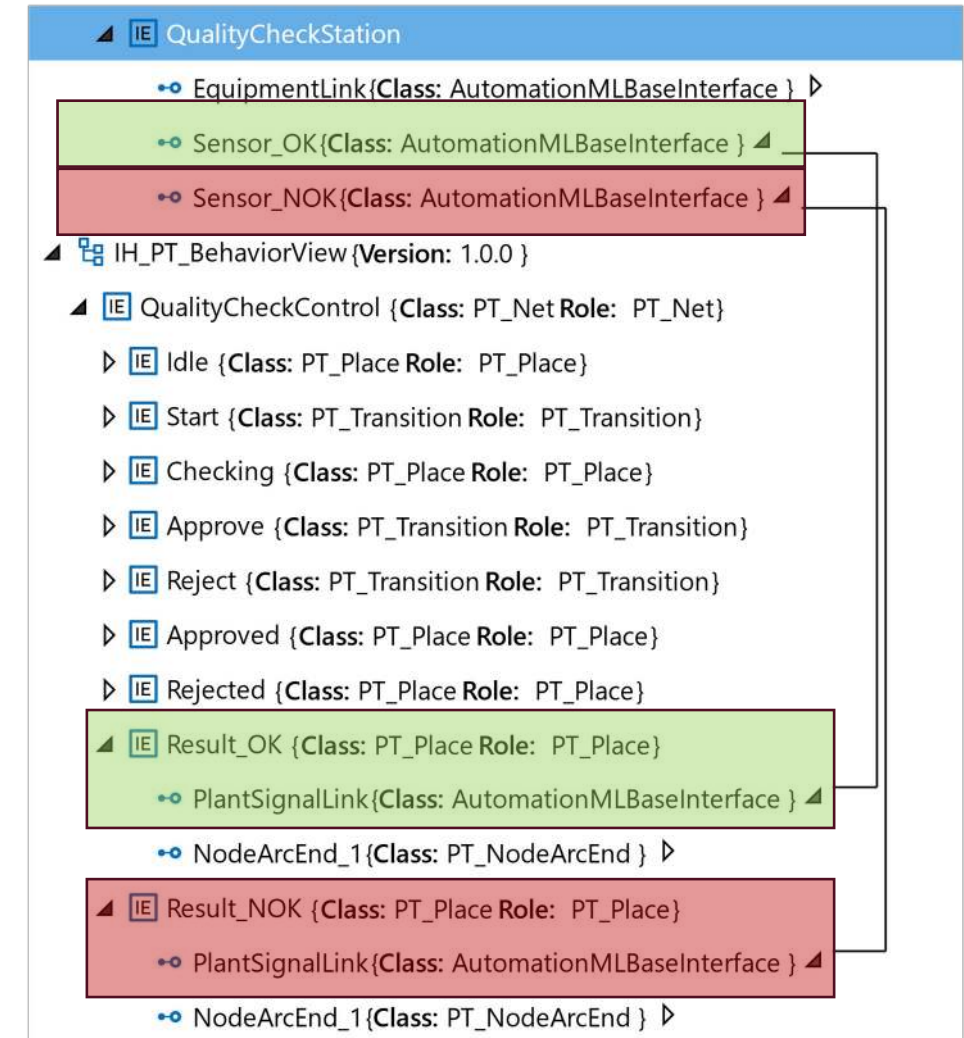
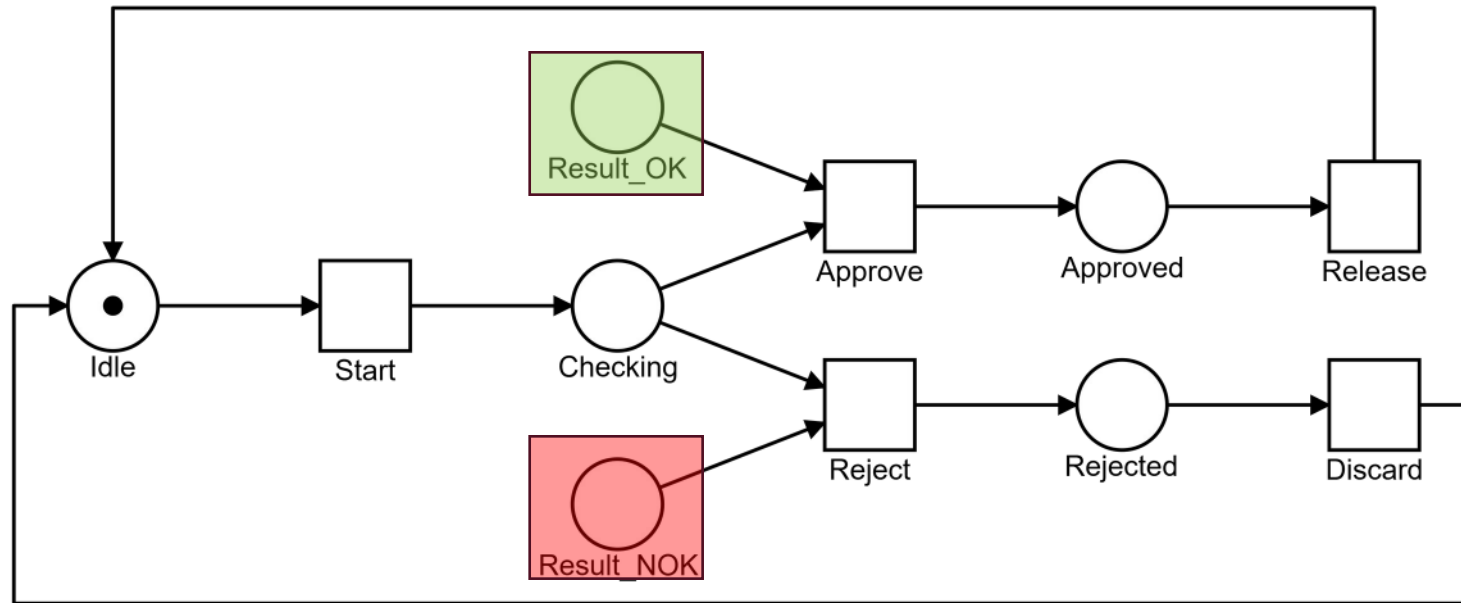
Behaviour

An arc is an element, not a line



A link cannot carry a weight. An InternalElement can, and it can be referenced and versioned.

Result_OK listens to the plant



refObj: one object, several representations



AutomationML_ObjectReferences_AttributeTypeLib

- refObj**
abstract, semantics unspecified
- refBaseObj**
on an aspect object, points at the base object
- refAspectObj**
on a base object, optional way back
- refDetailObj**
on an abstract object, points at the detail
- refAbstractObj**
on a detail object, optional way back

5 AttributeTypes, all xs:IDREF. Nothing in CAEX changes.

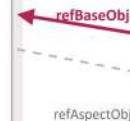
Same object, a second context

P&ID, certified

P1 {Pump}
«A» NominalFlow = 12 m³/h
«A» Medium = water

Security model

P1_sec {Pump}
«A» refBaseObj = ID of P1
«A» RiskCategory = high
«A» AccessLevel = 2



Same pump, seen by the security team: own attributes, and the certified model stays untouched.

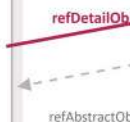
Same subject, two levels of detail

Process view, top level

Manufacture_Cylinder {FPD_ProcessOperator}
«A» refProcess = ID of the sub process
«A» Identification

Sub process, decomposed

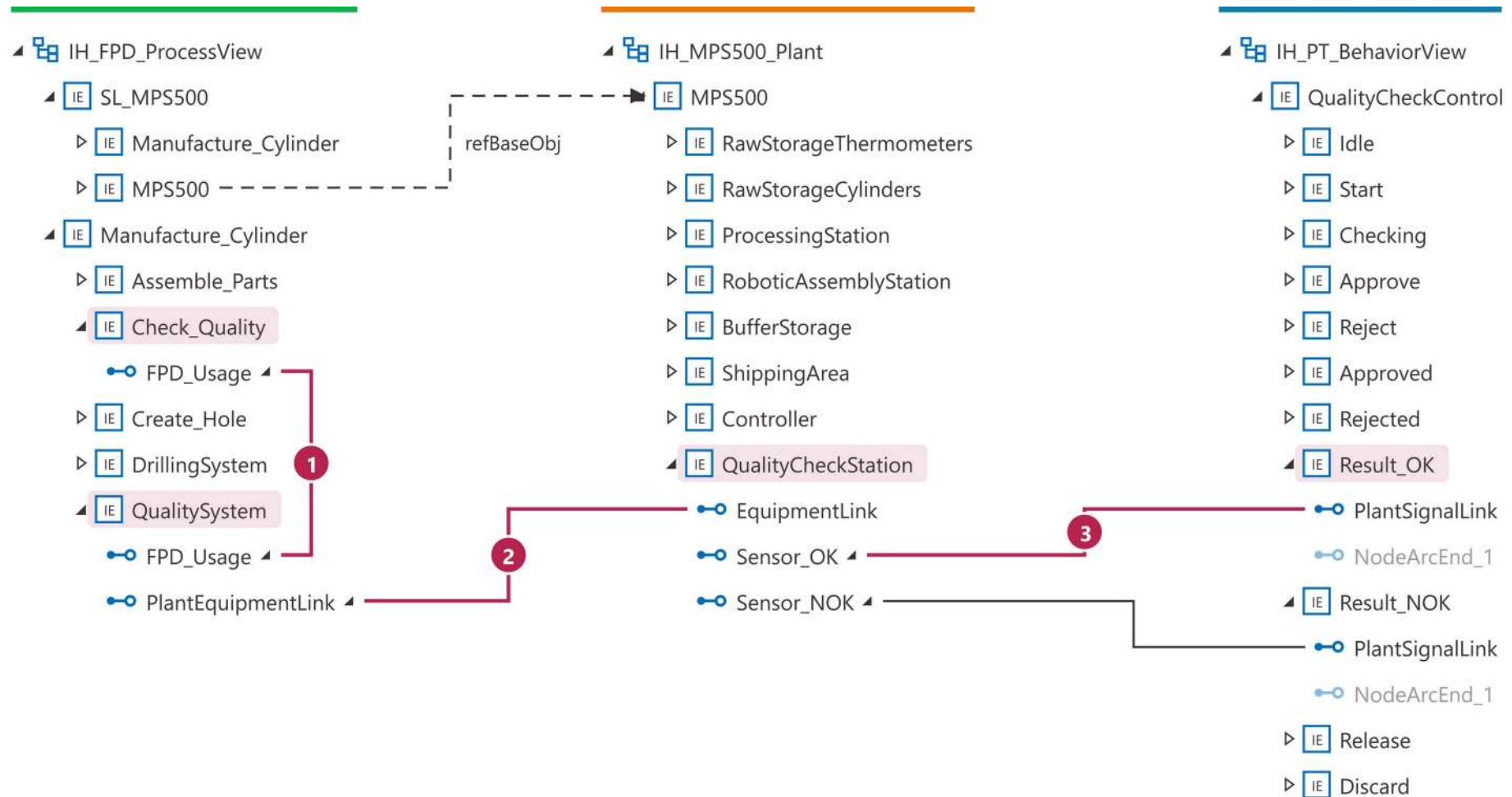
SL_Manufacture_Cylinder {FPD_Process}
«A» refObj = ID of the operator
«A» Distribute_Cylinder, Create_Hole, ...



Different type, different structure, may live in another hierarchy. The way back is optional.

The reference is an attribute whose value is the ID of the target. The target does not have to know. Delete the aspect, the base stays. Base and aspect: same type. Abstract and detail: any type.

Now we have a continuous chain



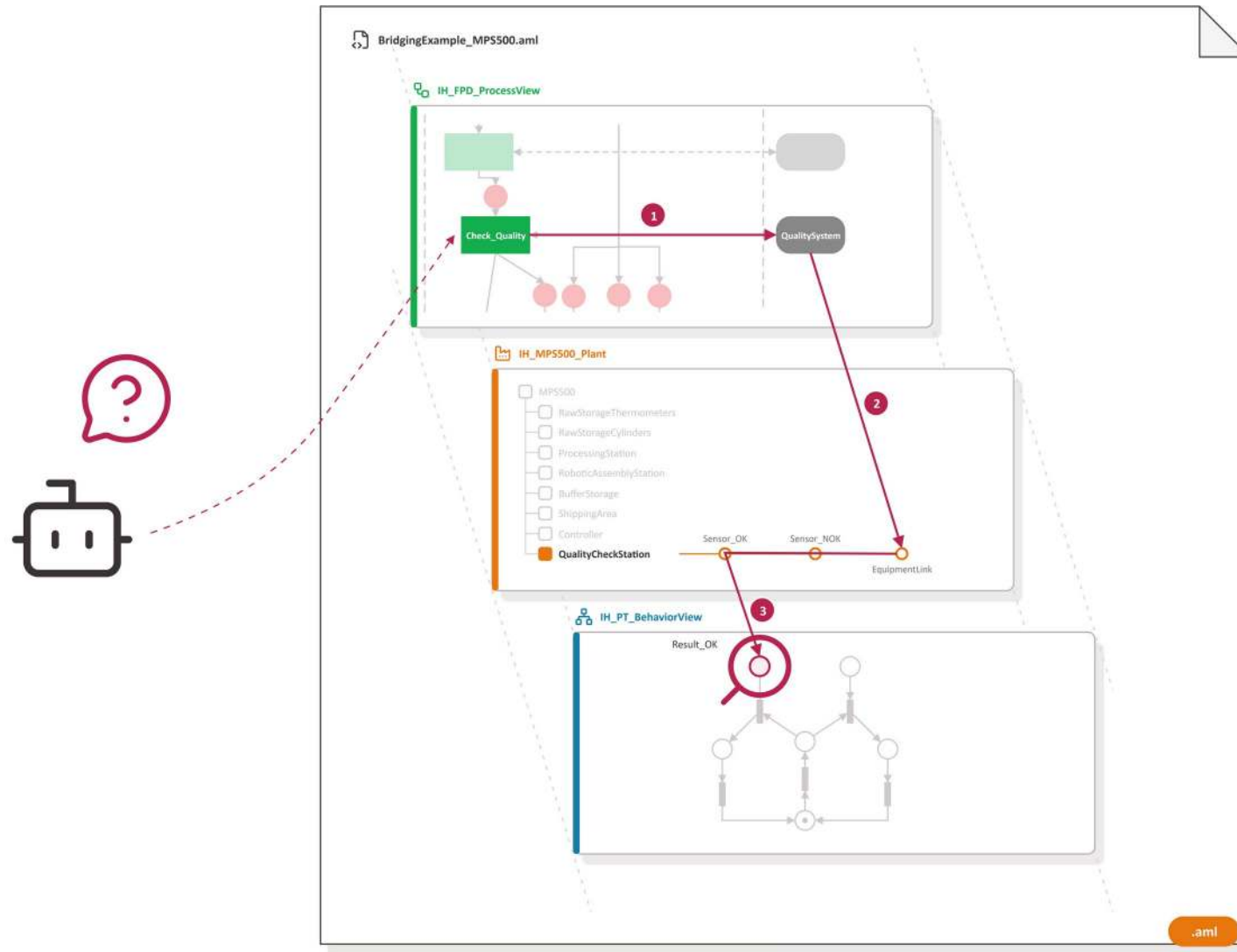
Process, plant, behavior: three instance hierarchies, one document, links across.

Process

Plant

Behaviour

Now an AI assistant can ask the file



Process

Plant

Behaviour

Nine questions, not the whole file

overview

what is in the document

tree

the containment below an element

search

elements by name, class or role

element

one element, with attributes and links

neighbours

what an element is connected to

path

how two elements are connected

class

a class and its base classes

class list

the libraries in the document

reference check

dangling and duplicate references

A local server reads the document and answers nine kinds of questions, read only. The assistant gets the answers, not the file.

Process

Plant

Behaviour

Let's use Claude via MCP

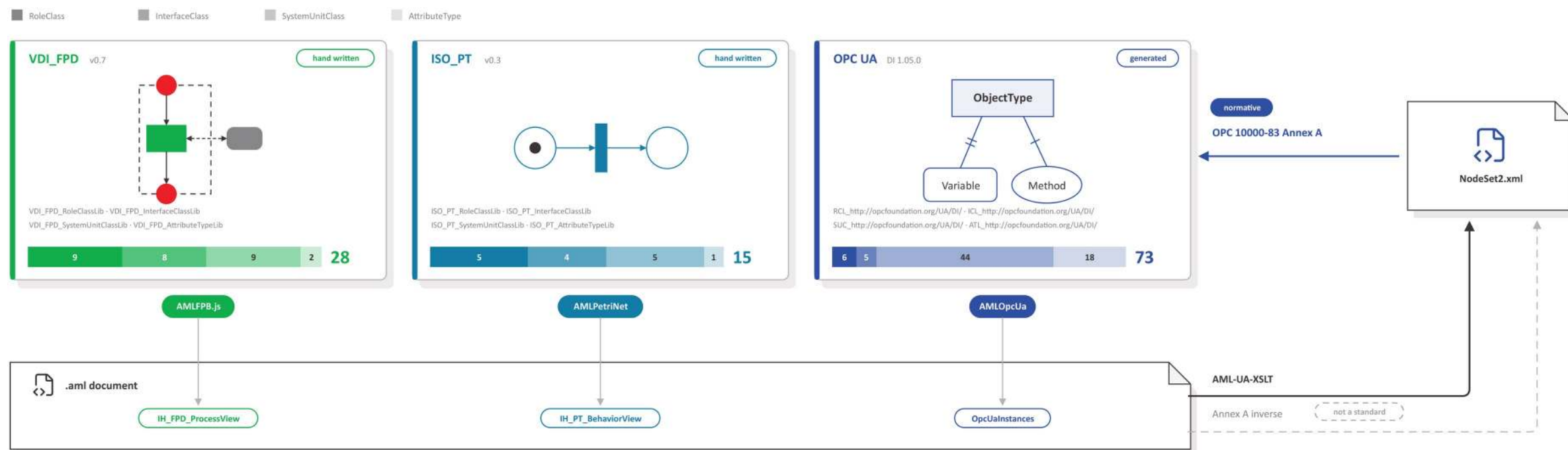
The screenshot displays the AML (AutomationML) software interface. The top menu bar includes File, Edit, View, Extras, Plugins, and Information. The Plugins menu is open, showing options like SvgExport, AMLFPBjs, AMLPetriNet, and AmlMcp. The main workspace is divided into several panels:

- Instance Hierarchy:** Lists various components like IH_FPD_ProcessView, IH_MPSS00_Plan, and IH_PT_Behavior.
- Diagram View:** Displays a Petri net diagram with places (red circles) and transitions (green rectangles). Places include Cylinder, Lid, Spring, Piston, and various Cylinder states (e.g., Cylinder_OK, Cylinder_NOK). Transitions include Distribute_Cylinder, Check_Head, Check_Quality, Assemble_Piston, and Distribute_Finished.
- Process View:** Shows a search bar and a list of processes, including IH_FPD_ProcessView and Manufacture_Cylinder.
- Plant View:** Displays a search bar and a list of plant components, including Cylinder_Distributor, Piston_Distributor, and others.

At the bottom, there is a status bar showing "VDI 3682 Findings" and a table with columns for RULE, ELEMENT, and MESSAGE. Below the status bar, there are three red buttons labeled "Process", "Plant", and "Behaviour".

What's next?

<AutomationML/>



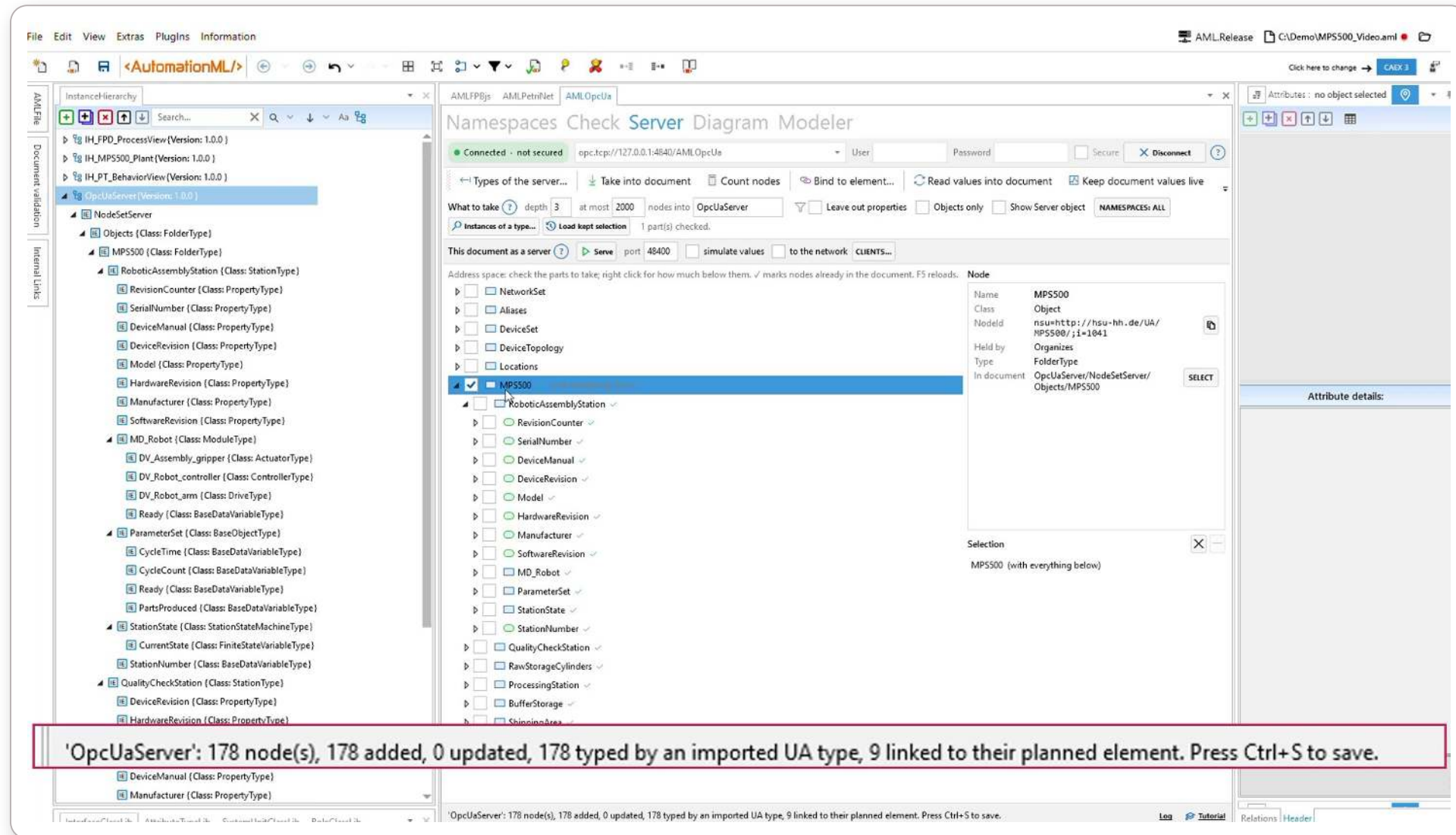
Process

Plant

Behaviour

OPC UA

What's next? OPC UA, in the same editor



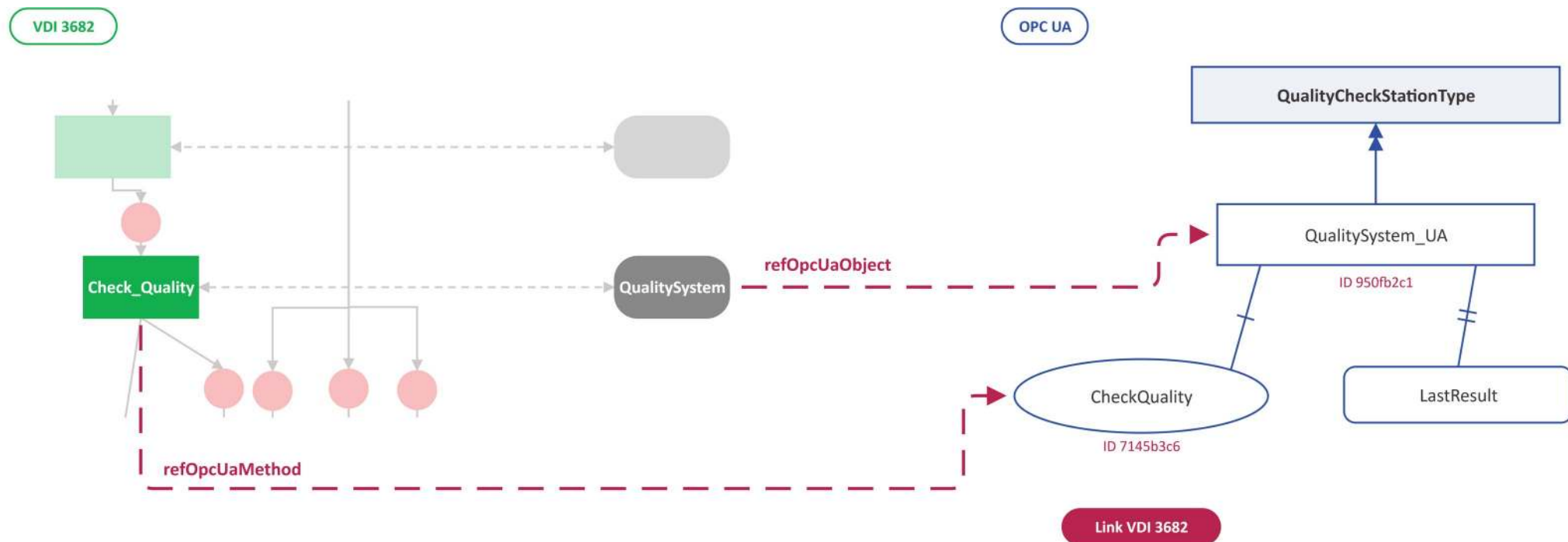
Process

Plant

Behaviour

OPC UA

And it attaches to the process, like everything else



IE QualitySystem (Class: FPD_TechnicalResource)

«A» refOpcUaObject = 950fb2c1-ea1c-45d5-9315-7c30e115b0e4

IE Check_Quality (Class: FPD_ProcessOperator)

«A» refOpcUaMethod = 7145b3c6-08f8-4052-8049-5d352f2fd303

«A» refObj (AttributeTypeLib: AutomationML_ObjectReferences)

«A» refOpcUaObject (AttributeDataType: xs:IDREF)

«A» refOpcUaMethod (AttributeDataType: xs:IDREF)

The mapping is not mine. It is Annex A.



THE NORM

OPC 10000-83, Part 83 of the OPC UA specification: UAFX OfflineEngineering.

THE ANNEX

Annex A, OPC UA Metamodel Mapping to AML.
Normative.

THE TOOL

Opc2Aml, the OPC Foundation's reference implementation, cited in the norm.

MINE

The tool around it: server, selection, import, instances, and the link to the process.

Annex A OPC UA Metamodel Mapping to AML for configuration (Normative)

A.1 Overview

OPC UA Servers and the underlying systems that they expose can be conveniently configured using *AutomationML* (AML) through the use of standardised AML libraries built to mirror the OPC UA modelling concepts defined in OPC 10000-3 and OPC 10000-5. Table A.1 shows the OPC UA modelling concept and the AML modelling concept used to realize the OPC UA counterpart in the AML libraries.

Table A.1 - Mapping OPC UA modelling concepts to AML

UA Modelling Concept	AML Modelling Concept
DataType	AttributeType
ObjectType	SystemUnitClass
VariableType	SystemInitClass

OPC 10000-83, v1.00.03, published 8 June 2026. The page of Annex A.

Process

Plant

Behaviour

OPC UA

Take it with you



All tools mentioned in this talk
and a playbook for creating your
own Plugins:

[hamiednabizada.github.io/aml-toolshelf](https://github.com/HamiedNabizada/aml-plugin-playbook)



Deep Dive into this topic:



aml-plugin-playbook

Everything needed to build tooling for your own graphical modelling language on top of AutomationML: a mapper between the language and CAEX, a browser modeler based on diagram-js, a web app, and a plugin for the AutomationML Editor that edits diagrams inside AML documents.

It is written so that you can hand the repository to a coding assistant together with a description of your language and get a working start instead of rediscovering the pitfalls. It works just as well read by a person.

What is in here