<AutomationML/>

The Glue for Seamless Automation Engineering
Collaboration of Tools for Production System Planning and PLC Programming by Using AutomationML

AutomationML User Conference 2018

Mario Thron, ifak e.V.
Klaus Hanisch, tarakos GmbH
Matthias Mueller, Mitsubishi Electric Europe B.V.
Agenda

Motivation – PLC programming
AutomationML – PPR Connector
Mapping AML to PLC program
Data Exchange
Requirements/Milestones
PLC Programming today

Production System Planning Tool

- Select components from libraries
- Arrange and combine them
- Define the tasks of those components
- Confirm/Validate the production system design

PLC Programming

- Import existing libraries of SDTs and FBs
- Instantiate SDTs and FBs from those libraries
- Assign values to the SDT members and FB inputs based on specification of the production planning system
- Program logic between those FBs
- Create SDTs and FBs if no library exists
Improving the PLC programming

The PLC programming for a single conveyor might not be so complicated and time consuming, but this changes when looking at huge production facilities.

Such facility consists of large numbers of conveyors, turntables, palletizers, etc.

In the PLC program the same SDTs and FBs are used to control the same types of conveyors. Only the values of SDT members and FB inputs are different for each instance.

If a digital description of the production system would exist and if a mapping to SDTs and FBs, between the component types and the tasks they are executing, would exist, the PLC program could be partly generated automatically.
PPR Concept

- AutomationML introduces the Product-Process-Resource concept in Whitepaper Part 1
- In order to structure complex plant engineering data, trisection of data into resources, processes and products has delivered proven performance in practice.
- Resources, Processes and Products are connected by the PPR-Connector
- This concept can be applied to the example below

**Resources:**
- Straight Conveyor
- Turntable

**Processes**
- Moving
- Turning
Details of the Mapping: Types

Mapping of Role Classes to Definitions

- Based on the Role Classes a mapping is created
- A Role Class derived from Resource is mapped to a definition of an SDT and an FB library
  - Attributes of the Role Class are mapped to Members of the SDT
- A Role Class derived from Process is mapped to a definition of an FB of the library
  - Attributes of the Role Class are mapped to In- and Outputs of the FB
Details of the Mapping: Instances

AutomationML Instance Hierarchy

② Mapping of Internal Elements to Instances

- Based on the Internal Elements in the Instance Hierarchy, which implements a Resource or a Process role, instances of SDTs and FBs are generated in the PLC program
  - Generate a POU for each resource
  - Create an Instance of the SDT mapped to this resource in the POU
    - Assign the values of the AML Attributes to the members of the instantiated SDT
  - Create an Instance of the FB mapped to the process in the POU
    - Assign the values of the AML Attributes to the members of the instantiated FB
Defining a reusable mapping and store it in AutomationML

- Development of a prototype tool that can import AutomationML Instance Hierarchy and extract Internal Elements with a PPR Relation
- Create a mapping library based on the Role Classes and System Unit Classes of the Internal Elements which can be reused when other Instance Hierarchies are imported

Instance Hierarchy filtered by Resources and Processes that are connected by PPR Connector

Elements of the Instance Hierarchy grouped by System Unit Class and Role Class

System Unit Class library that stores mapping of AutomationML data to PLC program relevant data such as function blocks and structured data types.
Where to get such data from?

Design 3D Layouts by wide ranged libraries for logistics and production planning

- Import of 2D/3D
- Design 3D Layout
- Create Animated 3D Visualization
- Present Videos/Virtual Reality
- Export of plant configuration

Animated Videos
Virtual Reality
Plant Konfiguration
Setup the factory layout

- Create 3D factory layout
- taraVRbuilder - Object types
- Position of all objects
- Attributes of taraVRbuilder objects

Conveying Goods (Products)

Position, Shape, Size (Ressource)

Behaviour (Process)
Collaboration of Tools for Production System Planning and PLC Programming

Link to Geometry (Collada), optional

Root-Element

Component Instance

Child-Elemente

predecessor-successor-relationship

taraVRbuilder-AML-Export - Base Structure
Add your individual RoleClasses

- Select object in factory layout
- Add requirements from imported RoleClassLibs to objects
- Optional: Set values for RoleClass-Attributes
From Layout to PLC

Design 3D Plant Layout

Import RoleClassLibs related to PLC

3D Planning Tool

Define RoleClassLibs
- Ressources
- Processes

PLC-Programming

- Import InstanceHierarchy including RoleClasses
- Match Roles to FB and SDT
- (Partly) generate the PLC program automatically

Add Roles and attributes to objects
Project Planning based on AutomationML

- **Context**
  - Planning of industrial production systems involves specifications of
    - Functional requirements, e.g. definitions of the materials to be handled, definitions of production and transportation processes, structural definitions of production resources, …
    - Non-functional requirements, e.g. definition of safety and security constraints, delivery times, budgets, …
  - AutomationML can be used as exchange-format for engineering data

- **Problem statement**
  - How can AutomationML be used to formalize data about the process of creation or modification of industrial production systems?
The goal

- Formalize milestone plans

### Project Planning based on AutomationML

<table>
<thead>
<tr>
<th>Key results</th>
<th>2016</th>
<th>2017</th>
<th>Involved staff</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
</tr>
<tr>
<td>Product definition</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process definition</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rough layout of site</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Planning Milestone (M1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supplier identification</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Detailed engineering</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtual commissioning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Engineering Milestone (M2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Commissioning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tuning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documentation (as built)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acceptance test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Handover Milestone (M3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First month production</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fine tuning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Milestone (M4)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- to be exchanged
- to be assigned via ERP (not exchanged)
Motivation

- Planning data is exchanged over large parts of the supplier-chain
- Formalized exchange of time plans improves accuracy of equipment delivery
Solution

- Re-use the plant structure created with other (3D-planning) tools
- Specify a project time plan in terms of milestones
- Add relations between equipment structures and result types to milestones
- Define a formalization of the planning data in AutomationML
Project Planning based on AutomationML

- Detail of Solution
Define a formalization of the planning data in AutomationML
Project Planning based on AutomationML

- Hand-writing files is possible, but inconvenient.
Project Planning based on AutomationML

- Use the right tools for the planning purposes

3D planning tool

Project planning tool
Demonstration of the data exchange and generation of the PLC program.

- Export from Tarakos VRbuilder
- Import to Mitsubishi Mapping Tool prototype
- Confirm existing mappings and define missing additional mappings
- Generate PLC program framework in PLC Engineering Tool GX Works3
**3D Planning Tool**
- Create 3D Layout by using taraVRbuilder standard libraries objects (about 500)
- Import your AML-RoleClassLibs ➔ Add roles and attribute values to objects

**PLC Programming Tool**
- Create the mapping from scratch from Instance Hierarchies with Resources and Processes using AutomationML RoleClassLibs
- Create function blocks and structured data types for them
- Reuse the mapping for other Instance Hierarchies which are using the same Resources and Processes

**Milestone Planning tool**
- Re-use the plant structure created with a (3D-planning) tool
- Specify a project time plan in terms of milestones
- Add relations between equipment structures and result types to milestones

---

**2D or 3D CAD-Layout (optional)**

- **Product**
- **Process**
- **Ressource**
Conclusion & Outlook

- Based on the PPR Connector and a mapping, the generation of the PLC program framework can be automated and simplified.
- Effort for entering the same parameters in several tools can be reduced and errors can be prevented.
- After importing and generating the PLC program framework the Instance Hierarchy could be extended the reference into the PLC program.
- This could enable a roundtrip engineering where changes in one tool can be reflected back to the others:
  - Parameter changes in the PLC tool could be reflected back to the origin tool.
  - Additions to the factory layout can be imported to the PLC tool and the program can be updated.
  - Tools at later stages of the engineering chain could use the AutomationML data which includes factory layout and PLC information as well.
Thanks for your attention!

Parts of this conference contribution (plug-in for taraVR builder and project planning tool) have been funded by the German Federal Ministry of Education and Research (BMBF) in scope of the European ITEA3 project 15015 „ENTOC“ (https://entoc.eu/).