



<AutomationML/>

**The Glue for Seamless
Automation Engineering**

**Whitepaper AutomationML
Part 4: AutomationML Logic**
Document Identifier: WP Logic, V 1.6.0

State: June 2020

© AutomationML consortium

Version 1.6.0 June 2020

Contact: www.automationml.org

Table of content

| | |
|---|----|
| Table of content..... | 3 |
| List of Tables | 9 |
| Introduction..... | 11 |
| 1 Scope | 13 |
| 2 Normative references..... | 13 |
| 3 Terms, definitions and abbreviated terms..... | 14 |
| 3.1 Terms and definitions..... | 14 |
| 3.2 Abbreviated terms..... | 15 |
| 4 Conformity | 15 |
| 5 Overview | 16 |
| 5.1 Logic information in production system engineering | 16 |
| 5.2 Logic models in production system engineering..... | 17 |
| 5.3 Storing logic models in AML logic XML | 18 |
| 5.4 Referencing logic information | 20 |
| 6 Logic models | 21 |
| 6.1 General | 21 |
| 6.2 Gantt charts | 21 |
| 6.2.1 General..... | 21 |
| 6.2.2 Graphical elements | 21 |
| 6.2.3 Chart structure..... | 21 |
| 6.2.4 Logic information | 21 |
| 6.2.5 Logic information within Gantt chart..... | 22 |
| 6.3 Activity-on-node networks..... | 22 |
| 6.3.1 General..... | 22 |
| 6.3.2 Graphical elements | 22 |
| 6.3.3 Node structure..... | 23 |
| 6.3.4 Network structure | 23 |
| 6.3.5 Logic information | 23 |
| 6.3.6 Logic information within activity-on-node networks..... | 23 |
| 6.4 Timing diagrams | 24 |
| 6.4.1 General..... | 24 |
| 6.4.2 Graphical elements | 24 |
| 6.4.3 Diagram structure..... | 25 |
| 6.4.4 Logic information | 25 |
| 6.4.5 Logic information within timing diagrams | 26 |
| 6.5 Sequential function charts..... | 26 |
| 6.6 Function block diagrams | 27 |
| 6.7 Mathematical expression | 27 |
| 7 AML logic XML schema description | 27 |
| 7.1 Schema overview..... | 27 |
| 7.1.1 Use of IEC 61131-10 schema | 27 |
| 7.1.2 Schema versioning..... | 28 |
| 7.2 Root element "AMLLogic" | 29 |
| 7.2.1 General..... | 29 |
| 7.2.2 Attributes | 29 |

| | | |
|-------|--|----|
| 7.2.3 | Sub-element "WriterHeader" | 29 |
| 7.2.4 | Sub-element "Types" | 29 |
| 7.2.5 | Sub-element "Documentation" | 30 |
| 7.3 | Complex type "FunctionBlock" | 30 |
| 7.3.1 | General | 30 |
| 7.3.2 | Attributes | 30 |
| 7.3.3 | Sub-element "Parameters" | 30 |
| 7.3.4 | Sub-element "Vars" | 30 |
| 7.3.5 | Sub-element "MainBody" | 30 |
| 7.3.6 | Complex type "ParameterSet" | 30 |
| 7.3.7 | Complex type "VariableDecl" | 31 |
| 7.4 | Complex Type "IML" | 32 |
| 7.4.1 | General | 32 |
| 7.4.2 | Attributes | 32 |
| 7.4.3 | Sub-element "Resource" | 32 |
| 7.4.4 | Sub-element "TimeInformation" | 33 |
| 7.4.5 | Choice of "IML****" element | 33 |
| 7.4.6 | Complex type "IMLStep" | 33 |
| 7.4.7 | Complex type "IMLTransition" | 34 |
| 7.4.8 | Complex type "IMLSimultaneousDivergence" | 34 |
| 7.4.9 | Complex type "IMLSimultaneousConvergence" | 35 |
| 7.5 | Complex Type "MathematicalExpression" | 35 |
| 7.5.1 | General | 35 |
| 7.5.2 | Attributes | 35 |
| 7.5.3 | Sub-element "VariableMapping" | 35 |
| 7.5.4 | Sub-element "MathML" | 35 |
| 7.6 | Simple type "LogicModelTypeEnum" | 36 |
| 7.7 | Simple type "TimeUnion" | 36 |
| 7.8 | Simple type "TimeFormatEnum" | 36 |
| 7.9 | Simple type "UuidString" | 36 |
| 8 | Storing logic models | 36 |
| 8.1 | General | 36 |
| 8.2 | Storing Gantt charts in AML logic XML | 36 |
| 8.2.1 | Common rules | 36 |
| 8.2.2 | Storing the start of a Gantt chart | 37 |
| 8.2.3 | Storing bars | 37 |
| 8.2.4 | Storing arrows | 38 |
| 8.2.5 | Storing successor bars | 38 |
| 8.2.6 | Storing predecessor bars | 40 |
| 8.3 | Storing activity-on-node networks in AML logic XML | 41 |
| 8.3.1 | Common rules | 41 |
| 8.3.2 | Storing the start of an activity-on-node network | 41 |
| 8.3.3 | Storing nodes | 42 |
| 8.3.4 | Storing arrows | 42 |
| 8.3.5 | Storing successor nodes | 44 |
| 8.3.6 | Storing predecessor nodes | 45 |

| | | |
|--------|---|----|
| 8.4 | Storing timing diagrams in AML logic XML | 47 |
| 8.4.1 | Common rules | 47 |
| 8.4.2 | Storing the timeline of a timing diagram | 47 |
| 8.4.3 | Storing resources and resource states | 47 |
| 8.4.4 | Storing lifelines | 48 |
| 8.4.5 | Storing the time signal and the resource signal | 51 |
| 8.5 | Storing sequential function charts in AML logic XML | 52 |
| 8.5.1 | Common rules | 52 |
| 8.5.2 | Storing variables..... | 52 |
| 8.6 | Storing function block diagrams in AML logic XML | 53 |
| 8.6.1 | Common rules | 53 |
| 8.6.2 | Storing variables..... | 53 |
| 8.7 | Storing mathematical expressions in AML logic XML..... | 54 |
| 8.7.1 | Common rules | 54 |
| 8.7.2 | Storing variables..... | 55 |
| 8.7.3 | Storing variable mappings..... | 56 |
| 8.7.4 | Storing mathematical expressions | 56 |
| 9 | Meta information about AML logic XML writer tools | 56 |
| 10 | Extensions of AML classes for logic | 57 |
| 10.1 | General | 57 |
| 10.2 | AutomationMLLogicRoleClassLib | 58 |
| 10.2.1 | General | 58 |
| 10.2.2 | RoleClass InterlockingTargetGroup | 58 |
| 10.2.3 | RoleClass InterlockingSourceGroup | 58 |
| 10.2.4 | RoleClass LogicModelObject | 59 |
| 10.3 | AutomationMLLogicInterfaceClassLib | 60 |
| 10.3.1 | General | 60 |
| 10.3.2 | InterfaceClass LogicModelInterface | 60 |
| 10.3.3 | InterfaceClass SequencingLogicModelInterface | 60 |
| 10.3.4 | InterfaceClass BehaviourLogicModelInterface | 61 |
| 10.3.5 | InterfaceClass InterlockingLogicModelInterface | 61 |
| 10.3.6 | InterfaceClass LogicModelElementInterface | 61 |
| 10.3.7 | InterfaceClass VariableInterface | 61 |
| 10.3.8 | InterfaceClass InterlockingVariableInterface | 62 |
| 10.4 | AutomationMLPLCOpenXMLInterfaceClassLib | 62 |
| 10.4.1 | General | 62 |
| 10.4.2 | InterfaceClass VariableInterface | 63 |
| 10.5 | AutomationMLInterfaceClassLib | 63 |
| 10.5.1 | General | 63 |
| 10.5.2 | InterfaceClass InterlockingConnector | 63 |
| 10.5.3 | InterfaceClass PLCOpenXMLInterface | 63 |
| 11 | Referencing AML logic XML documents..... | 64 |
| 11.1 | General | 64 |
| 11.2 | Referencing logic information | 64 |
| 12 | Linking AML objects with interlocking information | 65 |
| 12.1 | General | 65 |

| | |
|--|-----|
| 12.2 Referencing interlocking information | 65 |
| Annex A (informative) Examples for storing logic models in AML logic XML | 67 |
| A.1 Example for storing Gantt charts | 67 |
| A.1.1 General..... | 67 |
| A.1.2 Storing of activities without predecessor and successor relation | 67 |
| A.1.3 Storing of an activity sequence | 68 |
| A.1.4 Storing of an activity sequence with divergences | 70 |
| A.1.5 Storing of an activity sequence with convergences | 71 |
| A.2 Example for storing activity-on-node networks | 72 |
| A.2.1 General..... | 72 |
| A.2.2 Storing of activities without predecessor and successor relation | 72 |
| A.2.3 Storing of an activity sequence | 73 |
| A.2.4 Storing of an activity sequence with divergences | 74 |
| A.2.5 Storing of an activity sequence with convergences | 75 |
| A.3 Example for storing timing diagrams | 78 |
| A.3.1 General..... | 78 |
| A.3.2 Example of storing internal signal | 78 |
| A.3.3 Example of storing external signal | 79 |
| A.3.4 Example of storing signal between two resource states flows | 80 |
| A.4 Example for storing sequential function charts | 82 |
| A.5 Example for storing function block diagrams | 85 |
| A.6 Example for storing mathematical expressions | 87 |
| Annex B (informative) Examples for referencing logic information | 93 |
| B.1 General | 93 |
| B.2 Referencing logic information expressed as logic models..... | 93 |
| B.2.1 General..... | 93 |
| B.2.2 Referencing logic information stored in one FunctionBlock..... | 93 |
| B.2.3 Referencing logic information, which is composed of several FunctionBlocks..... | 94 |
| B.2.4 Referencing logic information, which is composed of several AML logic XML documents..... | 95 |
| B.3 Referencing logic information as a part of logic models | 96 |
| B.3.1 General..... | 96 |
| B.3.2 Referencing a variable | 96 |
| B.3.3 Referencing a logic element..... | 97 |
| B.4 Referencing logic information as a part of already referenced logic models | 98 |
| Annex C (informative) Examples for referencing interlocking information | 101 |
| C.1 General | 101 |
| C.2 Interlocking information..... | 102 |
| C.3 Referencing interlocking information without interlocking condition | 102 |
| C.4 Referencing interlocking information with interlocking condition | 104 |
| Annex D (normative) XML representation of AML standard libraries..... | 108 |
| D.1 General | 108 |
| D.2 AutomationMLLogicRoleClassLib | 108 |
| D.3 AutomationMLLogicInterfaceClassLib | 110 |
| D.4 AutomationMLPLCopenXMLInterfaceClassLib | 111 |
| Annex E (normative) XML representation of AML logic XML schema | 112 |

List of Figures

| | |
|---|-----|
| Figure 1 – Overview of the engineering data exchange format AML | 11 |
| Figure 2 – Example of system representation with roles of information in AML..... | 17 |
| Figure 3 – Logic models in AML | 18 |
| Figure 4 – Storing logic models in AML logic XML | 19 |
| Figure 5 – Modelling elements of the AML logic XML | 20 |
| Figure 6 – Model elements of Gantt charts..... | 21 |
| Figure 7 – Information provided by Gantt charts | 22 |
| Figure 8 – Model elements of activity-on-node networks | 23 |
| Figure 9 – Information provided by activity-on-node networks | 24 |
| Figure 10 – Model elements of timing diagrams..... | 25 |
| Figure 11 – Information provided by timing diagrams | 26 |
| Figure 12 – AML logic schema overview | 28 |
| Figure 13 – Root element "AMMLLogic"..... | 29 |
| Figure 14 – Complex type "FunctionBlock" | 30 |
| Figure 15 – Complex type "ParameterSet" | 31 |
| Figure 16 – Complex type "VariableDecl"..... | 31 |
| Figure 17 – Complex type "IML" | 32 |
| Figure 18 – Complex type "IMLStep"..... | 33 |
| Figure 19 – Complex type "IMLTransition" | 34 |
| Figure 20 – Complex type "IMLSimultaneousDivergence" | 34 |
| Figure 21 – Complex type "IMLSimultaneousConvergence"..... | 35 |
| Figure 22 – Complex type "MathematicalExpression"..... | 35 |
| Figure 23 – AutomationMLLogicRoleClassLib..... | 58 |
| Figure 24 – AutomationMLLogicInterfaceClassLib..... | 60 |
| Figure 25 – AutomationMLPLCOpenXMLInterfaceClassLib | 63 |
| Figure A.1 – Flow rate of valves | 87 |
| Figure A.2 – Example for storing a mathematical expression | 92 |
| Figure B.1 – Referencing logic information (as SFC) stored in one FunctionBlock | 93 |
| Figure B.2 – XML text of the CAEX file for referencing logic information stored in one FunctionBlock | 94 |
| Figure B.3 – Referencing logic information, which is composed of several FunctionBlocks..... | 94 |
| Figure B.4 – Referencing logic information which is composed of several AML logic XML documents | 95 |
| Figure B.5 – XML text of the CAEX file for referencing logic information, which is composed of several AML logic XML documents | 96 |
| Figure B.6 – Referencing a variable | 96 |
| Figure B.7 – XML text of the CAEX file for referencing a variable..... | 97 |
| Figure B.8 – Referencing a logic element..... | 97 |
| Figure B.9 – XML text of the CAEX file for referencing a logic element..... | 98 |
| Figure B.10 – Referencing a variable of an already referenced logic model..... | 99 |
| Figure B.11 – XML text of the CAEX file for referencing a variable of an already referenced logic model..... | 100 |
| Figure C.1 – Example manufacturing system..... | 101 |
| Figure C.2 – Example interlocking source group and interlocking target group | 102 |
| Figure C.3 – Referencing interlocking information without interlocking condition | 103 |

| | |
|--|-----|
| Figure C.4 – XML text of the CAEX file for referencing interlocking information without interlocking condition | 104 |
| Figure C.5 – Referencing interlocking information with interlocking condition | 105 |
| Figure C.6 – XML text of the CAEX file for referencing interlocking information with interlocking condition..... | 107 |
| Figure C.7 – Linking logical interface with physical interface (extension to Figure C.5) | 107 |
| Figure C.8 – XML text of the CAEX file for linking logical interface with physical interface (extension to Figure C.6) | 107 |

List of Tables

| | |
|--|----|
| Table 1 – Abbreviated terms | 15 |
| Table 2 – Mapping the start of a Gantt chart to IML element | 37 |
| Table 3 – Mapping of Gantt chart bars to IML elements | 37 |
| Table 4 – Mapping of Gantt chart arrow to IML elements | 38 |
| Table 5 – Mapping of Gantt chart bar with one or more successor bars to IML elements | 39 |
| Table 6 – Mapping of Gantt chart bar with one and more predecessor bars to IML elements | 40 |
| Table 7 – Mapping of the start of an activity-on-node network to IML elements | 41 |
| Table 8 – Mapping of activity-on-node network nodes to IML elements | 42 |
| Table 9 – Mapping of activity-on-node network arrows to IML elements | 44 |
| Table 10 – Mapping of activity-on-node network successor nodes to IML elements | 44 |
| Table 11 – Mapping of activity-on-node network predecessor nodes to IML elements | 46 |
| Table 12 – Mapping of the timeline of a timing diagram to IML elements | 47 |
| Table 13 – Mapping of timing diagram resources and resource states to IML elements | 48 |
| Table 14 – Mapping of timing diagram lifelines to IML elements | 48 |
| Table 15 – Mapping of the time signal and the resource signal to IML elements | 51 |
| Table 16 – Storing variable of a sequential function chart in AML logic XML | 53 |
| Table 17 – Storing variable of a function block diagram in AML logic XML | 54 |
| Table 18 – Storing variable of a mathematical expression in AML logic XML | 55 |
| Table 19 – Storing variable mappings of a mathematical expression in AML logic XML | 56 |
| Table 20 – Storing a mathematical expression in AML logic XML | 56 |
| Table 21 – Meta information about each AML logic XML writer tool | 57 |
| Table 22 – RoleClass InterlockingTargetGroup | 58 |
| Table 23 – RoleClass InterlockingSourceGroup | 59 |
| Table 24 – RoleClass LogicModelObject | 59 |
| Table 25 – InterfaceClass LogicModelInterface | 60 |
| Table 26 – InterfaceClass SequencingLogicModelInterface | 60 |
| Table 27 – InterfaceClass BehaviourLogicModelInterface | 61 |
| Table 28 – InterfaceClass InterlockingLogicModelInterface | 61 |
| Table 29 – InterfaceClass LogicModelElementInterface | 61 |
| Table 30 – InterfaceClass VariableInterface | 62 |
| Table 31 – InterfaceClass InterlockingVariableInterface | 62 |
| Table 32 – InterfaceClass VariableInterface | 63 |
| Table 33 – InterfaceClass InterlockingConnector | 63 |
| Table 34 – InterfaceClass PLCOpenXMLInterface | 64 |
| Table A.1 – Storing of the Gantt chart example "activities without predecessor and successor relations" | 68 |
| Table A.2 – Storing of the Gantt chart example "activity sequence" | 69 |
| Table A.3 – Storing of the Gantt chart example "activity sequence with divergence" | 70 |
| Table A.4 – Storing of the Gantt chart example "activity sequence with convergences" | 71 |
| Table A.5 – Storing of the activity-on-node network example "activities without predecessor and successor relations" | 72 |
| Table A.6 – Storing of the activity-on-node network example "activity sequence" | 73 |
| Table A.7 – Storing of the activity-on-node network example "activity sequence with divergence" | 74 |
| Table A.8 – Storing of the activity-on-node network example "activity sequence with convergences" | 76 |

| | |
|---|----|
| Table A.9 – Storing of the timing diagram example "transition from a state change to the subsequent state" | 78 |
| Table A.10 – Mapping of the timing diagram example "two external signals fired with delay of three seconds" | 79 |
| Table A.11 – Storing of the timing diagram example "signal fired by one resource state and consumed by another" | 81 |
| Table A.12 – Example for storing sequential function chart | 83 |
| Table A.13 – Example for storing a function block diagram | 85 |

Introduction

The data exchange format defined in IEC 62714 (Automation Markup Language (AML)) is an XML schema-based data format and has been developed in order to support the data exchange between engineering tools in a heterogeneous engineering tool landscape. IEC 62714-1 gives an overview about the format.

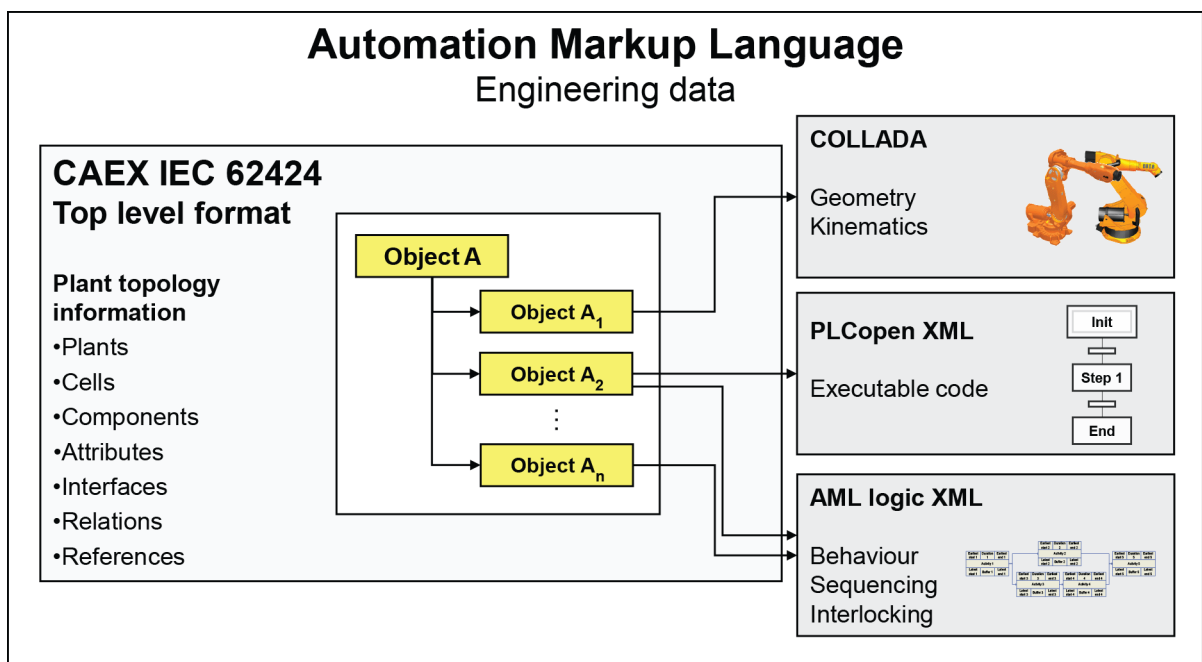
The goal of AML is to interconnect engineering tools from the existing heterogeneous tool landscape in their different disciplines, e.g. mechanical plant engineering, electrical design, process engineering, process control engineering, HMI development, PLC programming, robot programming, etc.

AML stores engineering information following the object-oriented paradigm and allows modelling of physical and logical plant components as data objects encapsulating different aspects. An object may consist of other sub-objects and may itself be part of a larger composition or aggregation. Typical objects in plant automation comprise information on topology, geometry, kinematics, and logic, whereas logic comprises sequencing, behaviour, and control.

AML combines existing industry data formats that are designed for the storage and exchange of different aspects of engineering information. These data formats are used on "as-is" basis within their own specifications and are not branched for AML needs.

The core of AML is the top-level data format CAEX that connects the different data formats. Therefore, AML has an inherent distributed document architecture.

Figure 1 illustrates the basic AML architecture and the distribution of topology, geometry, kinematic, and logic information.



IEC

Figure 1 – Overview of the engineering data exchange format AML

Owing to the different aspects of AML, IEC 62714 consists of different parts focussing on different aspects.

- IEC 62714-1: Architecture and general requirements
This part specifies the general AML architecture, the modelling of engineering data, classes, instances, relations, references, hierarchies, basic AML libraries and extended AML concepts.
- IEC 62714-2: Role class libraries
This part specifies additional AML libraries.
- IEC 62714-3: Geometry and kinematics
This part specifies the modelling of geometry and kinematics information.
- IEC 62714-4: Logic
This part specifies the modelling and referencing of logic information.

Further parts may be added in the future in order to interconnect further data standards to AML.

Clause 5 gives an informative overview of this part of the standard.

Clause 6 gives a normative description of the considered logic models.

Clause 7 gives a normative description of the AML logic XML schema, with which logic models can be stored.

Clause 8 specifies the normative provisions to store the logic models in AML logic XML.

Clause 9 defines how to store meta information about the source tool directly into the AML logic XML document.

Clause 10 defines a logic related role class library and interface class library.

Subclause 11 gives a normative description regarding referencing logic information in AML logic XML documents.

Clause 12 gives a normative description regarding referencing interlocking information in AML logic XML documents.

Annex A provides examples for the storage of logic models in AML logic XML.

Annex B describes the referencing methods for logic information.

Annex C describes the referencing methods for interlocking information.

Annex D gives a normative XML representation of the libraries defined in this document.

Annex E gives a normative XML representation of the AML logic XML schema defined in this document.

1 Scope

This part of IEC 62714 specifies the integration of logic information as part of an AML model for the data exchange in a heterogeneous engineering tool landscape of production systems.

This document specifies three types of logic information: sequencing, behaviour, and interlocking information.

This document deals with the six following sequencing and behaviour logic models (covering the different phases of the engineering process of production systems) and how they are integrated in AML: Gantt chart, activity-on-node network, timing diagram, Sequential Function Chart (SFC), Function Block Diagram (FBD), and mathematical expression.

This document specifies how to model Gantt chart, activity-on-node network, and timing diagram and how they are stored in Intermediate Modelling Layer (IML).

NOTE 1 With this, it is possible to transform one logic model into another one. A forward transformation supports the information enrichment process and reduces or avoids a re-entry of information between the exchanging engineering tools.

NOTE 2 Mapping of other logic models, e.g. event-driven logic models like state charts, onto IML is possible.

This document specifies how interlocking information is modelled (as interlocking source and target groups) in AML. The interlocking logic model is stored in Function Block Diagram (FBD).

This document specifies the AML logic XML schema that stores the logic models by using IEC 61131-10.

This document specifies how to reference PLC programs stored in PLCOpen XML documents.

This document does not define details of the data exchange procedure or implementation requirements for the import/export tools.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61131-3, *Programmable controllers – Part 3: Programming languages*

IEC 61131-10, *Programmable controllers – Part 10: PLC open XML exchange format*

IEC 62714-1:2014¹, *Engineering data exchange format for use in industrial automation systems engineering – Automation markup language – Part 1: Architecture and general requirements*

W3C. *Extensible Markup Language (XML) 1.0 (Third Edition)*, W3C Recommendation 04 February 2004 [online]. Edited by T. Bray et al., February 2004 [viewed on 2020-05-14]. Available at <http://www.w3.org/TR/2004/REC-xml-20040204>

¹ First edition. This first edition has been replaced in 2018 by a second edition IEC 62714-1:2018, *Engineering data exchange format for use in industrial automation systems engineering – Automation markup language – Part 1: Architecture and general requirements*.

W3C. *Mathematical Markup Language (MathML) Version 2.0 (Second Edition)*, W3C Recommendation 21 October 2003 [online]. Edited by D. Carlisle et al., October 2003 [viewed 2020-05-14]. Available at <<https://www.w3.org/TR/MathML2/>>

INTERNET ENGINEERING TASK FORCE (IETF). RFC 5646: *Tags for Identifying Languages* [online]. Edited by A. Phillips and M. Davis, September 2009 [viewed 2020-05-14]. Available at <https://tools.ietf.org/html/rfc5646>

INTERNET ENGINEERING TASK FORCE (IETF). RFC 4122, *A Universally Unique IDentifier (UUID) URN Namespace* [online]. Edited by P. Leach et al., July 2005 [viewed 2020-05-14]. Available at <https://tools.ietf.org/html/rfc4122>

PLCopen. *XML 2.01: XML formats for IEC 61131-3* [online]. Edited by K. Ketterle et al., May 2009 [viewed 2020-05-14]. Available at https://www.plcopen.org/system/files/downloads/tc6_xml_v201_technical_doc.pdf

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62714-1 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

3.1.1

AML logic object

AML object containing logic information

3.1.2

AML logic XML document

external XML document based on AML logic XML schema, which can be referenced from AML logic object

3.1.3

AML logic XML schema

XML schema for storing logic models of AML

3.1.4

behaviour information

information that describes the responses of the controlled system to the sequencing information and to other external interactions

3.1.5

behaviour logic model

representation of behaviour information

3.1.6

interlocking information

information that describes the instructions to the controlled system to avoid the system from causing harmful effects on humans and environment

3.1.7

interlocking logic model

representation of interlocking information

3.1.8**interlocking source group**

group of AML objects that indicate when actions are necessary to provide functional safety

3.1.9**interlocking target group**

group of AML objects that execute actions that are necessary to ensure functional safety

3.1.10**logic model element**

element of a logic model

3.1.11**logic information**

sequencing information, behaviour information, or interlocking information

3.1.12**logic model**

representation of logic information

3.1.13**mathematical expression**

meaningful combination of numbers, variables, and mathematical symbols to represent syntactically correct dependencies among them

Note 1 to entry: Mathematical expression may range from simple terms to complex equation systems.

3.1.14**sequencing information**

information that describes the instructions to the controlled system

3.1.15**sequencing logic model**

representation of sequencing information

3.2 Abbreviated terms

For the purpose of this document, the abbreviated terms of IEC 62714-1 apply as well as those listed in Table 1.

Table 1 – Abbreviated terms

| | |
|--------|--|
| SFC | Sequential Function Chart |
| FBD | Function Block Diagram |
| IML | Intermediate Modelling Layer |
| XSLT | Extensible Stylesheet Language Transformations |
| MathML | Mathematical Markup Language |

4 Conformity

To claim conformity to this part of IEC 62714 with respect to the support of AML, the requirements of Clause 6, 7, 8, 9, 10, 11, and 12 shall be fulfilled. In the scope of AML, an AML logic XML document shall conform to the AML logic XML schema defined in Clause 7 and Annex E.

A PLCopen XML document shall conform to the specification of PLCopen® XML 2.0 or 2.01, or to the specification of IEC 61131-10.

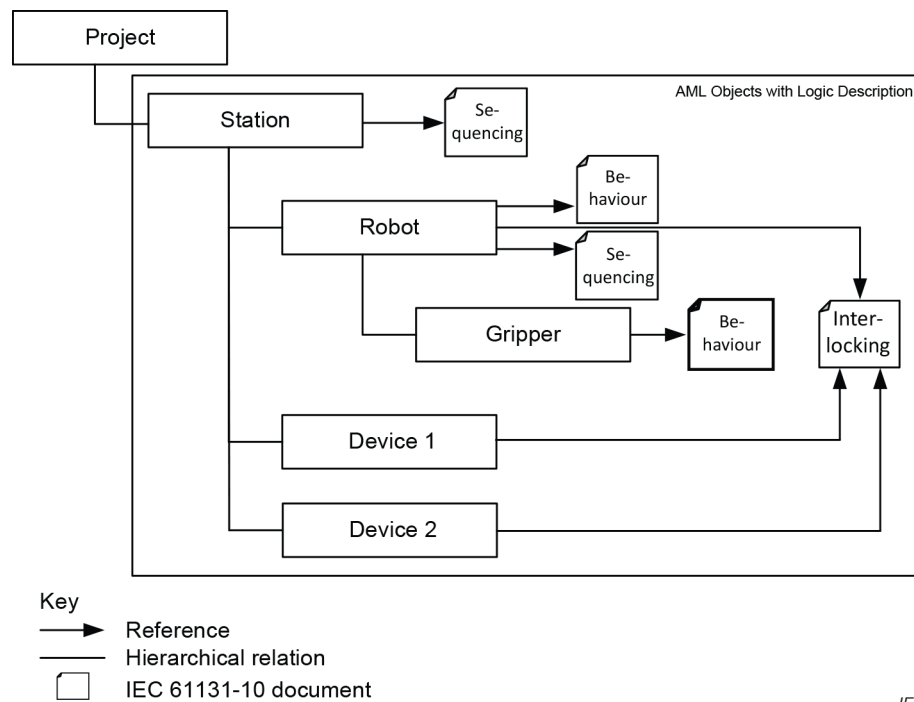
This document does not define provisions of the data exchange procedure or implementation requirements for the import/export tools.

5 Overview

5.1 Logic information in production system engineering

- This document focuses on the storage of logic information - an important aspect for electrical design, HMI development, PLC and robot control programming, for simulation purposes, and virtual commissioning.
- To support different phases in the iterative production system engineering process covering different levels of detail, AML needs to be able to cover logic information from different tools and disciplines. Thus, different roles of logic information belonging to a production system or to single components have to be stored. This variety of information can be classified into two main kinds: sequencing information and behaviour information. Those two roles of information serve different purposes, but those may overlap depending upon the point-of-view and utilization of elements in AML (see Figure 2).
 - Sequencing information is information that describes the instructions to the controlled system (or subsystem). In its most detailed form, it is often represented by a program executed in one or more controllers, e.g. robot controller or PLC. But the development of sequences starts earlier in the engineering process – for modelling on high level of abstraction – with models that represent the required operations of a larger scale unit (cell, line, plant etc.).
 - Behaviour information is information that describes the responses of the controlled system (or subsystem) to the sequencing information and to other external interactions. It is often represented by a given model which reacts on external input, e.g. behaviour of a gripper or valve. Those external inputs would trigger the behaviour or signalling of the gripper's states.
 - Besides sequencing information and behaviour information, interlocking information is a third complementary type of logic information (see Figure 2). It is an important part within the engineering of production systems to ensure their safe behaviour. It affects not only the safe interaction with humans and the environment, but it also helps to avoid unstable states of the systems possibly causing harmful effects on humans and environment or damages on machines and products.

A controlled system (or subsystem) may be represented by more than just one role of above information in AML - e.g. robot (see Figure 2).



IEC

Figure 2 – Example of system representation with roles of information in AML

5.2 Logic models in production system engineering

To be applicable within the engineering process of production systems, AML needs to support typical logic models – storing the logic information – which cover each phase of the engineering process (see Figure 3). These logic models are the following:

- Gantt charts are used mainly within early phases of the engineering process to represent the timing and duration of manufacturing processes on a high level of abstraction (see 6.2).
- Activity-on-node networks are applied mainly within early phases of the engineering process to represent the timing and interdependence of manufacturing processes with the capability to store timing information (see 6.3).
- Timing diagrams are used mainly within the later phases of the engineering process to describe the devices with their states and timing relationships (see 6.4). Real signals are introduced, and it is possible to express sequencing information.
- Sequential Function Charts (SFC) are one of the programming languages of the IEC 61131-3 mainly applied within later phases of production system engineering (see 6.5). Modelling of sequencing information as well as behaviour information is possible.
- Function Block Diagrams (FBD) are one of the programming languages of the IEC 61131-3 mainly applied within later phases of production system engineering (see 6.6). Modelling of interlocking information as well as behaviour information is possible.
- Mathematical expressions are used mainly within the later phases of the engineering process to describe mathematical, physical, chemical, or logical relationships of a device (see 6.7).

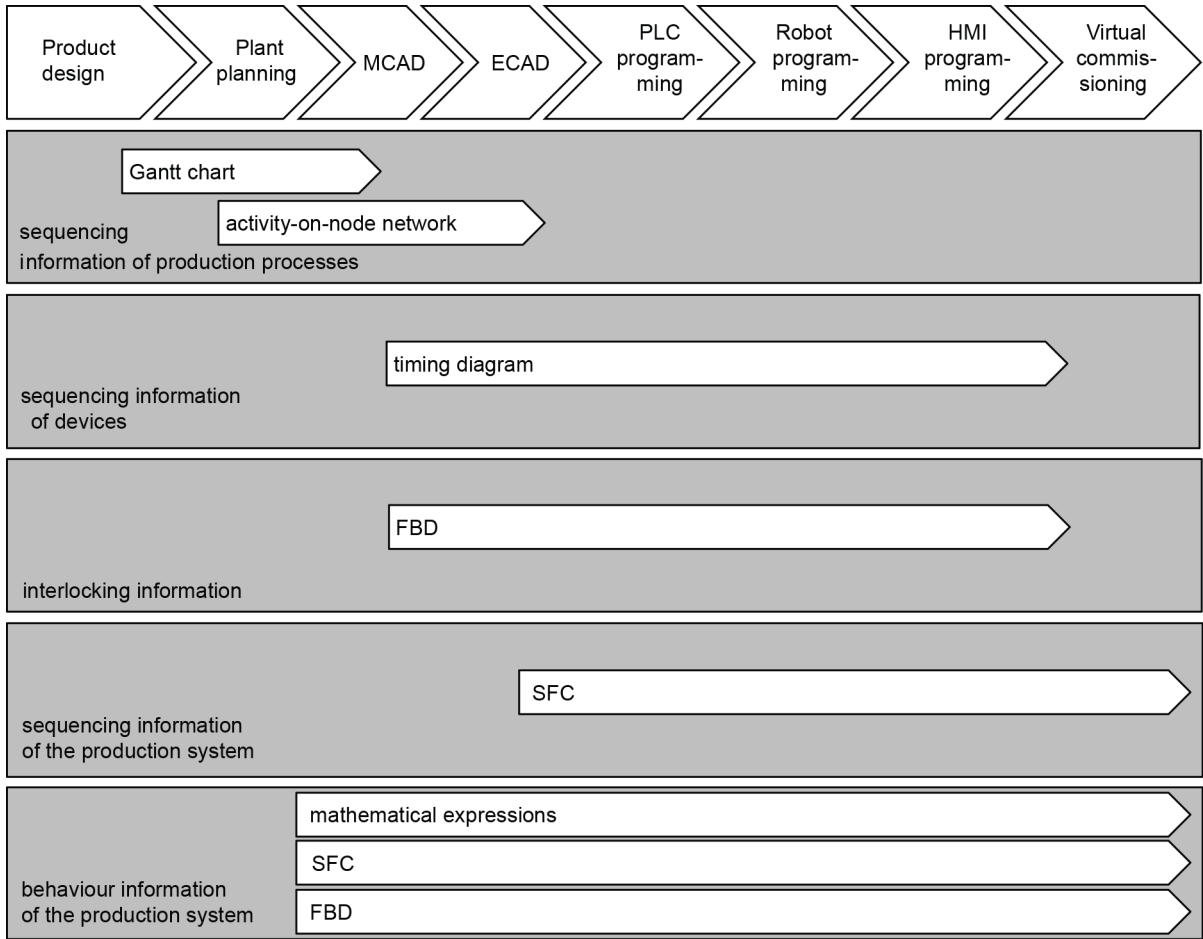


Figure 3 – Logic models in AML

5.3 Storing logic models in AML logic XML

It is intended to enable a transformation between the logic models. This is essential since the engineering process of production systems is a process of information enrichment. In the beginning, the sequencing information of a production system is only roughly described by using a Gantt chart and needs to be detailed in the next step. So, the Gantt chart is transformed into an activity-on-node network: the information from the Gantt chart is taken over and the complex timing conditions can be added now. Theoretically, this workflow can be carried out to the executable program, incrementing information in each step. A re-entry of information within the subsequent engineering tool is, therefore, not necessary. However, since the logic models have different modelling powers, forward and backward transformation between them, will cause information loss in particular cases, e.g. transforming an activity-on-node network to a Gantt chart.

An overview of the scope of this document is given in Figure 4.

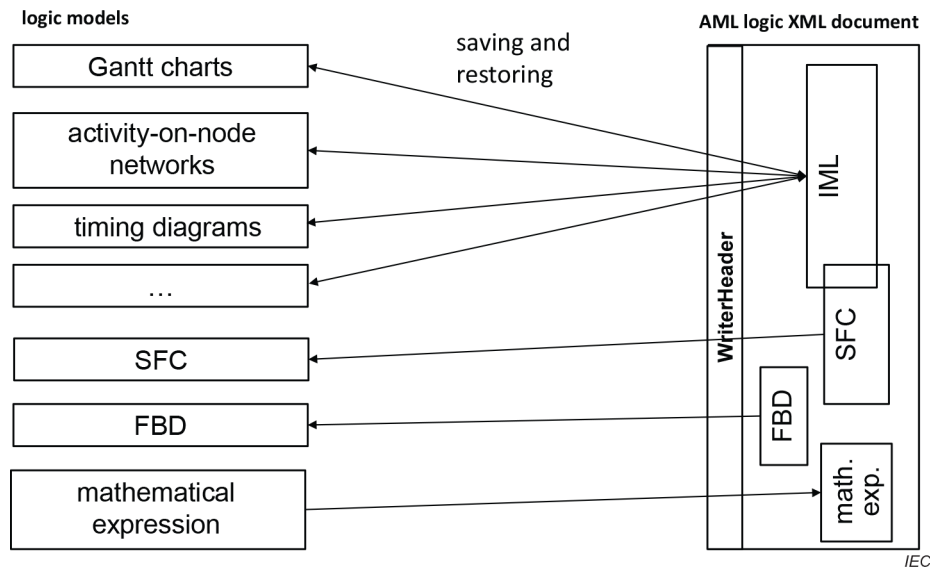
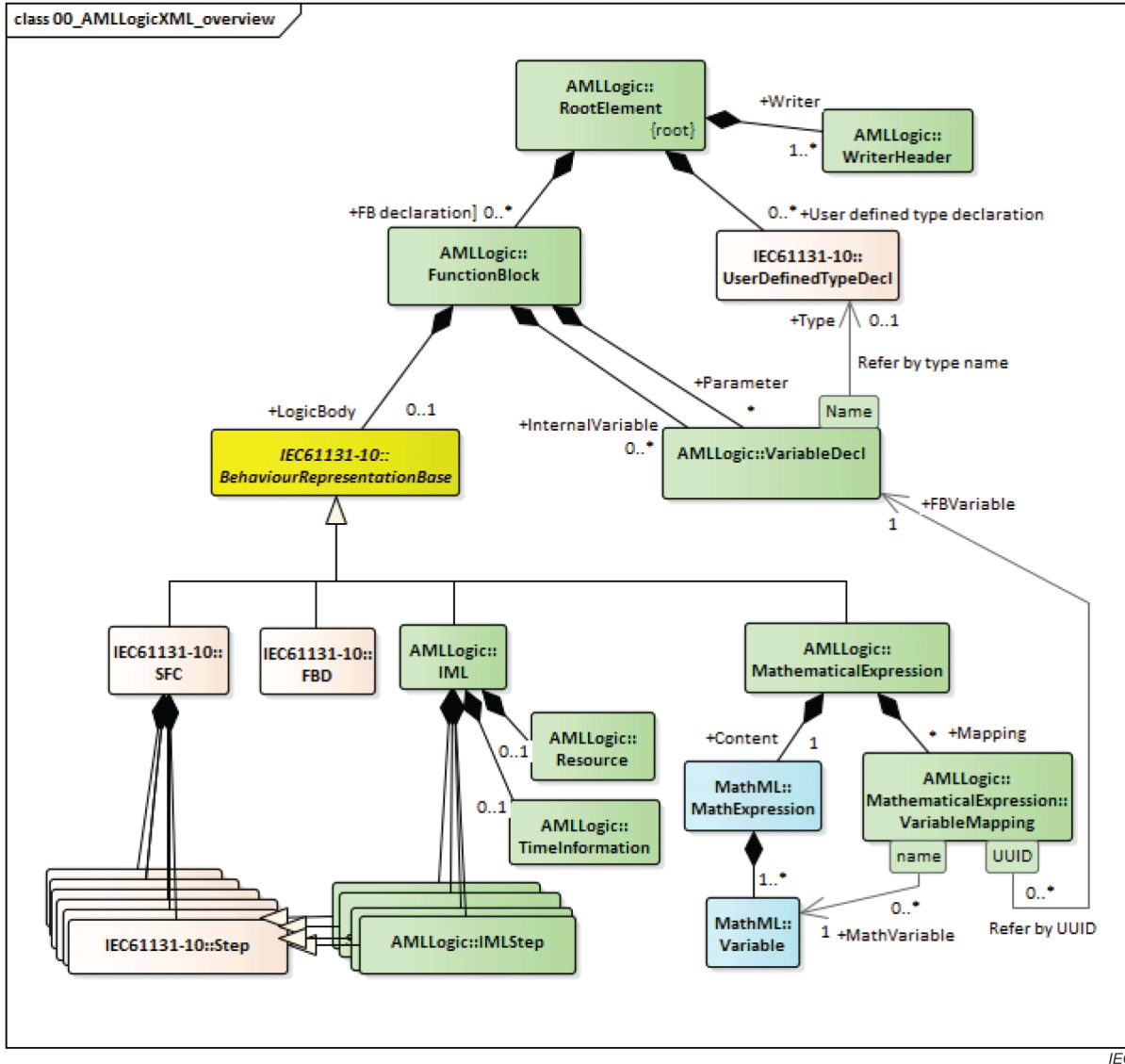


Figure 4 – Storing logic models in AML logic XML

NOTE To convert a SFC in the context of IML into a SFC in the context of IEC 61131-10, some XSLT files can be used that are available to public. When converting from IML SFC into an IEC 61131-10 SFC information might get lost.

AML uses IEC 61131-10 to store the logic models. To decouple IEC 61131-10 from the logic models, this part defines an Intermediate Modelling Layer (IML). Thus, IML forms the common model for logic models, which enhances the forward and backward transformation and the extensibility of AML for new logic models, like event-driven logic models like state charts.

Since this document is not simply about storing PLC programs, it adapts the usage of IEC 61131-10. An AML logic XML schema is used, specified in Clause 7, that allow the storage of the mentioned logic models. The modelling elements of AML logic XML to store the considered logic models are shown in Figure 5. To those modelling elements the logic models can be mapped. This is described in Clause 8. Annex A provides examples.



IEC

Figure 5 – Modelling elements of the AML logic XML

In addition, to enable a proper data exchange in the scope of AML, meta information on AML logic XML document level are necessary – called WriterHeader - to identify and characterize the document. Normative provisions are defined in Clause 9.

Since the AML logic XML is based on IEC 61131-10, it is also possible to use the other logic models provided by it. In this part of IEC 62714, only SFC (Sequential Function Charts) and FBD (Function Block Diagrams) are considered as relevant for the scope of this document.

Also, it is possible to store a logic model represented as a mathematical expression within the AML logic XML document. Either it is the logic model itself that contains the behaviour description of the system, or it can be used (or called) to provide a mathematical expression to one of the other logic models.

5.4 Referencing logic information

In the multi-document architecture of AML, logic information is stored in separate AML logic XML documents based on the IEC 61131-10 data format. Modelling logic information is, therefore, divided into two parts. First, an object, which has logic information, is modelled in CAEX, the top-level format. Second, an AML logic XML document has to be provided containing this logic information. Finally, a reference between the CAEX object and the AML logic XML

document or an object within needs to be created. Normative provisions for referencing all of the three roles of information are defined in Clause 10 and Subclause 1110.4.2. An informative overview is provided in Annex B.

For referencing interlocking information, additional provisions shall be considered. These are normatively defined in Clause 12. An informative overview is provided in Annex C.

6 Logic models

6.1 General

This clause defines the logic models that are considered in this part of IEC 62714. This comprises the graphical elements and their structuring rules, logic information they represent, and how this information is mapped to the model elements.

NOTE This document does not provide information about the execution logic, i.e. how the model elements are executed.

6.2 Gantt charts

6.2.1 General

Gantt charts are graphical representations of schedules typically used to model partially ordered sets of sequentially and concurrently running activities, which comprise of execution order and execution time of the activities. They do not provide means for modelling alternatives and cycles.

NOTE Gantt charts are also known as bar charts.

6.2.2 Graphical elements

In a Gantt chart, the following graphical elements shall be used (see Figure 6):

- one timeline (horizontal axis) realised as a global clock;
- one vertical axis listing all the names of the corresponding bars;
- zero or more horizontal bars;
- zero or more directed arrows.

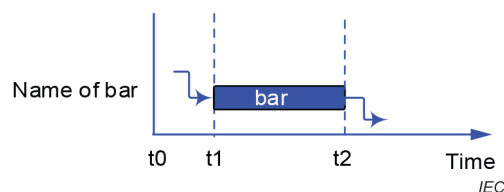


Figure 6 – Model elements of Gantt charts

6.2.3 Chart structure

The following structure for a Gantt chart shall be used:

- Each Gantt chart shall be represented by two orthogonal axes. The horizontal axis represents the time, having a direction, representing linear time. The vertical axis represents the list of bar names
- Each bar shall have its own line in the graphical representation with its corresponding name on the vertical axis.
- The timed start point, and timed end point of a bar shall correspond with the horizontal axis.
- Bars may be connected by arrows to other bars.
- An arrow shall be connected from the right side of one bar to the left side of another bar.

6.2.4 Logic information

Gantt charts shall comprise the following types of logic information:

- Activity information:
 - name;
 - start time;
 - finish time;
 - duration.
- Zero or more number of predecessor relation information between activities.
- Zero or more number of successor relation information between activities.

6.2.5 Logic information within Gantt chart

For the representation of the logic information within a Gantt chart, the following provisions apply (see Figure 7):

- An activity shall be represented by a bar.
 - Activity name shall be represented by the name of a bar listed on the vertical axis.
 - Activity start time shall be represented by the timed start point of a bar listed on the horizontal axis.
 - Activity finish time shall be represented by the timed end point of a bar listed on the horizontal axis.
 - Activity duration shall be represented by the length of a bar.
- A predecessor-successor-relation between activities shall be represented by an arrow.

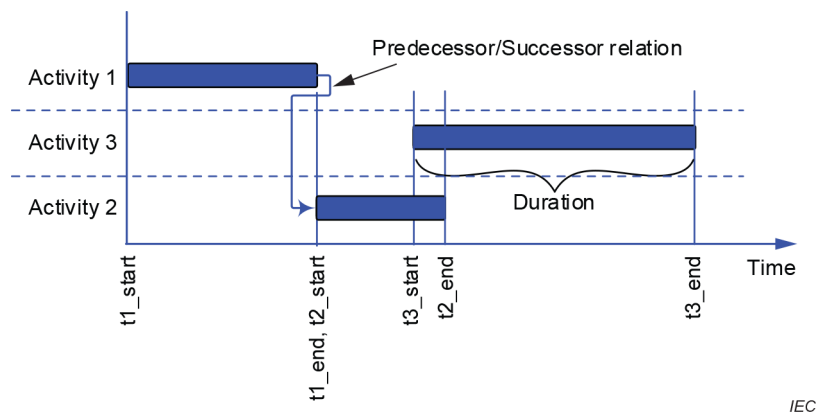


Figure 7 – Information provided by Gantt charts

6.3 Activity-on-node networks

6.3.1 General

Activity-on-node networks are used to describe and analyse temporal and causal relations of a set of interdependent activities. The end-start-relation between the nodes states that a node can only start after the previous node has ended. Activity-on-node networks do not provide means for modelling alternatives and cycles.

6.3.2 Graphical elements

In an activity-on-node network the following graphical elements shall be used (see Figure 8):

- zero or more nodes, represented by rectangles, each of them divided in seven boxes;
- zero or more directed arrows.

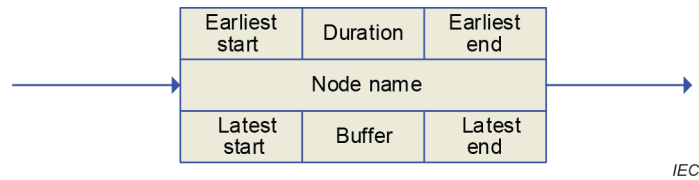


Figure 8 – Model elements of activity-on-node networks

6.3.3 Node structure

The following structure for a node shall be used:

- The earliest start (t_{es}) of a node should be located as a number in the box in the upper left corner of the node.
- The earliest end (t_{ee}) of a node should be located as a number in the box in the upper right corner of the node.
- The duration (t_d) of a node should be located as a number in the box in the upper centre part of the node and is calculated by the earliest start and earliest end of the node using the $t_d = t_{ee} - t_{es}$ formula
- The node name should be located in the box in the centre of the node.
- The latest start (t_{ls}) of a node should be located as a number in the box in the lower left corner of the node.
- The latest end (t_{le}) of a node should be located as a number in the box in the lower right corner of the node.
- The buffer value (t_b) of a node should be located as a number in the box in the lower centre part of the node and is calculated by the earliest end and the latest end of the node using the $t_b = t_{le} - t_{ee}$ formula.

6.3.4 Network structure

The following structure for an activity-on-node network shall be used:

- Nodes may be connected by arrows to other nodes.
- An arrow shall be connected from the right side of one node to the left side of another node.

6.3.5 Logic information

Activity-on-node networks shall comprise the following types of logic information:

- Activity information:
 - name;
 - earliest start time;
 - latest start time;
 - earliest end time;
 - latest end time;
 - duration;
 - buffer.
- Predecessor-successor relation information between activities.

6.3.6 Logic information within activity-on-node networks

For the representation of the logic information within an activity-on-node network, the following provisions apply (see Figure 9):

- An activity shall be represented by a node.
 - Activity name shall be represented by the name of a node.
 - Activity earliest start time shall be represented by the earliest start of a node.
 - Activity latest start time shall be represented by the latest start of a node.
 - Activity earliest end time shall be represented by the earliest end of a node.
 - Activity latest end time shall be represented by the latest end of a node.
 - Activity duration shall be represented by the duration of a node.
 - Activity delay shall be represented by the buffer of a node.
- A predecessor-successor-relation between activities shall be represented by an arrow.

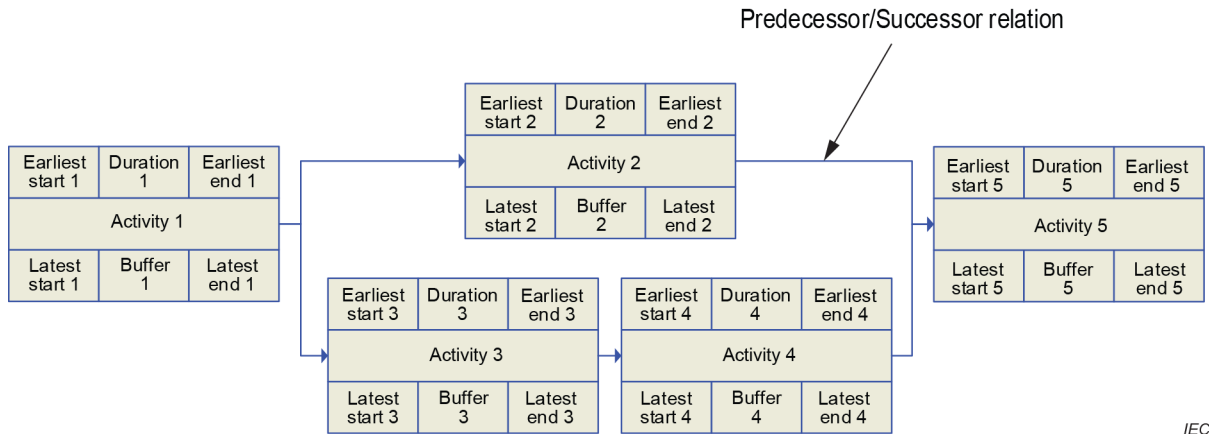


Figure 9 – Information provided by activity-on-node networks

6.4 Timing diagrams

6.4.1 General

Timing diagrams are used to describe the temporal and causal relations between signals and system states. Furthermore, they describe how changes of system states are affected by signals and how signals are generated by states changes. Timing diagrams enable the modelling of alternatives.

6.4.2 Graphical elements

In a timing diagram, the following graphical elements shall be used (see Figure 10):

- one or more lanes represented as columns;
- one or more levels represented as columns;
- zero or more directed and named arrows;
- one timeline (horizontal axis) realized as a global clock;
- one or more lifelines.

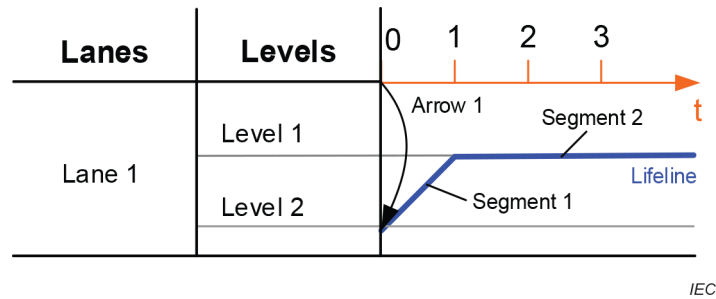


Figure 10 – Model elements of timing diagrams

6.4.3 Diagram structure

The following structure for a timing diagram shall be used:

- Each level shall have its own horizontal stripe. The names of the levels shall be put in the corresponding levels column.
- Levels that belong to one lane shall be grouped in the lanes column. The names of the lanes shall be put in the corresponding lanes column. One lane shall be represented by one or more levels.
- Each lane shall have one lifeline, representing the change of level over linear time.
- A lifeline shall be composed of one or more segments.
- A segment shall be represented by a straight line.
- An arrow shall connect:
 - The timeline with the starting point of one segment,
 - An end point of a segment with the starting point of the other segment in the same lifeline or other lifeline.
- The end point of a segment shall be the starting point of zero or more arrows.
- Zero or more arrows shall start at any point at the timeline.

6.4.4 Logic information

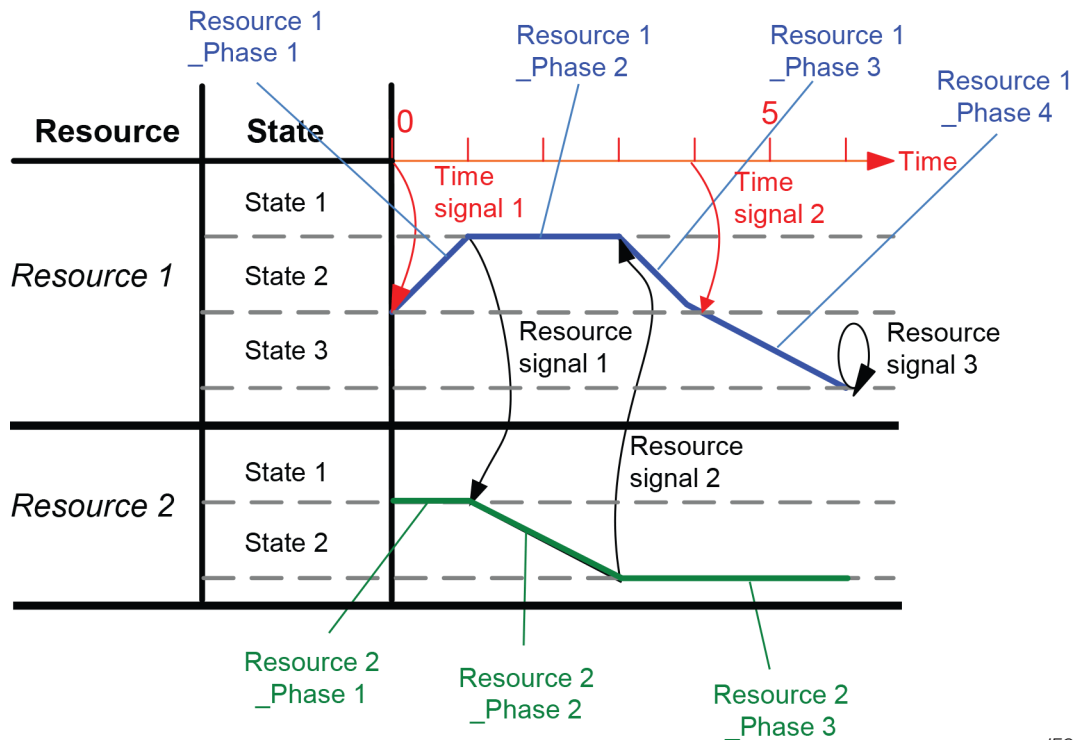
Timing diagrams shall comprise the following types of logic information:

- Resource information:
 - name.
- State information:
 - name;
 - phase (represents the duration);
 - state flow.
- Signal information:
 - name;
 - two different types: time and resource signal;
 - point in time when a signal occurs;
 - influence of a signal on the state of a resource.

6.4.5 Logic information within timing diagrams

For the representation of the logic information within a timing diagram, the following provisions apply (see Figure 11):

- A resource shall be represented by a lane.
 - Resource name shall be represented by the name of the lane.
- A state shall be represented by a level.
 - State name shall be represented by the name of the level.
 - Variation of state in time shall be presented by a lifeline.
 - A phase shall be represented by a segment.
- A signal shall be represented by an arrow.
 - Signal name shall be represented by the name of the arrow.
 - A time signal shall be represented by an arrow that starts at a point on the timeline. A resource signal shall be represented by an arrow that starts at the end of a segment of one lifeline.
 - Point in time when a signal occurs shall correspond with the x -axis.
 - Influence of a signal on the state of a resource shall be represented by a new beginning of a segment of the corresponding lifeline.



IEC

Figure 11 – Information provided by timing diagrams

6.5 Sequential function charts

Sequential Function Charts (SFC) describes causal and temporal dependencies of sequences of system steps and transitions. The modelling of complex sequences with loops and conditional execution is possible.

This document shall use SFCs as specified in IEC 61131-3.

6.6 Function block diagrams

Function Block Diagrams (FBD) describes logical networks of function blocks respectively how input and output variables are connected functionally.

This document shall use FBDs as specified in IEC 61131-3. The usage of function blocks specified in IEC 61131-3 shall be allowed.

6.7 Mathematical expression

Mathematical expressions are an arrangement of numbers, absolute terms, or letters to express mathematical, physical, chemical, or logical relationships. A mathematical expression consists of one formula or a system of equations.

For defining mathematical expressions, this document shall use the Mathematical Markup Language (MathML) version 2.0 content part.

7 AML logic XML schema description

7.1 Schema overview

7.1.1 Use of IEC 61131-10 schema

This clause describes "AML logic XML schema", which is formally defined in Annex E.

AML logic XML schema imports the IEC 61131-10 schema and partially reuses its type definitions to represent logic models using variables, function blocks, sequential function charts, etc.

The IEC 61131-10 schema supports two extension mechanisms allowing incorporation of information in the exchange format. The first mechanism is the "AddData" element, which can be nested below various other elements. Its usage is intended for the extension of other elements, by which they are enriched with information, but which is not crucial for the element's exchange between tools. The second mechanism is using an XML schema technique called "abstract type substitution". A new element type can be derived from the abstract type and used as a substitution for the element type where the abstract type is specified.

AML logic XML schema succeeds both extension mechanisms of the IEC 61131-10 schema and naming conventions for elements naturally. The schema structure overview is shown in Figure 12 in UML class diagram style. Classes whose names start with the prefix "IEC61131_10_Ed1_0" represent types imported from the IEC 61131-10 schema. Also, classes whose names are written in italic style (also drawn in bright yellow) represent abstract types that shall be substituted with one of its extended non-abstract types.

- User-defined data type declaration and its reference from variable declaration are fully reused by AML Logic XML schema as shown in Figure 12.
- Logic descriptions and their base type of IEC 61131-10 are also reused by AML Logic XML schema as shown in Figure 12. Thus, all logic descriptions of IEC 61131-10 are all available in AML Logic XML. In addition, AML logic XML schema adds its specific logic models such as IML and mathematical expression by "abstract type substitution", that is the second extension mechanism succeeded from IEC 61131-10.
- AML Logic XML schema extends fundamental SFC elements of IEC 61131-10 to represent Intermediate Modelling Layer (IML) described in 5.3, as shown in Figure 12.



7.1.2 Schema versioning

Additionally, the root element "AMLLogic" has a "schemaVersion" attribute. On creating an XML document based on the schema, its version number shall be set to the attribute.

7.2 Root element "AMLLogic"

7.2.1 General

The root element "AMLLogic" represents a kind of project or library, which holds logic information. Its content is shown in Figure 13.

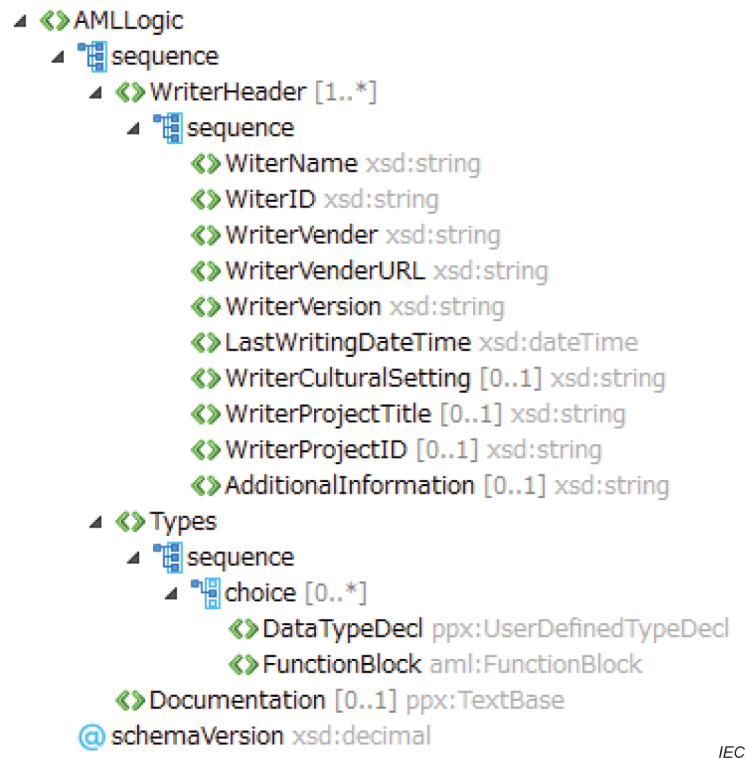


Figure 13 – Root element "AMLLogic"

7.2.2 Attributes

"schemaVersion" attribute shall hold the version number of AML Logic XML schema that the XML document is compliant to.

7.2.3 Sub-element "WriterHeader"

"WriterHeader" element represents meta information about the tool, which writes the XML document, as explained in Clause 9.

7.2.4 Sub-element "Types"

"Types" element is place holder for user-defined data type declaration and / or function block declaration.

"DataTypeDecl" element of "pplx:UserDefinedTypeDecl" type represents user-defined data type declaration of IEC 61131-3 language specification. Thus, any kind of data type available in IEC 61131-3 language specification can be defined in AML Logic XML as well. IEC 61131-10 document should be referenced for the detailed specification of the type "pplx:UserDefinedTypeDecl".

"FunctionBlock" element of "aml:FunctionBlock" type represents a kind of function block specialized for AML logic description, where only necessary features of IEC 61131-3 function block are supported. Its content is explained in 7.3.

7.2.5 Sub-element "Documentation"

"Documentation" element represents general description for the root "AMLLogic" element. The type "ppx: TextBase" is an abstract type imported from IEC 61131-10, which is substituted with plain text of "ppx: SimpleText" type by default.

7.3 Complex type "FunctionBlock"

7.3.1 General

Complex type "FunctionBlock" (as shown in Figure 14) represents a kind of function block specialized for AML logic description, where only necessary features of IEC 61131-3 function block are supported.

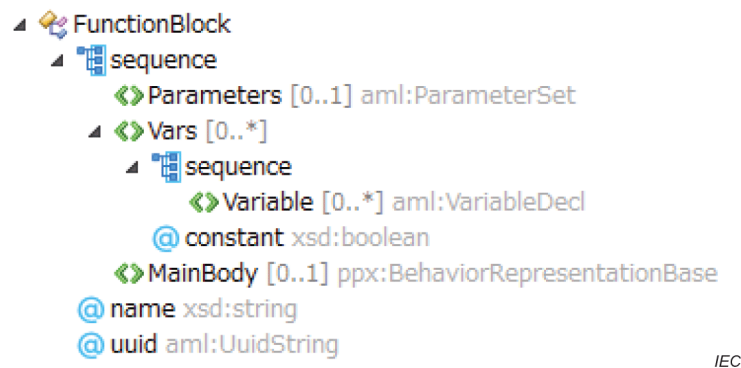


Figure 14 – Complex type "FunctionBlock"

7.3.2 Attributes

"name" attribute of "xsd:string" represents the type name of the function block.

"uuid" attribute of "aml:UuidString" represents universal unique identifier for the type definition of the function block. Simple type "UuidString" is explained in 7.9.

7.3.3 Sub-element "Parameters"

"Parameters" element of "aml:ParameterSet" type represents input / output / in-out variable declaration of the function block. The type "ParameterSet" is explained in 7.3.6 in detail.

7.3.4 Sub-element "Vars"

"Vars" element represents internal variable declarations of the function block.

"Variable" element of "aml:VariableDecl" type represents variable declaration. Complex type "aml:VariableDecl" is explained in 7.3.7.

"constant" attribute represents that variable declarations within the <Vars> element are variables with constant values.

7.3.5 Sub-element "MainBody"

"MainBody" element of "ppx:BehaviorRepresentationBase" type represents a logic model which shall be evaluated when the function block is executed. Any non-abstract complex type derived from "ppx:BehaviorRepresentationBase" can be substitute for the type of the element. Thus all logic models defined in IEC 61131-10 schema are also available in this function block of AML Logic XML, such as Sequential Function Chart (SFC), Function Block Diagram (FBD), Ladder Diagram (LD), and Structured Text (ST).

7.3.6 Complex type "ParameterSet"

Complex type "ParameterSet" (as shown in Figure 15) represents a declaration set of input variables, output variables, and in-out variables.

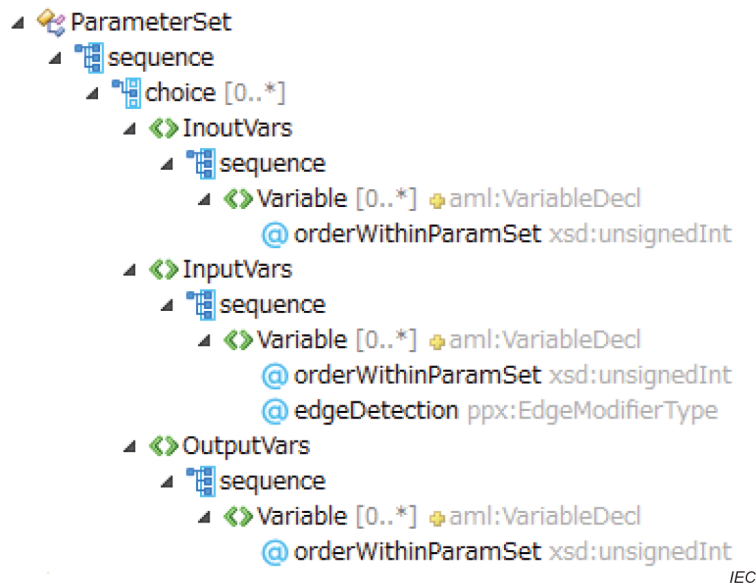


Figure 15 – Complex type "ParameterSet"

"InoutVars" element represents declaration section for In-Out variables. In terms of "Variable" element within the "InoutVars", its type is extended from "aml:VariableDecl" which is explained in 7.3.7, and added "orderWithinParamSet" attribute. The reason of this extension is because the order of parameters of a function block is significant but appearance order of elements of the same element type is not ensured to be significant in an XML document.

"InputVars" element represents declaration section for input variables of the function block. In terms of "Variable" element within the "InputVars", its type is extended from "aml:VariableDecl" explained in 7.3.7 as well, and added "edgeDetection" attribute in addition to the "orderWithinParamSet" attribute.

"OutputVars" element represents declaration section for output variables of the function block. In terms of "Variable" element within the "OutputVars", its type is extended from "aml:VariableDecl" explained in 7.3.7 as well, and added the "orderWithinParamSet" attribute as well.

7.3.7 Complex type "VariableDecl"

Complex type "VariableDecl" (as shown in Figure 16) represents variable declaration section.

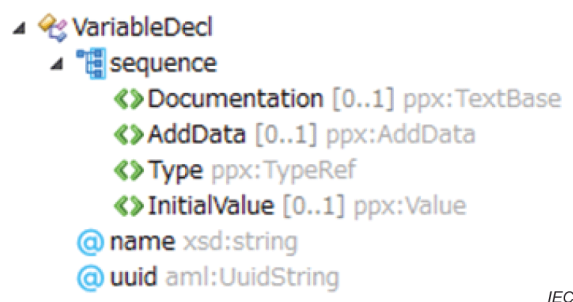


Figure 16 – Complex type "VariableDecl"

"Documentation" element of "ppx:TextBase" type represents the description of the variable.

"AddData" element of "ppx:AddData" type can be used to store any type of supplemental data.

"Type" element of "ppx:TypeRef" type represents reference to the type definition of the variable. It can be type name or instant declaration such as "ARRAY [0..8] OF INT". As for the details of the complex type "ppx:TypeRef", IEC 61131-10 should be referenced.

"InitialValue" element of "ppx:Value" type represents initial value for the variable. It can be single value, array value for array type, or structured value for structured data type depending on the data type of the variable. As for details of complex type "ppx:Value", IEC 61131-10 specification should be referenced.

"name" attribute of "xsd:string" type represents name of the variable.

"uuid" attribute of "aml:UuidString" represents universal unique identifier for the variable. Simple type "UuidString" is explained in 7.9.

7.4 Complex Type "IML"

7.4.1 General

Complex type "IML" (as shown in Figure 17) represents Intermediate Modelling Layer for certain multiple logic models, as described in 5.3. This complex type can be used as a substitute for the type of "MainBody" element of function block in AML Logic XML since it extends "ppx:BehaviourRepresentationBase", as described in Figure 12.

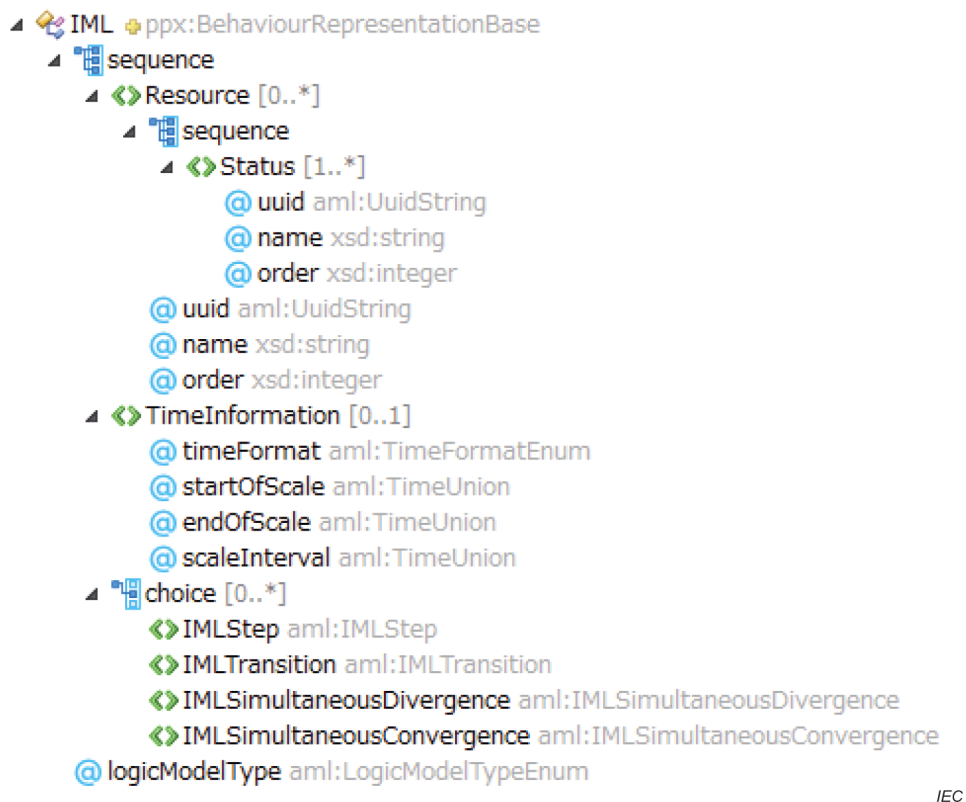


Figure 17 – Complex type "IML"

7.4.2 Attributes

"logicModelType" attribute of "aml:LogicModelTypeEnum" type represents what kind of logic model is represented with the IML. Simple type "LogicModelTypeEnum" is explained in 7.6.

7.4.3 Sub-element "Resource"

"Resource" element represents additional logic information necessary for timing diagram as described in 6.4.4.

- "uuid" attribute represents Universal Unique Identifier for the resource element. Simple type "UuidString" is explained in 7.9.
- "name" attribute represents the name of the resource.
- "order" attribute represents the order of the resource within the IML. Smaller numbers should be listed first.

"Status" elements represent possible status of the "Resource" element.

- "uuid" attribute represents Universal Unique Identifier for the status element. Simple type "UuidString" is explained in 7.9.
- "name" attribute represents the name of the status.
- "order" attribute represents the order of the status within the resource. Smaller numbers should be listed first.

7.4.4 Sub-element "TimeInformation"

"TimeInformation" represents logical time scale, especially for Gantt chart or timing diagram.

- "timeFormat" attribute of "aml:TimeFormatEnum" type represents format of the time expression within this IML. Simple type "aml:TimeFormatEnum" is explained in 7.8.
- "startOfScale" attribute of "aml:TimeUnion" type represents starting point of the time scale in the format specified by "timeFormat" attribute above. Simple type "aml:TimeUnion" is explained in 7.7.
- "endOfScale" attribute of "aml:TimeUnion" type represents ending point of the time scale in the format specified by "timeFormat" attribute above. Simple type "aml:TimeUnion" is explained in 7.7.
- "scaleInterval" attribute of "aml:TimeUnion" type represents scale interval of the time scale in the format specified by "timeFormat" attribute above. Simple type "aml:TimeUnion" is explained in 7.7.

7.4.5 Choice of "IML***" element

As the "choice" element, either of "IMLStep", "IMLTransition", "IMLSimultaneousDivergence", or "SimultaneousConvergence" can be selected. This schema structure is basically the same as "SFC" complex type of IEC 61131-10. However, alternatives for the choice are limited to what is necessary for IML representation. Also, the type of each alternative element is extended from original one of IEC 61131-10.

7.4.6 Complex type "IMLStep"

Complex type "IMLStep" (as shown in Figure 18) represents a phase or state of an activity. It extends "ppx:Step" of IEC 61131-10, which represents a STEP in the Sequential Function Chart.



Figure 18 – Complex type "IMLStep"

The following attributes are added as extension to the base type "ppx:Step" of IEC 61131-10.

- "uuid" attribute of "aml:UuidString" represents Universal Unique Identifier for the IMLStep element. Simple type "UuidString" is explained in 7.9.
- "startTime" attribute of "aml:TimeUnion" represents start time of the phase of the activity which corresponds to this IMLStep. Simple type "aml:TimeUnion" is explained in 7.7. The time shall be

represented in the format specified by the "timeFormat" attribute of the "TimeInformation" element explained in 7.4.4.

- "endTime" attribute of "xds:duration" represents end time of the phase of the activity which corresponds to this IMLStep. Simple type "aml:TimeUnion" is explained in 7.7. The time shall be represented in the format specified by the "timeFormat" attribute of the "TimeInformation" element explained in 7.4.4.
- "buffer" attribute of "xds:duration" represents a margin for the time span between "startTime" and "endTime".
- "refStartStatusId" attribute of "aml:UuidString" represents a reference to the source status of the segment of the pseudo waveform, which the "IMLStep" element represents in the case that the IML represents timing diagram. It concretely represents the "uuid" attribute's value of certain "Status" element within certain "Resource" element within the "IML" element.
- "refEndStatusId" attribute of "aml:UuidString" represents a reference to the destination status of the segment of the pseudo waveform which the "IMLStep" elements represents in the case that the IML represents timing diagram. It concretely represents the "uuid" attribute's value of certain "Status" element within certain "Resource" element within the "IML" element.

7.4.7 Complex type "IMLTransition"

Complex type "IMLTransition" (shown in Figure 19) represents a kind of predecessor/successor relationship between "IMLStep" elements. It extends "ppx:Transition" of IEC 61131-10, which represents a TRANSITION in Sequential Function Chart.



Figure 19 – Complex type "IMLTransition"

The following attributes are added as extension to the base type "ppx:Transition" of IEC 61131-10.

- "uuid" attribute of "aml:UuidString" represents Universal Unique Identifier for the IMLTransition element. Simple type "UuidString" is explained in 7.9.
- "refTime" attribute represents time when this transition should be triggered. Simple type "aml:TimeUnion" is explained in 7.7. The time shall be represented in the format specified by the "timeFormat" attribute of the "TimeInformation" element explained in 7.4.4.
- "refEndSegmentId" attribute of "aml:uuisString" type represents reference to the IMLStep element by UUID whose end should trigger this transition.

7.4.8 Complex type "IMLSimultaneousDivergence"

Complex type "IMLSimultaneousDivergence" extends "ppx:SimultaneousDivergence" of IEC 61131-10 (shown in Figure 20), which represents a starting point of parallel branch in Sequential Function Chart.

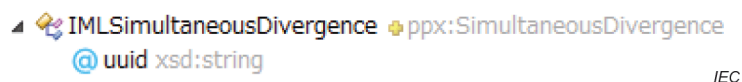


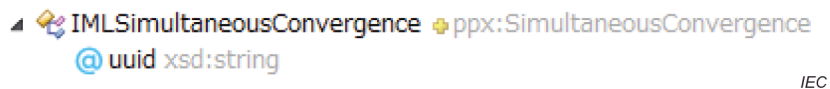
Figure 20 – Complex type "IMLSimultaneousDivergence"

The following attributes are added as extension to the base type "ppx:SimultaneousDivergence" of IEC 61131-10.

- "uuid" attribute of "aml:UuidString" represents Universal Unique Identifier for the IMLSimultaneousDivergence element. Simple type "UuidString" is explained in 7.9.

7.4.9 Complex type "IMLSimultaneousConvergence"

Complex type "IMLSimultaneousConvergence" extends "ppx:SimultaneousConvergence" of IEC 61131-10 (shown in Figure 21), which represents a ending point of parallel branch in Sequential Function Chart.



IEC

Figure 21 – Complex type "IMLSimultaneousConvergence"

The following attributes are added as extension to the base type "ppx:SimultaneousConvergence" of IEC 61131-10.

- "uuid" attribute of "aml:UuidString" represents Universal Unique Identifier for the IMLSimultaneousConvergence element. Simple type "UuidString" is explained in 7.9.

7.5 Complex Type "MathematicalExpression"

7.5.1 General

Complex type "MathematicalExpression" (as shown in Figure 22) represents logic description in mathematical formula expression as explained in 6.7. It can be used as substitution for the "MainBody" element of function block in AML logic XML, since it extends the abstract complex type "ppx:BehaviorRepresentaionBase", as described in Figure 12.



IEC

Figure 22 – Complex type "MathematicalExpression"

7.5.2 Attributes

"name" attribute of string type represents the name of the mathematical expression.

"uuid" attribute represents Universal Unique Identifier for the mathematical expression. Simple type "UuidString" is explained in 7.9.

7.5.3 Sub-element "VariableMapping"

"VariableMapping" element represents mapping between mathematical variable written in MathML and variable declared in the function block which this logic model belongs to.

"refMathMLVariableName" represents the name of the variable defined in MathML.

"refFBVariableUUID" represents UUID of the variable declared in the same function block. Simple type "UuidString" is explained in 7.9.

"direction" attribute represents mapping direction with enumerated string which only allows "In" or "Out".

7.5.4 Sub-element "MathML"

"MathML" element of string type represents the XML text data of MathML. It is stored just as plane text. Parsing with MathML schemata is out of scope of AML Logic XML schema.

7.6 Simple type "LogicModelTypeEnum"

Simple type "LogicModelTypeEnum" is restricted string type which only allows one of the enumerated strings, which are "GanttChart", "ActivityOnNodeNetwork", "TimingDiagram", or "Unknown". It is used to specify what kind of logic model type the IML represents.

7.7 Simple type "TimeUnion"

Simple type "TimeUnion" is a union whose members are composed of following time-related simple types built-in XML schema. XML schema specification shall be referenced for detailed value format specification of each built-in type.

- xsd:duration
 - The value format shall be "PnYnMnDTnHnMnS", for instance "P10DT15H" (10 days and 15 hours), "PT1M20S" (1 minute and 20 seconds), "PT95.4" (95,4 seconds), etc.
- xsd:time
 - The value format shall be "hh:mm:ss", for instance simply "15:45:30", or "15:45:30+02:00" to represent time zone offset from UTC.
- xsd:dateTime
 - The value format shall be "YYYY-MM-DDThh:mm:ss", for instance simply "2017-05-30T09:00:00", or "2017-05-30T09:00:00+02:00" to represent time zone offset from UTC.
- xsd:date
 - The value format shall be "YYYY-MM-DD", for instance "2017-09-24".

7.8 Simple type "TimeFormatEnum"

Simple type "TimeFormatEnum" is restricted string type which only allows one of the enumerated strings, which are "xsd:duration", "xsd:time", "xsd:dateTime", or "xsd:date". It is used to specify which time format shall be used for the "aml:TimeUnion" type explained in 7.7.

7.9 Simple type "UuidString"

Simple type "UuidString" is restricted string type to represent UUID in text form. No syntax restriction is actually defined.

It is recommended to implement the UUID as a GUID (as specified in RFC 4122).

8 Storing logic models

8.1 General

This clause defines rules for the storage of logic models to IML. A textual representation in the IML is presented for mapping of each logic model element. For examples, see Annex A.

8.2 Storing Gantt charts in AML logic XML

8.2.1 Common rules

For storing a Gantt chart in AML logic XML, the following provisions apply:

- One Gantt chart shall be mapped to one IML logic element of type "GanttChart" using the following provisions.
- The IML shall start with the "TimeInformation" element, defining the timeline of the Gantt chart, see 8.2.2.
- Each Gantt bar shall be represented by an IMLStep, see 8.2.3.
- Each Gantt arrow shall be represented by an IMLTransition, see 8.2.4.
- For predecessor and successor relations in a Gantt chart the following provisions apply:

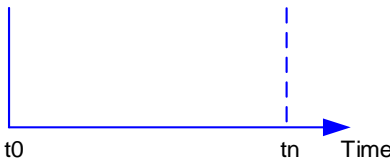
- If a Gantt bar has two or more successor bars, it shall be connected to them by an IMLTransition and IMLSimultaneousDivergence elements, see 8.2.5.
- If a Gantt bar has two or more predecessors, they shall be connected to them by an IMLSimultaneousConvergence and IMLTransition elements, see 8.2.6.
- If there are no explicit relations between Gantt bars, i.e. they are concurrent, the order of associated IMLStep shall be parallel and they shall have no ordering relation based on step-step transition sequences, see 8.2.5 and 8.2.6.

If predecessor/successor relations between Gantt bars are defined, the resulting structure shall be a sequence of IMLStep and IMLTransition elements, see 8.2.5 and 8.2.6.

8.2.2 Storing the start of a Gantt chart

Table 2 specifies the mapping of properties and relations of the start of a Gantt chart to IML elements.

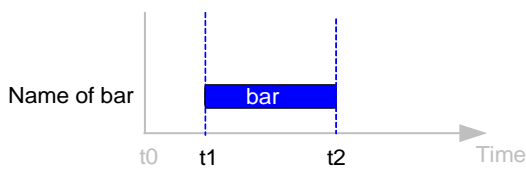
Table 2 – Mapping the start of a Gantt chart to IML element

| Type | Representation |
|---------------------|--|
| Gantt chart element |  |
| IML element | <pre> <xsd:element name="TimeInformation" minOccurs="0"> <xsd:complexType> <xsd:attribute name="timeFormat" type="aml:TimeFormatEnum" use="required" /> <!-- This shall be value of type simple type "TimeFormatEnum" --> <xsd:attribute name="startOfScale" type="aml:TimeUnion" use="optional" default="PT0S"/> <!-- value of t0 in format defined by "timeFormat" attribute --> <xsd:attribute name="endOfScale" type="aml:TimeUnion" use="optional"/> <!-- value of tn in format defined by "timeFormat" attribute --> <xsd:attribute name="scaleInterval" type="aml:TimeUnion" use="optional"/> <!-- value of the time scale in the format defined by xsd:duration type--> </xsd:complexType> </xsd:element> </pre> |

8.2.3 Storing bars

Table 3 specifies the mapping of Gantt chart bars to IML elements.

Table 3 – Mapping of Gantt chart bars to IML elements

| Type | Representation |
|---------------------|--|
| Gantt chart element |  |

| | |
|-------------|--|
| IML element | <pre> <xsd:complexType name="IMLStep"> <xsd:complexContent> <xsd:extension base="ppx:Step"> <xsd:attribute name="uuid" type="aml:UuidString"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="name" type="xsd:string" use="required"/> <!-- "Name of bar" designation in textual representation of type xsd:string --> <xsd:attribute name="startTime" type="aml:TimeUnion"/> <!-- value of t1 in format of type "aml:TimeUnion " --> <xsd:attribute name="endTime" type="aml:TimeUnion"/> <!-- value of t2 in format of type "aml:TimeUnion " --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </pre> |
|-------------|--|

8.2.4 Storing arrows

Table 4 specifies the mapping of Gantt chart bar with one or more successor activities to IML elements.

Table 4 – Mapping of Gantt chart arrow to IML elements

| Type | Representation |
|---------------------|--|
| Gantt chart element |  |
| IML element | <pre> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of the "Name of bar1" element --> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of the "Name of bar2" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

8.2.5 Storing successor bars

Table 5 specifies the mapping of Gantt chart bar with one or more successor bar to IML elements.

Table 5 – Mapping of Gantt chart bar with one or more successor bars to IML elements

| Type | Representation |
|---------------------|---|
| Gantt chart element | <p>The diagram illustrates a Gantt chart with three bars: 'Name of bar1', 'Name of bar3', and 'Name of bar2'. The horizontal axis represents 'Time'. 'Name of bar1' starts at 'bar1_start' and ends at 'bar1_end'. 'Name of bar2' starts at 'bar1_end' and ends at 'bar2_end'. 'Name of bar3' starts at 'bar1_end' and ends at 'bar3_end'. Blue arrows indicate dependencies: one from 'bar1_end' to 'bar2_start' and another from 'bar1_end' to 'bar3_start'.</p> |
| IML element | <pre> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of the "Name of bar1" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLSimultaneousDivergence"> <xsd:complexContent> <xsd:extension base="ppx:IMLSimultaneousDivergence"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of "Name of bar2" element --> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of "Name of bar3" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

8.2.6 Storing predecessor bars

Table 6 specifies the mapping of Gantt chart bar with one or more predecessor bars to IML elements.

Table 6 – Mapping of Gantt chart bar with one and more predecessor bars to IML elements

| Type | Representation |
|---------------------|--|
| Gantt chart element | |
| IML element | <pre> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of the "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "Name of bar3" --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLSimultaneousConvergence"> <xsd:complexContent> <xsd:extension base="ppx:SimultaneousConvergence"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "Name of bar1" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "Name of bar2" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

| Type | Representation |
|------|--|
| | <pre> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

8.3 Storing activity-on-node networks in AML logic XML

8.3.1 Common rules

For storing an activity-on-node network in AML logic XML, the following provisions apply:

- One activity-on-node network shall be mapped to one IML logic element of type "ActivityOnNodeNetwork" using the following provisions.
- The entire activity-on-node network shall be mapped to one IML using the following provisions.
- The IML shall start with the "TimeInformation" element, defining the time format of the activity-on-node network, see 8.3.2.
- Each activity-on-node network node shall be represented by an IMLStep, see 8.3.3.
- Each activity-on-node arrow shall be represented by an IMLTransition, see 8.3.4.
- For predecessor and successor relations in an activity-on-node network the following provisions apply:
 - If an activity-on-node network node has two or more successor nodes, it shall be connected to them by an IMLTransition and IMLSimultaneousDivergence elements, see 8.3.5.
 - If an activity-on-node network node has two or more predecessors, they shall be connected to them by an IMLSimultaneousConvergence and IMLTransition elements, see 8.3.6.
 - If there are no explicit relations between activity-on-node network nodes, i.e. they are concurrent, the order of associated IMLStep shall be parallel and they shall have no ordering relation based on step-step transition sequences, see 8.3.5 and 8.3.6.
 - If predecessor/successor relations between activity-on-node network nodes are defined, the resulting structure shall be a sequence of IMLStep and IMLTransition elements, see 8.3.5 and 8.3.6.

8.3.2 Storing the start of an activity-on-node network

Table 7 specifies the mapping of properties and relations of the start of an activity-on-node network to IML elements.

Table 7 – Mapping of the start of an activity-on-node network to IML elements

| Type | Representation |
|----------------------------------|--|
| Activity-on-node network element | No graphical representation available. |
| IML element | <pre> <xsd:element name="TimeInformation" minOccurs="0"> <xsd:complexType> <xsd:attribute name="timeFormat" type="aml:TimeFormatEnum" use="required" /> <!-- This shall be value of type simple type "TimeFormatEnum" --> </xsd:complexType> </xsd:element> </pre> |

8.3.3 Storing nodes

Table 8 specifies the mapping of activity-on-node network nodes to IML elements.

Table 8 – Mapping of activity-on-node network nodes to IML elements

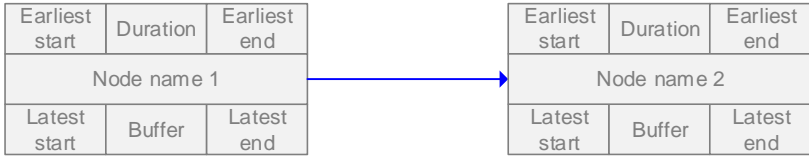
| Type | Representation | | | | | | | | | |
|----------------------------------|--|----------------|----------|--------------|-----------|--|--|--------------|--------|------------|
| Activity-on-node network element | <table><tr><td>Earliest start</td><td>Duration</td><td>Earliest end</td></tr><tr><td colspan="3">Node name</td></tr><tr><td>Latest start</td><td>Buffer</td><td>Latest end</td></tr></table> | Earliest start | Duration | Earliest end | Node name | | | Latest start | Buffer | Latest end |
| Earliest start | Duration | Earliest end | | | | | | | | |
| Node name | | | | | | | | | | |
| Latest start | Buffer | Latest end | | | | | | | | |
| IML element | <pre><xsd:complexType name="IMLStep"> <xsd:complexContent> <xsd:extension base="ppx:Step"> <xsd:attribute name="uuid" type="aml:UuidString"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="name" type="xsd:string use="required"/> <!-- "Name of bar" designation in textual representation of type xsd:string --> <xsd:attribute name="startTime" type="aml:TimeUnion"/> <!-- value of earliest start time in format of type "aml:TimeUnion "--> Having the earliest start time and the earliest end time, the duration can be calculated.--> <xsd:attribute name="endTime" type="aml:TimeUnion"/> <!-- value of earliest end time in format of type "aml:TimeUnion " Having the earliest start time and the earliest end time, the duration can be calculated.--> <xsd:attribute name="buffer" type="aml:duration use="optional"/> <!-- value of buffer in format of type "aml:duration " Having the earliest start time, earliest end time, and the buffer, the latest start time and the latest end time can be calculated.--> </xsd:extension> </xsd:complexContent> </xsd:complexType></pre> | | | | | | | | | |

8.3.4 Storing arrows

Table 9 specifies the mapping of activity-on-node network arrows to IML elements.

NOTE Both nodes are are not mapped to IML elements, only the arrow, which connects both.

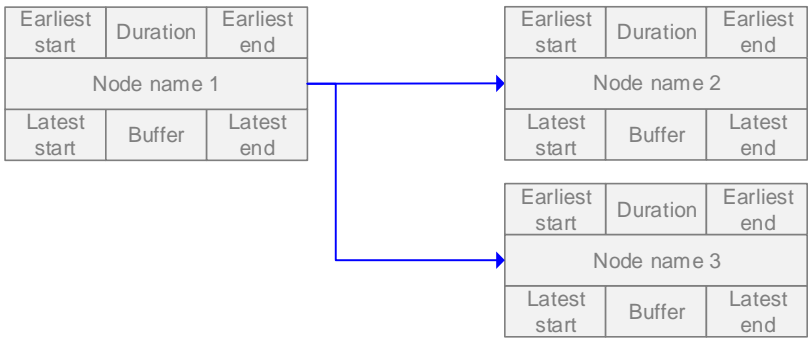
Table 9 – Mapping of activity-on-node network arrows to IML elements

| Type | Representation |
|----------------------------------|--|
| Activity-on-node network element |  |
| IML element | <pre> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of the "Name of bar1" element --> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of the "Name of bar2" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

8.3.5 Storing successor nodes

Table 10 specifies the mapping of activity-on-node network successor activities to IML elements.

Table 10 – Mapping of activity-on-node network successor nodes to IML elements

| Type | Representation |
|----------------------------------|--|
| Activity-on-node network element |  |
| IML element | <pre> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> </pre> |

| Type | Representation |
|------|---|
| | <pre> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of the "Name of bar1" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLSimultaneousDivergence"> <xsd:complexContent> <xsd:extension base="ppx: IMLSimultaneousDivergence"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the inward connection point of "Name of bar2" element --> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the inward connection point of "Name of bar3" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

8.3.6 Storing predecessor nodes

Table 11 specifies the mapping of activity-on-node network predecessor nodes to IML elements.

Table 11 – Mapping of activity-on-node network predecessor nodes to IML elements

| Type | Representation |
|----------------------------------|--|
| Activity-on-node network element | <p>The diagram illustrates the mapping of activity-on-node network elements to IML elements. It shows three nodes: Node name 1, Node name 2, and Node name 3. Each node is represented by a table with columns: Earliest start, Duration, Earliest end, Latest start, Buffer, and Latest end. Node name 1 and Node name 2 are connected to Node name 3 by a blue arrow.</p> |
| IML element | <pre> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the inward connection point of the "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "Name of bar3" --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLSimultaneousConvergence"> <xsd:complexContent> <xsd:extension base="ppx:SimultaneousConvergence"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "Name of bar1" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "Name of bar2" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

| Type | Representation |
|------|--|
| | <pre><xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of "IMLTransition" element --> </xsd:complexType> </xsd:element> </xsd:complexType></pre> |

8.4 Storing timing diagrams in AML logic XML

8.4.1 Common rules

For storing a timing diagram in AML logic XML, the following provisions apply:


- One timing diagram shall be mapped to one IML logic element of type "TimingDiagram" using the following provisions.
- The IML shall start with the timeline definition, see 8.4.2.
- The Resources and their States within a timing diagram shall be represented through IML elements, see 8.4.3.
- The lifeline for each Resource shall be defined through its segments as IMLStep elements coupled with IMLTransition elements, see 8.4.4.
- The signals in a timing diagram shall be defined through IMLTransition elements, see 8.4.5.

NOTE Signals can be fired by the timeline at any time or by a resource after remaining within a resource state with a predefined duration.

8.4.2 Storing the timeline of a timing diagram

Table 12 specifies the mapping of the timeline of a timing diagram to IML elements.

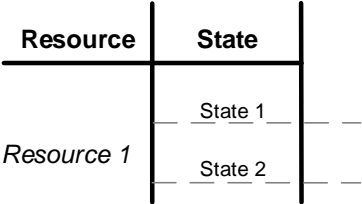
Table 12 – Mapping of the timeline of a timing diagram to IML elements

| Type | Representation |
|------------------------|---|
| Timing diagram element |  |
| IML element | <pre><xsd:element name="TimeInformation" minOccurs="0"> <xsd:complexType> <xsd:attribute name="timeFormat" type="aml:TimeFormatEnum" use="required" /> <!-- This shall be value of type simple type "TimeFormatEnum" --> <xsd:attribute name="startOfScale" type="aml:TimeUnion" use="optional" default="PT0S"/> <!-- value of t0 in format defined by "timeFormat" attribute --> <xsd:attribute name="endOfScale" type="aml:TimeUnion" use="optional"/> <!-- value of tn in format defined by "timeFormat" attribute --> <xsd:attribute name="scaleInterval" type="aml:TimeUnion" use="optional"/> <!-- value of the time scale in the format defined by xsd:duration type --> </xsd:complexType> </xsd:element></pre> |

8.4.3 Storing resources and resource states

Table 13 specifies the mapping of timing diagram resources and resource states to IML elements.

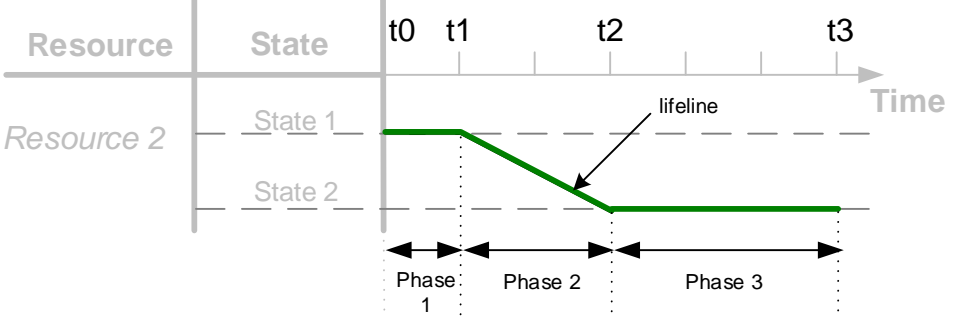
Table 13 – Mapping of timing diagram resources and resource states to IML elements

| Type | Representation |
|------------------------|--|
| Timing diagram element |  |
| IML element | <pre> <xsd:element name="Resource" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:sequence> <xsd:element name="Status" maxOccurs="unbounded"> <xsd:complexType> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="name" type="xsd:string" use="required"/> <!-- Name of state in textual representation of type xsd:string --> <xsd:attribute name="order" type="xsd:integer" use="optional"/> <!-- Order of state in xsd:integer representation --> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </xsd:element> </pre> |

8.4.4 Storing lifelines

Table 14 specifies the mapping of timing diagram lifelines to IML elements.

Table 14 – Mapping of timing diagram lifelines to IML elements

| Type | Representation |
|------------------------|--|
| Timing diagram element |  |
| IML element | <pre> <xsd:complexType name="IMLStep"> <xsd:complexContent> <xsd:extension base="ppx:Step"> <xsd:attribute name="uuid" type="aml:UuidString"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="name" type="xsd:string" use="required"/> <!-- "Phase 1" segment in textual representation of type xsd:string --> <xsd:attribute name="startTime" type="aml:TimeUnion"/> <!-- value of t0 in format of type "aml:TimeUnion" --> <xsd:attribute name="endTime" type="aml:TimeUnion"/> <!-- value of t1 in format of type "aml:TimeUnion" --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </pre> |

| Type | Representation |
|------|--|
| | <pre> </xsd:complexContent> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "Phase 1" --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLStep"> <xsd:complexContent> <xsd:extension base="ppx:Step"> <xsd:attribute name="uuid" type="aml:UuidString"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="name" type="xsd:string" use="required"/> <!-- "Phase 2" segment in textual representation of type xsd:string --> <xsd:attribute name="startTime" type="aml:TimeUnion"/> <!-- value of t1 in format of type "aml:TimeUnion " --> <xsd:attribute name="endTime" type="aml:TimeUnion"/> <!-- value of t2 in format of type "aml:TimeUnion " --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of "Phase 2" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "Phase 2" --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLStep"> <xsd:complexContent> <xsd:extension base="ppx:Step"> <xsd:attribute name="uuid" type="aml:UuidString"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="name" type="xsd:string" use="required"/> <!-- "Phase 3" segment in textual representation of type xsd:string --> <xsd:attribute name="startTime" type="aml:TimeUnion"/> <!-- value of t2 in format of type "aml:TimeUnion " --> <xsd:attribute name="endTime" type="aml:TimeUnion"/> <!-- value of t3 in format of type "aml:TimeUnion " --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of "Phase 3" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> </pre> |

| Type | Representation |
|------|---|
| | <pre> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "Phase 1" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the inward connection point of "Phase 2" --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "Phase 2" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the inward connection point of "Phase 3" --> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

8.4.5 Storing the time signal and the resource signal

Table 15 specifies the mapping of the time signal and the resource signal to IML elements.

Table 15 – Mapping of the time signal and the resource signal to IML elements

| Type | Representation |
|------------------------|--|
| Timing diagram element | <p>The diagram illustrates the mapping of time and resource signals to IML elements. It shows two resources, Resource 1 and Resource 2, over time. Resource 1 has three states (State 1, State 2, State 3) and Resource 2 has two states (State 1, State 2). The time axis is marked with t0, t1, t2, and t3. The diagram is divided into three phases: Phase 1 (t0 to t1), Phase 2 (t1 to t2), and Phase 3 (t2 to t3). A 'Time signal' is shown as a step function, and a 'Resource signal' is shown as a piecewise linear function.</p> |
| IML element | <pre> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="refTime" type="aml:TimeUnion" use="optional"/> <!-- value of t2 in format of type "aml:TimeUnion" --> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the outward connection point of "Phase 2 Resource 1" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!-- Global ID of the inward connection point of "Phase 3 Resource 1" --> </xsd:complexType> </xsd:element> </xsd:complexType> <xsd:complexType name="IMLTransition"> <xsd:complexContent> <xsd:extension base="ppx:Transition"> <xsd:attribute name="uuid" type="aml:UuidString" use="required"/> <!-- This shall be some textual representation of UUID, which can be GUID. --> <xsd:attribute name="refEndSegmentId" type="aml:UuidString" use="optional"/> <!-- This shall be some textual representation of UUID, which can be GUID, of the IMLStep for "Phase 1 Resource 1" segment--> </xsd:extension> </xsd:complexContent> <xsd:element name="ConnectionPointIn" type="ppx:connectionPointIn" minOccurs="0" maxOccurs="1"> <xsd:complexType> </pre> |

| Type | Representation |
|------|---|
| | <pre> <xsd:element name="Connection" type="ppx:connection" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the outward connection point of "Phase 1 Resource 2" --> </xsd:complexType> </xsd:element> </xsd:complexType> </xsd:element> <xsd:element name="ConnectionPointOut" type="ppx:connectionPointOut" minOccurs="0" maxOccurs="1"> <xsd:complexType> <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/> <!--Global ID of the inward connection point of "Phase 2 Resource 2" --> </xsd:complexType> </xsd:element> </xsd:complexType> </pre> |

8.5 Storing sequential function charts in AML logic XML

8.5.1 Common rules

For storing a sequential function chart in AML logic XML, the following provisions apply:

- The entire sequential function chart shall be mapped to one "FunctionBlock" element of the type "SFC" within an "AMLLogic" element.
- The elements of a sequential function chart shall be used in accordance with IEC 61131-10.

8.5.2 Storing variables

Table 16 specifies the storage of variables of a sequential function chart in AML logic XML.

Table 16 – Storing variable of a sequential function chart in AML logic XML

| Type | Representation |
|-----------------------------------|---|
| Sequential function chart element | No graphical representation available. |
| AML logic XML element | <pre> <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0" maxOccurs="1"/> <xsd:complexType name="ParameterSet"> <xsd:sequence> <xsd:choice minOccurs="0" maxOccurs="unbounded"> <xsd:element name="InputVars"> <xsd:complexType> <xsd:sequence> <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:complexContent> <xsd:extension base="ppx:VariableDecl"> <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/> <!-- This shall indicate the order of the parameters. --> <xsd:attribute name="edgeDetection" type="ppx:EdgeModifierType" use="optional" default="none"/> <!--This parameter can be characterized regarding edge detection. --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:choice> <xsd:element name="OutputVars"> <xsd:complexType> <xsd:sequence> <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:complexContent> <xsd:extension base="ppx:VariableDecl"> <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/> <!-- This shall indicate the order of the parameters. --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </pre> |

8.6 Storing function block diagrams in AML logic XML

8.6.1 Common rules

For storing a function block diagram in AML logic XML, the following provisions apply:

- The entire function block diagram shall be mapped to one "FunctionBlock" element of the type "FBD" within an "AMLLogic" element.
- The elements of a function block diagram shall be used in accordance with IEC 61131-10.

8.6.2 Storing variables

Table 17 specifies the storage of variables of a function block diagram in AML logic XML.

Table 17 – Storing variable of a function block diagram in AML logic XML

| Type | Representation |
|--------------------------------|---|
| Function block diagram element | No graphical representation available. |
| AML logic XML element | <pre> <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0" maxOccurs="1"/> <xsd:complexType name="ParameterSet"> <xsd:sequence> <xsd:choice minOccurs="0" maxOccurs="unbounded"> <xsd:element name="InputVars"> <xsd:complexType> <xsd:sequence> <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:complexContent> <xsd:extension base="ppx:VariableDecl"> <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/> <!-- This shall indicate the order of the parameters. --> <xsd:attribute name="edgeDetection" type="ppx:EdgeModifierType" use="optional" default="none"/> <!--This parameter can be characterized regarding edge detection. --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </xsd:element> <xsd:element name="OutputVars"> <xsd:complexType> <xsd:sequence> <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:complexContent> <xsd:extension base="ppx:VariableDecl"> <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/> <!-- This shall indicate the order of the parameters. --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:choice> </xsd:sequence> </xsd:complexType> </pre> |

8.7 Storing mathematical expressions in AML logic XML

8.7.1 Common rules

For storing a mathematical expression AML logic XML, the following provisions apply:

- The entire mathematical expression shall be mapped to one "FunctionBlock" element of the type "MathematicalExpression" within an "AMLLogic" element.
- Variables of the "FunctionBlock" element shall be mapped to the elements "InputVars" or "OutputVars" within the "Parameters" element accordingly.
- The mathematical expression shall be stored in the "MathML" element in accordance with MathML 2.0 content part.
- How the variables of the "MathML" element correlate with the variables of the "FunctionBlock" element shall be modelled in the "VariableMapping" element.
 - The "VariableMapping" element shall have the following attributes:

- The attribute "refMathMLVariableName" shall store the name of the variable in the "MathML" element.
- The attribute "refFBVariableUUID" shall store the UUID of the variable of the "FunctionBlock" element.
- The attribute "direction" shall indicate whether it is an input variable or an output variable for the mathematical expression stored in the "MathML" element. The value "In" shall indicate an input variable. The value "Out" shall indicate an output variable.

8.7.2 Storing variables

Table 18 specifies the storage of variables of a mathematical expression in AML logic XML.

Table 18 – Storing variable of a mathematical expression in AML logic XML

| Type | Representation |
|---------------------------------|---|
| Mathematical expression element | No graphical representation available. |
| AML logic XML element | <pre> <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0" maxOccurs="1"/> <xsd:complexType name="ParameterSet"> <xsd:sequence> <xsd:choice minOccurs="0" maxOccurs="unbounded"> <xsd:element name="InputVars"> <xsd:complexType> <xsd:sequence> <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:complexContent> <xsd:extension base="ppx:VariableDecl"> <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/> <!-- This shall indicate the order of the parameters. --> <xsd:attribute name="edgeDetection" type="ppx:EdgeModifierType" use="optional" default="none"/> <!--This parameter can be characterized regarding edge detection. --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:choice> <xsd:element name="OutputVars"> <xsd:complexType> <xsd:sequence> <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:complexContent> <xsd:extension base="ppx:VariableDecl"> <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/> <!-- This shall indicate the order of the parameters. --> </xsd:extension> </xsd:complexContent> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:complexType> </pre> |

8.7.3 Storing variable mappings

Table 19 specifies the storage of variable mappings of a mathematical expression in AML logic XML.

Table 19 – Storing variable mappings of a mathematical expression in AML logic XML

| Type | Representation |
|---------------------------------|--|
| Mathematical expression element | No graphical representation available. |
| AML logic XML element | <pre> <xsd:element name="VariableMapping" minOccurs="0" maxOccurs="unbounded"> <xsd:complexType> <xsd:attribute name="refMathMLVariableName" type="xsd:string" use="required"/> <!--It shall represent the name of the variable defined in MathML.--> <xsd:attribute name="refFBVariableUUID" type="aml:UuidString" use="required"/> <!--It shall represent UUID of the variable declared in the same function block.--> <xsd:attribute name="direction" use="required"> <xsd:simpleType> <xsd:restriction base="xsd:string"> <xsd:enumeration value="In"/> <xsd:enumeration value="Out"/> </xsd:restriction> </xsd:simpleType> </xsd:attribute> <!--It shall represent mapping direction with enumerated string which only allows "In" or "Out". --> </xsd:complexType> </xsd:element> </pre> |

8.7.4 Storing mathematical expressions

Table 20 specifies the storage of a mathematical expression in AML logic XML.

Table 20 – Storing a mathematical expression in AML logic XML

| Type | Representation |
|---------------------------------|--|
| Mathematical expression element | Mathematical expression |
| AML logic XML element | <pre> <xsd:element name="MathML"> <xsd:complexType> <xsd:sequence> <xsd:any namespace="##any" processContents="lax"/> </xsd:sequence> </xsd:complexType> </xsd:element> </pre> |

9 Meta information about AML logic XML writer tools

In order to simplify the data exchange between a source tool and a destination tool, it is useful to store information about the writer tools directly into the AML logic XML document. Hence, the following provisions apply:

NOTE In the engineering process it is useful to transfer the information of the project the data belongs to and the time of the export of the data. This supports the implementation process of interfaces.

- An AML logic XML document shall provide information about each writer tool that has written the AML logic XML document.
- In a data exchange tool chain, the last participating writer tools shall store this information in the AML logic XML document.
- Each AML logic XML document shall have one or more XML elements "WriterHeader".

- In a data exchange tool chain, all participating tools shall store this information in the AML logic XML document in the same way. Hence, the document may contain information about multiple tools of a data exchange tool chain. A tool may remove the writer information of other tools. This may hinder the iterative data exchange with the other tools: hence the removal of writer information of other tools is not recommended.
- This information shall be stored within XML elements of the XML element "WriterHeader".
- The meta information shall provide information about:
 - name of the writer tool;
 - unique ID of the writer tool (e.g. UUID, URL of the tool);
 - name of the writer tool vendor;
 - URL of the writer tool vendor;
 - complete version information of the writer tool (including major, minor and build revision); and
 - last writing date and time of the AML logic XML document.
- Additionally, the following optional information may be included:
 - cultural settings of the writer tool following RFC 5646;
 - name of the project associated with exported data;
 - unique ID of the project associated with exported data; and
 - additional information.
- The above information shall be stored by means of the XML elements shown in Table 21 for each writer tool.

Table 21 – Meta information about each AML logic XML writer tool

| XML element name | Type | Cardinality | Example |
|-----------------------|-------------|-------------|---|
| WriterName | xs:string | 1 | "ToolX" |
| WriterID | xs:string | 1 | "4F1D5716-1C2A-4CBC-8C40-91A91F7E09A4" |
| WriterVendor | xs:string | 1 | "ToolX Vendor" |
| WriterVendorURL | xs:string | 1 | "http://www.ToolX-Vendor.org" |
| WriterVersion | xs:string | 1 | "0.2.123 prealpha" |
| LastWritingDateTime | xs:dateTime | 1 | "2011-05-25T09:30:47" |
| WriterCulturalSetting | xs:string | 0..1 | "en-US" |
| WriterProjectTitle | xs:string | 0..1 | "ProjectY" |
| WriterProjectID | xs:string | 0..1 | "6BA300EA-DF77-4E78-80F9-2B050E2C6799" |
| AdditionalInformation | xs:string | 0..1 | "This tool vendor was previously called ToolYVendor." |

10 Extensions of AML classes for logic

10.1 General

This clause defines extensions of the AML role classes and AML interface classes. These classes are either a part of own libraries or are specializations of AML standard classes defined in IEC 62714-1. All described attributes and interfaces are part of these libraries and may be removed in the InstanceHierarchy if not needed.

10.2 AutomationMLLogicRoleClassLib

10.2.1 General

Figure 23 presents the normative AutomationMLLogicRoleClassLib as object tree. AutomationMLLogicRoleClassLib specifies all logic related role classes. Details to each role class are given in 10.2.2 to 10.2.4. The XML text of AutomationMLLogicRoleClassLib is given in Clause D.2.

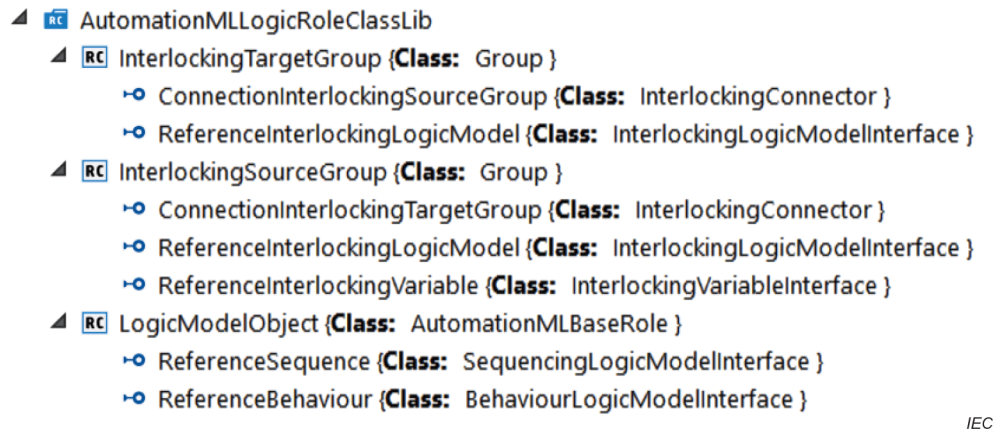


Figure 23 – AutomationMLLogicRoleClassLib

10.2.2 RoleClass InterlockingTargetGroup

The role class "InterlockingTargetGroup" shall be used as specified in Table 22.

Table 22 – RoleClass InterlockingTargetGroup

| | | |
|-----------------------------------|--|--|
| Class name | InterlockingTargetGroup | |
| Description | The role class "InterlockingTargetGroup" shall be used for interlocking target groups. Details are specified in Clause 12. For examples see Annex C. | |
| Parent class | AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group | |
| Path for element reference | AutomationMLLogicRoleClassLib/InterlockingTargetGroup | |
| Attributes | none | |
| Interfaces | Interface of type="InterlockingConnector" | The interfaces shall be of type "InterlockingConnector" specified in 10.5.2. This interface shall be required for the interlocking concept specified in Clause 12. The name is user-defined. The use of the interface shall be mandatory. |
| | Interface of type="InterlockingLogicModelInterface" | The interfaces shall be of type "InterlockingLogicModelInterface" specified in 10.3.5. This interface shall be used for the interlocking concept specified in Clause 12 for referencing the interlocking logic model. The name is user-defined. The use of the interface shall be optional. |

10.2.3 RoleClass InterlockingSourceGroup

The role class "InterlockingSourceGroup" shall be used as specified in Table 23.

Table 23 – RoleClass InterlockingSourceGroup

| | | |
|-----------------------------------|--|--|
| Class name | InterlockingSourceGroup | |
| Description | The role class "InterlockingSourceGroup" shall be used for interlocking source groups. Details are specified in Clause 12. For examples see Annex C. | |
| Parent class | AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group | |
| Path for element reference | AutomationMLLogicRoleClassLib/InterlockingSourceGroup | |
| Attributes | none | |
| Interfaces | Interface of type="InterlockingConnector" | The interfaces shall be of type "InterlockingConnector" specified in 10.5.2. This interface shall be required for the interlocking concept specified in Clause 12. The name is user-defined. The use of the interface shall be mandatory. |
| | Interface of type="InterlockingLogicModelInterface" | The interfaces shall be of type "InterlockingLogicModelInterface" specified in 10.3.5. This interface shall be used for the interlocking concept specified in Clause 12 for referencing the interlocking logic model. The name is user-defined. The use of the interface shall be optional. |
| | Interface of type="InterlockingVariableInterface" | The interfaces shall be of type "InterlockingVariableInterface" specified in 10.3.8. This interface shall be used for the interlocking concept specified in Clause 12 for referencing the output variable in a logic model containing interlocking information. The name is user-defined. The use of the interface shall be optional. |

10.2.4 RoleClass LogicModelObject

The role class "LogicModelObject" shall be used as specified in Table 24.

Table 24 – RoleClass LogicModelObject

| | | |
|-----------------------------------|--|--|
| Class name | LogicModelObject | |
| Description | The role class "LogicModelObject" shall denote an AML logic object, instantiating a logic model. | |
| Parent class | AutomationMLBaseRoleClassLib/AutomationMLBaseRole | |
| Path for element reference | AutomationMLLogicRoleClassLib/LogicModelObject | |
| Attributes | none | |
| Interfaces | Interface of type="SequencingLogicModelInterface" | The interfaces shall be of type "SequencingLogicModelInterface" specified in 10.3.3. This interface shall be used to specify the logic model semantics specified in 11. The name is user-defined. The use of the interface shall be optional. |
| | Interface of type="BehaviourLogicModelInterface" | The interfaces shall be of type "BehaviourLogicModelInterface" specified in 10.3.4. This interface shall be used to specify the logic model semantics specified in 11. The name is user-defined. The use of the interface shall be optional. |

NOTE These objects can be enriched and further referenced in CAEX.

10.3 AutomationMLLogicInterfaceClassLib

10.3.1 General

Figure 24 presents the normative AutomationMLLogicInterfaceClassLib as object tree. AutomationMLLogicInterfaceClassLib specifies all logic related interface classes for referencing AML logic XML documents. Details to each interface class are given in 10.3.2 to 10.3.8. The XML text of AutomationMLInterfaceClassLib is given in Clause D.3.

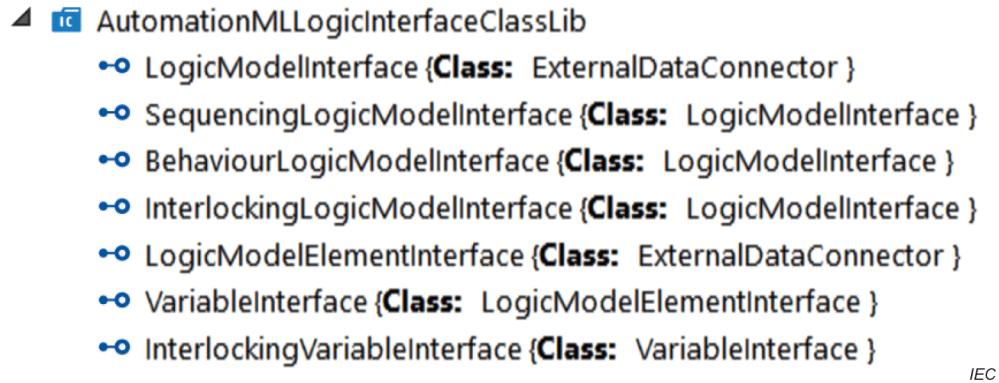


Figure 24 – AutomationMLLogicInterfaceClassLib

10.3.2 InterfaceClass LogicModelInterface

The interface class "LogicModelInterface" shall be used as specified in Table 25.

Table 25 – InterfaceClass LogicModelInterface

| | |
|-----------------------------------|---|
| Class name | LogicModelInterface |
| Description | All derivations of the interface class "LogicModelInterface" shall be used in order to reference a logic model stored in a FunctionBlock within an external AML logic XML document. Details are specified in 11. For examples, see Annex B. |
| Parent class | AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector |
| Path for element reference | AutomationMLLogicInterfaceClassLib/LogicModelInterface |
| Attributes | none |

10.3.3 InterfaceClass SequencingLogicModelInterface

The interface class "SequencingLogicModelInterface" shall be used as specified in Table 26.

Table 26 – InterfaceClass SequencingLogicModelInterface

| | |
|-----------------------------------|--|
| Class name | SequencingLogicModelInterface |
| Description | The interface class "SequencingLogicModelInterface" shall be used in order to reference a logic model containing sequencing information stored in a FunctionBlock within an external AML logic XML document. Details are specified in 11. For examples, see Annex B. |
| Parent class | AutomationMLLogicInterfaceClassLib/LogicModelInterface |
| Path for element reference | AutomationMLLogicInterfaceClassLib/SequencingLogicModelInterface |
| Attributes | none |

10.3.4 InterfaceClass BehaviourLogicModelInterface

The interface class "BehaviourLogicModelInterface" shall be used as specified in Table 27.

Table 27 – InterfaceClass BehaviourLogicModelInterface

| | |
|-----------------------------------|--|
| Class name | BehaviourLogicModelInterface |
| Description | The interface class "BehaviourLogicModelInterface" shall be used in order to reference a logic model containing behaviour information stored in a FunctionBlock within an external AML logic XML document. Details are specified in 11. For examples, see Annex B. |
| Parent class | AutomationMLLogicInterfaceClassLib/LogicModelInterface |
| Path for element reference | AutomationMLLogicInterfaceClassLib/BehaviourLogicModelInterface |
| Attributes | none |

10.3.5 InterfaceClass InterlockingLogicModelInterface

The interface class "InterlockingLogicModelInterface" shall be used as specified in Table 28.

Table 28 – InterfaceClass InterlockingLogicModelInterface

| | |
|-----------------------------------|---|
| Class name | InterlockingLogicModelInterface |
| Description | The interface class "InterlockingLogicModelInterface" shall be used in order to reference a logic model containing interlocking information stored in a FunctionBlock within an external AML logic XML document. This interface class shall only be used by interlocking source groups and interlocking target groups. Details are specified in Clause 12. For examples, see Annex C. |
| Parent class | AutomationMLLogicInterfaceClassLib/LogicModelInterface |
| Path for element reference | AutomationMLLogicInterfaceClassLib/InterlockingLogicModelInterface |
| Attributes | none |

10.3.6 InterfaceClass LogicModelElementInterface

The interface class "LogicModelElementInterface" shall be used as specified in Table 29.

Table 29 – InterfaceClass LogicModelElementInterface

| | |
|-----------------------------------|---|
| Class name | LogicModelElementInterface |
| Description | The interface class "LogicModelElementInterface" shall be used to reference an element in a logic model stored in a FunctionBlock within an external AML logic XML document. Details are specified in Clause 11. For examples, see Annex B. |
| Parent class | AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector |
| Path for element reference | AutomationMLLogicInterfaceClassLib/LogicModelElementInterface |
| Attributes | none |

10.3.7 InterfaceClass VariableInterface

The interface class "VariableInterface" shall be used as specified in Table 30.

Table 30 – InterfaceClass VariableInterface

| | | |
|-----------------------------------|---|---|
| Class name | VariableInterface | |
| Description | The interface class "VariableInterface" shall be used to reference a variable in a logic model stored in a FunctionBlock within an external AML logic XML document. Details are specified in Clause 11. For examples see Annex B. | |
| Parent class | AutomationMLLogicInterfaceClassLib/LogicModelElementInterface | |
| Path for element reference | AutomationMLLogicInterfaceClassLib/VariableInterface | |
| Attributes | Direction (type="xs:string") | <p>This attribute shall be used to describe a direction of the VariableInterface. Allowed values: "In", "Out", or "InOut".</p> <p>The attribute value shall be "In" if the variable is consumed by CAEX. The attribute value shall be "Out" if the information flow is from CAEX to the variable.</p> <p>The attribute value shall be "InOut" if both directions are possible.</p> <p>The use of the attribute shall be optional.</p> |

10.3.8 InterfaceClass InterlockingVariableInterface

The interface class "InterlockingVariableInterface" shall be used as specified in Table 31.

Table 31 – InterfaceClass InterlockingVariableInterface

| | | |
|-----------------------------------|--|---|
| Class name | InterlockingVariableInterface | |
| Description | The interface class "InterlockingVariableInterface" shall be used to reference the output variable in a logic model containing interlocking information stored in a FunctionBlock within an external AML logic XML document. This variable represents the result of a function describing the interlocking condition. This interface class shall only be used by interlocking source groups and interlocking target groups. Details are specified in Clause 12. For examples, see Annex C. | |
| Parent class | AutomationMLLogicInterfaceClassLib/VariableInterface | |
| Path for element reference | AutomationMLLogicInterfaceClassLib/InterlockingVariableInterface | |
| Attributes | SafeConditionEquals (type="xs:boolean") | <p>The attribute "SafeConditionEquals" shall be used in order to specify the result of the function describing the interlocking condition. The allowed values shall be "true" or "false" and shall indicate which value of a unique Boolean variable represents the safe state of an interlocking source group.</p> <p>The use of the attribute shall be optional. The default value shall be "true".</p> |

10.4 AutomationMLPLCOpenXMLInterfaceClassLib

10.4.1 General

Figure 25 presents the normative AutomationMLPLCOpenXMLInterfaceClassLib as object tree. AutomationMLPLCOpenXMLInterfaceClassLib specifies the logic related interface class for referencing IEC 61131-10 documents. Details to the interface class are given in 10.4.2. The XML text of AutomationMLInterfaceClassLib is given in Clause D.4.


AutomationMLPLCopenXMLInterfaceClassLib
 VariableInterface {Class: PLCopenXMLInterface }
IEC

Figure 25 – AutomationMLPLCopenXMLInterfaceClassLib

10.4.2 InterfaceClass VariableInterface

The interface class "VariableInterface" shall be used as specified in Table 32.

Table 32 – InterfaceClass VariableInterface

| | |
|-----------------------------------|--|
| Class name | VariableInterface |
| Description | The interface class "VariableInterface" shall be used in order to publish signals or variables that are defined inside of a PLCopen XML document of the PLCopenXML version 2.0, 2.01, or IEC 61131-10. |
| Parent class | AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/PLCopenXMLInterface |
| Path for element reference | AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/PLCopenXMLInterface/VariableInterface |
| Attributes | none |

10.5 AutomationMLInterfaceClassLib

10.5.1 General

Subclause 10.5 defines specifications for AML standard interface classes. These classes are part of the AML standard library. All described attributes and interfaces are part of the standard library and may be removed in the InstanceHierarchy if not needed. Details to the interface classes are given in 10.5.2 and 10.5.3. The XML text of AutomationMLInterfaceClassLib is given in IEC 62714-1.

10.5.2 InterfaceClass InterlockingConnector

The interface class "InterlockingConnector" is a class of the AML standard interface class library, defined in IEC 62714-1, and shall be used as specified in Table 33.

Table 33 – InterfaceClass InterlockingConnector

| | |
|-----------------------------------|--|
| Class name | InterlockingConnector |
| Description | The interface class "InterlockingConnector" shall be used in order to model relations between an interlocking source group and an interlocking target group. Details are specified in Clause 12. For examples see Annex C. |
| Parent class | AutomationMLInterfaceClassLib/AutomationMLBaseInterface |
| Path for element reference | AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector |
| Attributes | none |

10.5.3 InterfaceClass PLCopenXMLInterface

The interface class "PLCopenXMLInterface" is a class of the AML standard interface class library, defined in IEC 62714-1, and shall be used as specified in Table 34.

Table 34 – InterfaceClass PLCopenXMLInterface

| | |
|-----------------------------------|--|
| Class name | PLCopenXMLInterface |
| Description | The interface class "PLCopenXMLInterface" shall be used in order to reference external PLCopen XML documents of the PLCopenXML version 2.0, 2.01, or IEC 61131-10. |
| Parent class | AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector |
| Path for element reference | AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/PLCopenXMLInterface |
| Attributes | none |

11 Referencing AML logic XML documents

11.1 General

According to the distributed document structure of AML, it is necessary to put the different information within the different documents in relation to each other. This clause focuses on the referencing of AML objects (within a CAEX document) with logic information stored in AML logic XML documents. An informative overview of the referencing methods is provided in Annex B.

11.2 Referencing logic information

Regarding referencing logic information, the following provisions apply:

- A reference from an AML object to logic information within an AML logic XML document shall be modelled by means of a CAEX ExternalInterface using the AML InterfaceClasses "LogicModelInterface", "LogicModelElementInterface" and their derivations, specified in 10.3.
 - Logic information shall be referenced by its URI within the attribute "refURI" of these CAEX ExternalInterfaces.
 - The value of the attribute "refURI" shall point to the logic information within the AML logic XML document and shall follow the URI syntax, using the AML logic XML attribute "uuid" as fragment.

NOTE 1 Example for the value of the refURI: file:///behaviour.xml#74e94dc8-b5a2-490c-bf0f-f0fdaa4515ef

NOTE 2 In this part of IEC 62714, UUIDs are presented in a short form such as "UUID1", "UUID100" etc. This serves the readability and acts as a real UUID.

NOTE 3 It is not intended to just reference the AML logic XML document itself, since the document might contain several logic models containing logic information.

- Every "LogicModelElementInterface" and its derivations shall refer to one corresponding derivation of "LogicModelInterface" via an InternalLink.

NOTE 4 A derivation of "LogicModelInterface" can be connected to more than one "LogicModelElementInterface" and its derivations via an InternalLink.

NOTE 5 The variables can, then, be assigned, e.g. to signals of other AML objects or to variables of other logic models, by using InternalLinks.

- Logic models shall be distinguishable, i.e. an AML logic object shall not have more than one CAEX ExternalInterface of the same AML InterfaceClass "SequencingLogicModelInterface", "BehaviourLogicModelInterface" or a derivation of them. Every interlocking source group or interlocking target group shall only have one CAEX ExternalInterface of the AML InterfaceClass "InterlockingLogicModelInterface".

NOTE 6 For example, different levels of detail of one logic model are not allowed at one AML logic object.

12 Linking AML objects with interlocking information

12.1 General

Interlocking information is an important part of the engineering of production systems to ensure their safe behaviour. This clause focuses on the interlocking concept which comprises two levels of detail to model and store interlocking information in AML. An informative overview of interlocking concept is provided in Annex C.

NOTE In this part of IEC 62714, interlocking addresses safety relevant dependencies between equipment, e.g. light barriers which prevent humans from getting harmed by a robot. In this part of IEC 62714, interlocking does not address the functional relevant dependencies between equipment, e.g. that one of two collaborating robots needs to wait until it had finished its operation.

12.2 Referencing interlocking information

On the first level of detail of the interlocking concept, interlocking source groups and interlocking target groups are linked to describe functional dependencies among AML object groups respectively AML objects. For this, the following provisions apply:

- An interlocking target group shall be created by using the AML Group concept according to IEC 62714-1.
 - For identifying the group as an interlocking target group the role class "InterlockingTargetGroup" shall be used, which is described in 10.2.2.

NOTE 1 The objects of an interlocking target group execute the actions which are necessary to ensure functional safety.

- An interlocking source group shall be created by using the AML Group concept according to IEC 62714-1.
 - For identifying the group as an interlocking source group the role class "InterlockingSourceGroup" shall be used, which is described in 10.2.3.

NOTE 2 The objects of an interlocking source group indicate when actions are necessary to provide functional safety.

- Each interlocking target group and each interlocking source group shall have the CAEX ExternalInterface of the AML InterfaceClass "InterlockingConnector", which is described in 10.5.2.
 - The CAEX ExternalInterface of the interlocking source group shall be linked to the CAEX ExternalInterface of the corresponding interlocking target group via an InternalLink.
- An AML object shall be in zero or more interlocking groups.
- An AML object shall be allowed to be in an interlocking source group as well as in an interlocking target group at the same time.
- One or more interlocking source groups shall be allowed to be connected to one or more interlocking target groups (i.e. the assignment between both groups is not restricted to a 1:1 relation)
- The relations between the different groups shall be independent.

NOTE 3 A state of the interlocking source group affects the interlocking target group.

On the second level of detail of the interlocking concept, the simple grouping of AML objects into interlocking source groups and interlocking target groups (of the first level of detail) is extended by a function describing the interlocking condition of interlocking source groups. For this, the following provisions apply:

- The function shall be represented by a FunctionBlock of the type "FBD", which is described in Clause 7.
- This FunctionBlock shall be referenced by the CAEX ExternalInterface of the AML interface class "InterlockingLogicModelInterface", as specified in 10.3.5.
- The FunctionBlock shall result in a unique Boolean variable describing the output respectively evaluation result of this FunctionBlock.

NOTE 4 This indicates whether the safe state of the interlocking source group is represented by a "true" or "false" value.

- The unique Boolean variable shall be referenced by an interlocking source group modelling one CAEX ExternalInterface of the AML InterfaceClass "InterlockingVariableInterface", as specified in 10.3.8.
 - This CAEX ExternalInterface may have an attribute "SafeConditionEquals", which shall store the Boolean value indicating the safe state. The default value is "true".
- Following the mechanism described in 11, the ExternalInterfaces of the input and output variables of the FunctionBlock shall be connected with the ExternalInterface of the interlocking logic model by InternalLinks.
 - The modelling of the reaction of the corresponding interlocking target groups shall follow the same provisions as defined for the interlocking source group.

Annex A

(informative)

Examples for storing logic models in AML logic XML

A.1 Example for storing Gantt charts

A.1.1 General

The examples given in Table A.1 to Table A.4 show the usage of the mapping rules to transform Gantt charts to IML.

A.1.2 Storing of activities without predecessor and successor relation

The Gantt chart example in Table A.1 has no predecessor and successor relations among the bars.

Table A.1 – Storing of the Gantt chart example "activities without predecessor and successor relations"

| Type | Representation |
|---------------|--|
| Gantt Chart | <p>The Gantt chart displays three sequential activities on a time axis from 0 to 9 seconds. The first activity, 'Handover to Conveyor', starts at 0s and ends at 4s. The second activity, 'Move to Lift Position', starts at 4s and ends at 7s. The third activity, 'Lift skid', starts at 7s and ends at 9s. All activities are represented by blue bars.</p> |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="http://www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema- instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Gantt-Chart_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="GanttChart"> <aml:TimeInformation timeFormat="xsd:duration" startOfScale="PT0S" endOfScale="PT9S" scaleInterval="PT1S"/> <aml:IMLStep name="Handover to Conveyor" uuid="f103c162-c087-48b8-a755-3293f63d8fbf" startTime="PT0S" endTime="PT4S"/> <aml:IMLStep name="Move to Lift Position" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT4S" endTime="PT7S"/> <aml:IMLStep name="Lift skid" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT7S" endTime="PT9S"/> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.1.3 Storing of an activity sequence

Table A.2 depicts a Gantt chart with a predecessor/successor sequence among the bars.

Table A.2 – Storing of the Gantt chart example "activity sequence"

| Type | Representation |
|---------------|---|
| Gantt Chart | |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema- instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WiterName>ToolA</aml:WiterName> <aml:WiterID>010990003</aml:WiterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Gantt-Chart_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="GanttChart"> <aml:TimeInformation timeFormat="xsd:duration" startOfScale="PT0S" endOfScale="PT9S" scaleInterval="PT1S"> </aml:TimeInformation> <aml:IMLStep name="Handover to Conveyor" uuid="f103c162-c087-48b8-a755-3293f63d8fbf" startTime="PT0S" endTime="PT4S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLStep> <aml:IMLStep name="Move to Lift Position" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT4S" endTime="PT7S"> <ConnectionPointOut connectionPointOutId="2"/> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLStep name="Lift skid" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT7S" endTime="PT9S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLTransition uuid="78cb4068-b9ea-4492-ab64-587839949a8e"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> </aml:IMLTransition> <aml:IMLTransition uuid="909b2b5b-8575-41ea-986e-b20d81d302d7"> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLTransition> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

| Type | Representation |
|------|--|
| | <pre> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.1.4 Storing of an activity sequence with divergences

In Table A.3, a simultaneous divergence is introduced to describe multiple successors of one bar in IML.

Table A.3 – Storing of the Gantt chart example "activity sequence with divergence"

| Type | Representation |
|---------------|--|
| Gantt Chart | <p>The Gantt chart displays three activities over a time axis from 0 to 18 seconds. 'Initialize Robot 1' runs from 0 to 6 seconds. 'Execute Manufacturing Robot 1' runs from 6 to 18 seconds. 'Initialize Robot 2' runs from 6 to 10 seconds. Arrows indicate the flow from the end of 'Initialize Robot 1' to the start of the other two activities.</p> |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema- instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Gantt-Chart_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="GanttChart"> <aml:TimeInformation timeFormat="xsd:duration" startOfScale="PT0S" endOfScale="PT18S" scaleInterval="PT1S"> </aml:TimeInformation> <aml:IMLStep name="Initialize Robot 1" uuid="f103c162-c087-48b8-a755-3293f63d8fbf" startTime="PT0S" endTime="PT6S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLStep> <aml:IMLStep name="Execute Manufacturing Robot 1" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT9S" endTime="PT18S"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLStep name="Initialize Robot 2" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT6S" endTime="PT10S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> </aml:IMLStep> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

| Type | Representation |
|------|---|
| | <pre> <aml:IMLTransition uuid="78cb4068-b9ea-4492-ab64-587839949a8e"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="2"/> </aml:IMLTransition> <aml:IMLSimultaneousDivergence uuid="ffc63a33-5f4f-48fe-98d3-6a49deaaeec1"> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLSimultaneousDivergence> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.1.5 Storing of an activity sequence with convergences

The Gantt chart example in Table A.4 includes a convergence with the predecessor sequence among the bars of the Gantt chart. This results in a simultaneous convergence in the IML.

Table A.4 – Storing of the Gantt chart example "activity sequence with convergences"

| Type | Representation |
|---------------|---|
| Gantt Chart | |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Gantt-Chart_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="GanttChart"> <aml:TimeInformation timeFormat="xsd:duration" startOfScale="PT0S" endOfScale="PT18S" scaleInterval="PT1S"> </aml:TimeInformation> <aml:IMLStep name="Initialize Robot 1" uuid="f103c162-c087-48b8-a755-3293f63d8bf" startTime="PT0S" endTime="PT6S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLStep> <aml:IMLStep name="Lift skid" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT7S" </pre> |

| Type | Representation |
|------|--|
| | <pre> endTime="PT9S"> <ConnectionPointOut connectionPointOutId="2"/> </aml:IMLStep> <aml:IMLStep name="Execute Manufacturing Robot 1" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT9S" endTime="PT18S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLSimultaneousConvergence uuid="ffc63a33-5f4f-48fe-98d3-6a49deaaeec1"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> </aml:IMLSimultaneousConvergence> <aml:IMLTransition uuid="78cb4068-b9ea-4492-ab64-587839949a8e"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLTransition> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.2 Example for storing activity-on-node networks

A.2.1 General

The examples in A.2.2 to A.2.5 show the usage of the mapping rules to transform activity-on-node networks to IML

A.2.2 Storing of activities without predecessor and successor relation

The activity-on-node network example shown in Table A.5 has no predecessor and successor relations among the nodes.

Table A.5 – Storing of the activity-on-node network example "activities without predecessor and successor relations"

| Type | Representation | | | | | | | | | | | | | | | | | |
|--|--|---|-----------------------|---|---|--|--|--|---|---|---|--|--|--|---|---|---|---|
| Activity-on-node network | <table><tr><td>0</td><td>4</td><td>4</td></tr></table> | | | 0 | 4 | 4 | <table><tr><td>4</td><td>3</td><td>7</td></tr></table> | | | 4 | 3 | 7 | <table><tr><td>7</td><td>2</td><td>9</td></tr></table> | | | 7 | 2 | 9 |
| | 0 | 4 | 4 | | | | | | | | | | | | | | | |
| | 4 | 3 | 7 | | | | | | | | | | | | | | | |
| 7 | 2 | 9 | | | | | | | | | | | | | | | | |
| Handover to Conveyor | | | Move to Lift Position | | | Lift Skid | | | | | | | | | | | | |
| <table><tr><td>0</td><td>0</td><td>4</td></tr></table> | | | 0 | 0 | 4 | <table><tr><td>6</td><td>2</td><td>9</td></tr></table> | | | 6 | 2 | 9 | <table><tr><td>7</td><td>0</td><td>9</td></tr></table> | | | 7 | 0 | 9 | |
| 0 | 0 | 4 | | | | | | | | | | | | | | | | |
| 6 | 2 | 9 | | | | | | | | | | | | | | | | |
| 7 | 0 | 9 | | | | | | | | | | | | | | | | |
| AML logic XML | <pre><?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema- instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WiterName> <aml:WiterID>010990003</aml:WiterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime></pre> | | | | | | | | | | | | | | | | | |

| Type | Representation |
|------|---|
| | <pre> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Activity-on- node_network_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="ActivityOnNodeNetwork "> <aml:TimeInformation timeFormat="xsd:duration"/> <aml:IMLStep name="Handover to Conveyor" uuid="f103c162-c087-48b8-a755-3293f63d8fbf" startTime="PT0S" endTime="PT4S"/> <aml:IMLStep name="Move to Lift Position" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT4S" endTime="PT7S"/> <aml:IMLStep name="Lift skid" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT7S" endTime="PT9S"/> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.2.3 Storing of an activity sequence

The example given in Table A.6 depicts an activity-on-node network with a predecessor/successor sequence among the nodes.

Table A.6 – Storing of the activity-on-node network example "activity sequence"

| Type | Representation |
|--------------------------|---|
| Activity-on-node network | <pre> graph LR N1[Handover to Conveyor] --> N2[Move to Lift Position] N2 --> N3[Lift Skid] </pre> |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="http://www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Activity-on- node_network_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="ActivityOnNodeNetwork "> <aml:TimeInformation timeFormat="xsd:duration"> </aml:TimeInformation> <aml:IMLStep name="Handover to Conveyor" uuid="f103c162-c087-48b8-a755-3293f63d8fbf" </pre> |

| Type | Representation |
|------|---|
| | <pre> startTime="PT0S" endTime="PT4S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLStep> <aml:IMLStep name="Move to Lift Position" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT4S" endTime="PT7S"> buffer="PT2H"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="2"/> </aml:IMLStep> <aml:IMLStep name="Lift skid" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT7S" endTime="PT9S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLTransition uuid="78cb4068-b9ea-4492-ab64-587839949a8e"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> </aml:IMLTransition> <aml:IMLTransition uuid="909b2b5b-8575-41ea-986e-b20d81d302d7"> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLTransition> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.2.4 Storing of an activity sequence with divergences

In example given in Table A.7 a simultaneous divergence is introduced to describe multiple successors of one node in IML.

Table A.7 – Storing of the activity-on-node network example "activity sequence with divergence"

| Type | Representation |
|--------------------------|--|
| Activity-on-node network | <pre> graph LR A[0 0 6 Initialize Robot 1] --> B[6 4 10 Execute Manufacturing Robot 1] A --> C[9 9 18 Initialize Robot 2] B --> D[8 2 12] C --> E[9 0 18] </pre> |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema- </pre> |

| Type | Representation |
|------|---|
| | <pre> instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Activity-on- node_network_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="ActivityOnNodeNetwork"> <aml:TimeInformation timeFormat="xsd:duration"/> <aml:IMLStep name="Initialize Robot 1" uuid="f103c162-c087-48b8-a755-3293f63d8bf" startTime="PT0S" endTime="PT6S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLStep> <aml:IMLStep name="Execute Manufacturing Robot 1" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT9S" endTime="PT18S"> buffer="PT2H"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLStep name="Initialize Robot 2" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT6S" endTime="PT10S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLTransition uuid="78cb4068-b9ea-4492-ab64-587839949a8e"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="2"/> </aml:IMLTransition> <aml:IMLSimultaneousDivergence uuid="ffc63a33-5f4f-48fe-98d3-6a49deaaeec1"> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLSimultaneousDivergence> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.2.5 Storing of an activity sequence with convergences

The activity-on-node network example given in Table A.8 includes a convergence with the predecessor sequence among the nodes of the activity-on-node network. This results in a simultaneous convergence in the IML.

Table A.8 – Storing of the activity-on-node network example "activity sequence with convergences"

| Type | Representation |
|--------------------------|---|
| Activity-on-node network | <pre> graph LR subgraph Path1 [] direction TB A1["7 2 9"] A2["Lift Skid"] A3["9 2 11"] end subgraph Path2 [] direction TB A4["0 6 6"] A5["Initialize Robot 1"] A6["0 6 6"] end subgraph Final [] direction TB A7["9 9 18"] A8["Execute Manufacturing Robot 1"] A9["9 0 18"] end A3 --> A8 A6 --> A8 </pre> |

| Type | Representation |
|---------------|--|
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLogic xmlns="http://www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVendor>Vendor1</aml:WriterVendor> <aml:WriterVendorURL>https://www.automationml.org/</aml:WriterVendorURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="2edd1b2e-64e1-4d2f-9fa6-12022d2ea768" name="Activity-on-node_network_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="ActivityOnNodeNetwork"> <aml:TimeInformation timeFormat="xsd:duration"> </aml:TimeInformation> <aml:IMLStep name="Initialize Robot 1" uuid="f103c162-c087-48b8-a755-3293f63d8fbf" startTime="PT0S" endTime="PT6S"> <aml:IMLStep name="Lift skid" uuid="415d47ed-c13d-4727-8f13-bc8f65921ee6" startTime="PT7S" endTime="PT9S"> <aml:IMLStep name="Execute Manufacturing Robot 1" uuid="d60104de-ec05-46df-967b-5ab846e85888" startTime="PT9S" endTime="PT18S"> <aml:IMLSimultaneousConvergence uuid="ffc63a33-5f4f-48fe-98d3-6a49deaaec1"> <aml:IMLTransition uuid="78cb4068-b9ea-4492-ab64-587839949a8e"> </aml:IMLTransition> </aml:IMLSimultaneousConvergence> </aml:IMLStep> </aml:IMLStep> </aml:IMLStep> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLogic> </pre> |

A.3 Example for storing timing diagrams

A.3.1 General

The examples given in Table A.9 to Table A.11 show the use of the mapping rules to transform timing diagrams to IML.

A.3.2 Example of storing internal signal

The example in Table A.9 handles the transition from a state change to the subsequent state.

Table A.9 – Storing of the timing diagram example "transition from a state change to the subsequent state"

| Type | Representation |
|----------------|--|
| Timing diagram | |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema- instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="7040cf3a-745e-413e-8220-557545d25497" name="Timing-Diagram_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="TimingDiagram"> <aml:Resource name="Motor1" uuid="9be204e8-f3b5-4551-b8f7-a6f8894899fd"> <aml:Status name="Fast_run" uuid="630f2307-2aa5-4d57-a436-fc30aeb10fa"/> <aml:Status name="Slow_run" uuid="2a956645-faea-4c08-91e1-0ea8fd24f75d"/> </aml:Resource> <aml:TimeInformation timeFormat="xsd:duration" startOfScale="PT0S" endOfScale="PT4S" scaleInterval="PT1S"/> <aml:IMLStep name="Motor1_1 segment" uuid="cc8c1223-3c82-4e4b-a1da-341db56a4ec3" startTime="PT0S" endTime="PT1S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLStep> <aml:IMLTransition uuid="bf3b9413-9312-4527-a9b2-944ef91c5ec4" refTime="PT1S"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="2"/> </aml:IMLTransition> <aml:IMLStep name="Motor1_2 segment" </pre> |

| Type | Representation |
|------|---|
| | <pre> uuid="ac84ba1b-ff34-4f85-ba40-a3ebe4c3829c" startTime="PT1S" endTime="PT2S"> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> </aml:IMLStep> <aml:IMLTransition uuid="bf3b9413-9312-4527-a9b2-944ef91c5ec4" refEndSegmentId="ac84ba1b-ff34-4f85-ba40-a3ebe4c3829c"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLTransition> <aml:IMLStep name="Motor1_3 segment" uuid="ac84ba1b-ff34-4f85-ba40-a3ebe4c3829c" startTime="PT2S" endTime="PT3S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> </aml:IMLStep> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.3.3 Example of storing external signal

The example in Table A.10 handles two external signals, which are fired with a delay of three seconds.

Table A.10 – Mapping of the timing diagram example "two external signals fired with delay of three seconds"

| Type | Representation |
|----------------|---|
| Timing diagram | |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="http://www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> </pre> |

| Type | Representation |
|------|--|
| | <pre> <aml:FunctionBlock uuid="7040cf3a-745e-413e-8220-557545d25497" name="Timing-Diagram_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="TimingDiagram"> <aml:Resource name="Motor1" uuid="9be204e8-f3b5-4551-b8f7-a6f8894899fd"> <aml:Status name="Fast_run" uuid="630f2307-2aa5-4d57-a436-fc30aeb10faf"/> <aml:Status name="Slow_run" uuid="2a956645-faea-4c08-91e1-0ea8fd24f75d"/> </aml:Resource> <aml:TimeInformation timeFormat="xsd:duration" startOfScale="PT0S" endOfScale="PT3S" scaleInterval="PT1S"/> <aml:IMLTransition uuid="5b6fa623-6bd1-4d06-99cd-6fcfec0e2960" refTime="PT0S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLTransition> <aml:IMLStep name="Motor1_1 segment" uuid="cc8c1223-3c82-4e4b-a1da-341db56a4ec3" startTime="PT0S" endTime="PT1S"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="2"/> </aml:IMLStep> <aml:IMLTransition uuid="bf3b9413-9312-4527-a9b2-944ef91c5ec4"> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> </aml:IMLTransition> <aml:IMLStep name="Motor1_2 segment" uuid="ac84ba1b-ff34-4f85-ba40-a3ebe4c3829c" startTime="PT1S" endTime="PT3S"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLStep> <aml:IMLTransition uuid="e9e38b5d-efcc-4b0d-be04-ba281756f511" refTime="PT3S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="5"/> </aml:IMLTransition> <aml:IMLStep name="Motor1_3 segment" uuid="053d73af-242d-4049-b7af-9af0ecef150c" startTime="PT3S" endTime="PT4S"> <ConnectionPointIn> <Connection refConnectionPointOutId="5"/> </ConnectionPointIn> </aml:IMLStep> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.3.4 Example of storing signal between two resource states flows

The example in Table A.11 handles the transformation of a signal fired by one resource state and consumed by another.

Table A.11 – Storing of the timing diagram example "signal fired by one resource state and consumed by another"

| Type | Representation |
|----------------|---|
| Timing diagram | |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLogic xmlns="http://www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVendor>Vendor1</aml:WriterVendor> <aml:WriterVendorURL>https://www.automationml.org/</aml:WriterVendorURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="7040cf3a-745e-413e-8220-557545d25497" name="Timing-Diagram_Example"> <aml:MainBody xsi:type="aml:IML" logicModelType="TimingDiagram"> <aml:Resource name="Motor1" uuid="9be204e8-f3b5-4551-b8f7-a6f8894899fd"> <aml:Status name="Fast_run" uuid="630f2307-2aa5-4d57-a436-fc30aeb10faf"/> <aml:Status name="Slow_run" uuid="2a956645-faea-4c08-91e1-0ea8fd24f75d"/> </aml:Resource> <aml:Resource name="Gripper1" uuid="9238a217-0f72-48ae-b68b-adc79089c89d"> <aml:Status name="Open" uuid="558412d4-844f-4ab1-ab6f-5e8de37cb871"/> <aml:Status name="Close" uuid="480bb954-151b-49a0-acc4-c06410c70dd0"/> </aml:Resource> <aml:TimeInformation timeFormat="xsd:duration" startOfScale="PT0S" endOfScale="PT3S" scaleInterval="PT1S"/> <aml:IMLStep name="Motor1_1 segment" uuid="cc8c1223-3c82-4e4b-a1da-341db56a4ec3" startTime="PT0S" endTime="PT1S"> <ConnectionPointOut connectionPointOutId="1"/> </aml:IMLStep> <aml:IMLTransition uuid="bf3b9413-9312-4527-a9b2-944ef91c5ec4"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="2"/> </aml:IMLTransition> <aml:IMLStep name="Motor1_2 segment" uuid="ac84ba1b-ff34-4f85-ba40-a3ebe4c3829c" startTime="PT1S" endTime="PT3S"> </pre> |

| Type | Representation |
|------|--|
| | <pre> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> </aml:IMLStep> <aml:IMLStep name="Gripper1_1 segment" uuid="db04c65d-907d-4f7d-9477-966f09c2c880" startTime="PT0S" endTime="PT1S"> <ConnectionPointOut connectionPointOutId="3"/> </aml:IMLStep> <aml:IMLTransition uuid="635abe59-5e21-4107-88be-e70fdb05a764" refEndSegmentId="cc8c1223-3c82-4e4b-a1da-341db56a4ec3"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="4"/> </aml:IMLTransition> <aml:IMLStep name="Gripper1_2 segment" uuid="d33b1e7c-68f1-4841-bc1e-54f6c77f7998" startTime="PT1S" endTime="PT3S"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> </aml:IMLStep> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.4 Example for storing sequential function charts

The example given in Table A.12 shows the use of the mapping rules to store sequential function charts in AML logic XML.

Table A.12 – Example for storing sequential function chart

| Type | Representation |
|---------------------------|---|
| Sequential function chart | <pre> sequenceDiagram S0 --> S1: TRUE S1 --> S2: S1.T > T#1s S2 --> S1: S2.T > T#1s S1 --> Output: N OUTPUT </pre> |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="http://www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WiterName>ToolA</aml:WiterName> <aml:WiterID>010990003</aml:WiterID> <aml:WriterVender>Vendor1</aml:WriterVender> <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock name="IEC61131-3_SFC" uuid="e7a28279-952c-465d-850e-cf234a3d27da"> <aml:Parameters> <aml:OutputVars> <aml:Variable name="Output" orderWithinParamSet="1" uuid="69200d1f-1752-4d5c-92ef-cd6fdb697e9f"> <aml:Type> <TypeName>BOOL</TypeName> </aml:Type> </aml:Variable> </aml:OutputVars> </aml:Parameters> <aml:MainBody xsi:type="SFC"> <!--Sub-elements within this element are completely same as IEC 61131-10 SFC.--> <SfcObject xsi:type="Step" name="S0" initialStep="true"> <ConnectionPointOut connectionPointOutId="1"/> </SfcObject> <SfcObject xsi:type="Transition"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> </SfcObject> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

| Type | Representation |
|------|---|
| | <pre> <ConnectionPointOut connectionPointOutId="2"/> <Condition> <TextualPredicate> <PredicateContent xsi:type="ST"> <ST>TRUE</ST> </PredicateContent> </TextualPredicate> </Condition> </SfcObject> <SfcObject xsi:type="SelectionConvergence"> <ConnectionPointIn> <Connection refConnectionPointOutId="14"/> </ConnectionPointIn> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="3"/> </SfcObject> <SfcObject xsi:type="Step" name="S1"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="5"/> <ConnectionPointOutAction connectionPointOutId="4"/> </SfcObject> <CommonObject xsi:type="ActionBlocks"> <ConnectionPointIn> <Connection refConnectionPointOutId="4"/> </ConnectionPointIn> <ActionBlock> <ActionQualifier qualifier="N"/> <ComplexOperand>Output</ComplexOperand> </ActionBlock> </CommonObject> <SfcObject xsi:type="Transition"> <ConnectionPointIn> <Connection refConnectionPointOutId="5"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="6"/> <Condition> <TextualPredicate> <PredicateContent xsi:type="ST"> <ST> <![CDATA[S1.t > T#1s]]> </ST> </PredicateContent> </TextualPredicate> </Condition> </SfcObject> <SfcObject xsi:type="Step" name="S2"> <ConnectionPointIn> <Connection refConnectionPointOutId="6"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="10"/> </SfcObject> <SfcObject xsi:type="Transition"> <ConnectionPointIn> <Connection refConnectionPointOutId="10"/> </ConnectionPointIn> <ConnectionPointOut connectionPointOutId="14"/> <Condition> <TextualPredicate> <PredicateContent xsi:type="ST"> <ST> <![CDATA[S2.t > T#1s]]> </ST> </PredicateContent> </TextualPredicate> </Condition> </pre> |

| Type | Representation |
|------|--|
| | <pre> </TextualPredicate> </Condition> </SfcObject> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.5 Example for storing function block diagrams

The example given in Table A.13 shows the use of the mapping rules to store function block diagrams in AML logic XML.

Table A.13 – Example for storing a function block diagram

| Type | Representation |
|------------------------|---|
| Function block diagram | |
| AML logic XML | <pre> <?xml version="1.0" encoding="utf-8"?> <aml:AMLLogic xmlns="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1 AMLLogicXML.xsd" schemaVersion="1.0"> <aml:WriterHeader> <aml:WriterName>ToolA</aml:WriterName> <aml:WriterID>010990003</aml:WriterID> <aml:WriterVendor>Vendor1</aml:WriterVendor> <aml:WriterVendorURL>https://www.automationml.org/</aml:WriterVendorURL> <aml:WriterVersion>1.00</aml:WriterVersion> <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime> <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting> <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle> <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID> <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation> </aml:WriterHeader> <aml:Types> <aml:FunctionBlock uuid="8d5952dc-05fd-4fb4-bbf8-64b6815c1d58" name="IEC61131-3_FBD"> <aml:Vars> <aml:Variable name="S1" uuid="746f9a22-236e-4933-9fd0-dfe66166f39c"> <aml:Type> <TypeName>BOOL</TypeName> </aml:Type> </aml:Variable> <aml:Variable name="S1TON" uuid="da6c6c9b-efb7-4841-b876-c8c88021876e"> <aml:Type> <TypeName>TON</TypeName> </aml:Type> </pre> |

| Type | Representation |
|------|--|
| | <pre> </aml:Variable> <aml:Variable name="S2" uuid="46d7b82a-b7cf-40b0-bb53-92d1477dbce4"> <aml:Type> <TypeName>BOOL</TypeName> </aml:Type> </aml:Variable> </aml:Vars> <aml:MainBody xsi:type="FBD"> <Network xsi:type="FbdNetwork" evaluationOrder="1"> <Documentation xsi:type="SimpleText">S1 -> S2</Documentation> <FbdObject xsi:type="DataSource" identifier="S1"> <ConnectionPointOut connectionPointOutId="1"/> </FbdObject> <FbdObject xsi:type="DataSource" identifier="T#1s"> <ConnectionPointOut connectionPointOutId="2"/> </FbdObject> <FbdObject xsi:type="Block" typeName="TON" instanceName="S1TON"> <InputVariables> <InputVariable parameterName="IN"> <ConnectionPointIn> <Connection refConnectionPointOutId="1"/> </ConnectionPointIn> </InputVariable> <InputVariable parameterName="PT"> <ConnectionPointIn> <Connection refConnectionPointOutId="2"/> </ConnectionPointIn> </InputVariable> </InputVariables> <OutputVariables> <OutputVariable parameterName="Q"> <ConnectionPointOut connectionPointOutId="3"/> </OutputVariable> <OutputVariable parameterName="ET"> <ConnectionPointOut connectionPointOutId="4"/> </OutputVariable> </OutputVariables> </FbdObject> <FbdObject xsi:type="DataSource" identifier="S1"> <ConnectionPointOut connectionPointOutId="5"/> </FbdObject> <FbdObject xsi:type="Block" typeName="AND"> <InputVariables> <InputVariable parameterName="IN1" suppressName="true" negated="true"> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> </InputVariable> <InputVariable parameterName="IN2" suppressName="true"> <ConnectionPointIn> <Connection refConnectionPointOutId="5"/> </ConnectionPointIn> </InputVariable> </InputVariables> <OutputVariables> <OutputVariable parameterName="OUT" suppressName="true"> <ConnectionPointOut connectionPointOutId="6"/> </OutputVariable> </OutputVariables> </FbdObject> <FbdObject xsi:type="DataSink" identifier="S1"> <ConnectionPointIn> <Connection refConnectionPointOutId="6"/> </ConnectionPointIn> </FbdObject> <FbdObject xsi:type="DataSource" identifier="S2"> <ConnectionPointOut connectionPointOutId="7"/> </FbdObject> <FbdObject xsi:type="Block" typeName="OR"> <InputVariables> <InputVariable parameterName="IN1" suppressName="true"> </pre> |

| Type | Representation |
|------|---|
| | <pre> <ConnectionPointIn> <Connection refConnectionPointOutId="3"/> </ConnectionPointIn> </InputVariable> <InputVariable parameterName="IN2" suppressName="true"> <ConnectionPointIn> <Connection refConnectionPointOutId="7"/> </ConnectionPointIn> </InputVariable> </InputVariables> <OutputVariables> <OutputVariable parameterName="OUT" suppressName="true"> <ConnectionPointOut connectionPointOutId="8"/> </OutputVariable> </OutputVariables> </FbdObject> <FbdObject xsi:type="DataSink" identifier="S2"> <ConnectionPointIn> <Connection refConnectionPointOutId="8"/> </ConnectionPointIn> </FbdObject> </Network> </aml:MainBody> </aml:FunctionBlock> </aml:Types> </aml:AMLLogic> </pre> |

A.6 Example for storing mathematical expressions

The following example shows the use of the mapping rules to store mathematical expressions in AML logic XML.

The example shows the flow rate of the valve. The flow rate depends on the pressure at the input and output side and the adjustment of the adjustment screw. In the supercritical range the flow rate remains constant with a changing input/output pressure ratio. In the subcritical range, the flow rate decreases if the input/output pressure decreases; see Figure A.1.

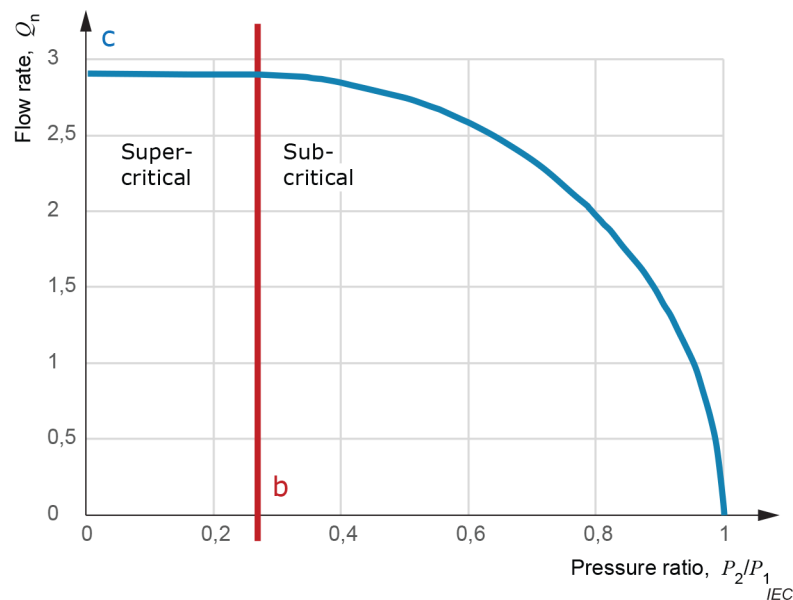


Figure A.1 – Flow rate of valves

The flow rate can be calculated with the following formula:

$$Q = \begin{cases} P_1 * n * C * \sqrt{1 - \left(\frac{\frac{P_2}{P_1} - b}{1 - b} \right)^2}, & \frac{P_2}{P_1} \geq b \text{ (subcritical)} \\ P_1 * n * C, & \frac{P_2}{P_1} < b \text{ (supercritical)} \end{cases} \quad (1)$$

Q = flow rate

P_1 = input absolute static pressure

P_2 = output absolute static pressure

C = sonic conductance

b = critical pressure ratio

n = setting of adjustment screw (0..1)

According to the provisions of Clause 8, mathematical expression can be stored in AML logic XML by expressing them with the content part of MathML version 2.0. The mapping is shown in Figure A.2.

```

<?xml version="1.0" encoding="utf-8"?>
<aml:AMLogic xmlns="http://www.iec.ch/public/TC65SC65BWG7TF10" xmlns:aml="http://www.automationml.org/IEC62714-4Ed1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.automationml.org/IEC62714-4Ed1
AMLogicXML.xsd" schemaVersion="1.0">
  <aml:WriterHeader>
    <aml:WiterName>ToolA</aml:WiterName>
    <aml:WiterID>010990003</aml:WiterID>
    <aml:WriterVender>Vendor1</aml:WriterVender>
    <aml:WriterVenderURL>https://www.automationml.org/</aml:WriterVenderURL>
    <aml:WriterVersion>1.00</aml:WriterVersion>
    <aml:LastWritingDateTime>2002-05-30T09:00:00</aml:LastWritingDateTime>
    <aml:WriterCulturalSetting>en-US</aml:WriterCulturalSetting>
    <aml:WriterProjectTitle>SampleXML</aml:WriterProjectTitle>
    <aml:WriterProjectID>829e1797-dd62-44ac-b29c-c6786d93ca93</aml:WriterProjectID>
    <aml:AdditionalInformation>This is a sample XML document.</aml:AdditionalInformation>
  </aml:WriterHeader>
  <aml:Types>
    <aml:FunctionBlock name="flow rate of a control valve" uuid="74e94dc8-b5a2-490c-bf0f-f0fdaa4515ef">
      <aml:Parameters>
        <aml:InputVars>
          <aml:Variable name="Var2" orderWithinParamSet="2" uuid="028a1a10-5c9f-11e7-9598-0800200c9a66">
            <aml:Type>
              <TypeName>REAL</TypeName>
            </aml:Type>
          </aml:Variable>
          <aml:Variable name="Var3" orderWithinParamSet="3" uuid="fac76284-c20f-4bd1-9c77-a65dff4fcacd">
            <aml:Type>
              <TypeName>REAL</TypeName>
            </aml:Type>
          </aml:Variable>
          <aml:Variable name="Var4" orderWithinParamSet="4" uuid="3d1c1a64-13cf-4fc0-a2bc-e70c2c7dd740">
            <aml:Type>
              <TypeName>REAL</TypeName>
            </aml:Type>
          </aml:Variable>
          <aml:Variable name="Var5" orderWithinParamSet="5" uuid="970b7f93-b3c8-4e4b-b3a6-86acfe533c94">
            <aml:Type>
              <TypeName>REAL</TypeName>
            </aml:Type>
          </aml:Variable>
          <aml:Variable name="Var6" orderWithinParamSet="6" uuid="970b7f93-b3c8-4e4b-b3a6-86acfe533c94">
            <aml:Type>
              <TypeName>REAL</TypeName>
            </aml:Type>
          </aml:Variable>
        </aml:InputVars>
        <aml:OutputVars>

```

```

<aml:Variable name="Var1" orderWithinParamSet="1" uuid="c3c8d294-3e18-4aa5-9a54-b72be85c9eb7">
  <aml:Type>
    <TypeName>REAL</TypeName>
  </aml:Type>
</aml:Variable>
</aml:OutputVars>
</aml:Parameters>
<aml:MainBody xsi:type="aml:MathematicalExpression" name="flow rate of a control valve" uuid="f59f2dba-442e-4f30-aa52-6bb83ec6180c">
  <aml:VariableMapping refMathMLVariableName="Q" refFBVariableUUID="c3c8d294-3e18-4aa5-9a54-b72be85c9eb7"
direction="Out"/>
  <aml:VariableMapping refMathMLVariableName="P1" refFBVariableUUID="028a1a10-5c9f-11e7-9598-0800200c9a66"
direction="In"/>
  <aml:VariableMapping refMathMLVariableName="P2" refFBVariableUUID="fac76284-c20f-4bd1-9c77-a65dff4fcacd"
direction="In"/>
  <aml:VariableMapping refMathMLVariableName="C" refFBVariableUUID="3d1c1a64-13cf-4fc0-a2bc-e70c2c7dd740"
direction="In"/>
  <aml:VariableMapping refMathMLVariableName="b" refFBVariableUUID="970b7f93-b3c8-4e4b-b3a6-86acfe533c94"
direction="In"/>
  <aml:VariableMapping refMathMLVariableName="n" refFBVariableUUID="970b7f93-b3c8-4e4b-b3a6-86acfe533c94"
direction="In"/>
  <aml:MathML>
    <math xmlns="http://www.w3.org/1998/Math/MathML" display = 'block'>
      <apply>
        <eq>
          <ci>Q</ci>
          <piecewise>
            <piece>
              <apply>
                <times/>
                <ci>P1</ci>
              </apply>
              <times/>
              <ci>n</ci>
            </piece>
            <apply>
              <times/>
              <ci>C</ci>
            </apply>
          </root>
          <apply>
            <minus/>
            <cn>1</cn>
          </apply>
          <power/>
          <apply>
            <divide/>
            <apply>
              <minus/>
              <apply>

```

```
</divide/>
<ci>P2</ci>
<ci>P1</ci>
</apply>
<ci>b</ci>
</apply>
<apply>
  <minus/>
  <cn>1</cn>
  <ci>b</ci>
</apply>
</apply>
<cn>2</cn>
</apply>
</apply>
</apply>
</apply>
</apply>
</apply>
<apply>
  <geq/>
  <apply>
    <divide/>
    <ci>P2</ci>
    <ci>P1</ci>
  </apply>
  <ci>b</ci>
</apply>
</piece>
<piece>
  <apply>
    <times/>
    <ci>P1</ci>
  </apply>
  <times/>
  <ci>n</ci>
  <ci>C</ci>
</apply>
</apply>
<apply>
  <lt/>
  <apply>
    <divide/>
    <ci>P2</ci>
    <ci>P1</ci>
  </apply>
```

```
<ci>b</ci>
</apply>
</piece>
</piecewise>
</apply>
</math>
</aml:MathML>
</aml:MainBody>
</aml:FunctionBlock>
</aml:Types>
</aml:AMLLLogic>
```

Figure A.2 – Example for storing a mathematical expression

Annex B (informative)

Examples for referencing logic information

B.1 General

This annex describes the referencing methods for logic information, which are stored in AML logic XML documents. It comprises the referencing of logic information that is stored within one FunctionBlock, which can be composed of several FunctionBlocks, and the referencing of interlocking information.

Logic information is expressed as logic models. To reference not only the logic model itself (as specified in B.2) but also certain parts of that logic model, e.g. a variable, additional referencing methods are specified in B.3.

NOTE 1 In the CAEX file (see Figure B.2, Figure B.5, Figure B.7, Figure B.9, and Figure B.11), GUIDs are presented in a short form such as "GUID1", "GUID100" etc. This serves the readability and acts as a real GUID.

NOTE 2 The role class "LogicModelObject" is not used in the following examples.

B.2 Referencing logic information expressed as logic models

B.2.1 General

This subclause describes the referencing of logic information expressed as logic models of an AML object as specified in 11.

B.2.2 Referencing logic information stored in one FunctionBlock

Sequencing, behaviour, or interlocking information, stored as a logic model within one FunctionBlock, is referenced by modelling a CAEX ExternalInterface with an AML InterfaceClass "LogicModelInterface" or a derivation of it, associated to it. This is depicted in Figure B.1 and Figure B.2, in which logic information expressed as SFC is referenced.

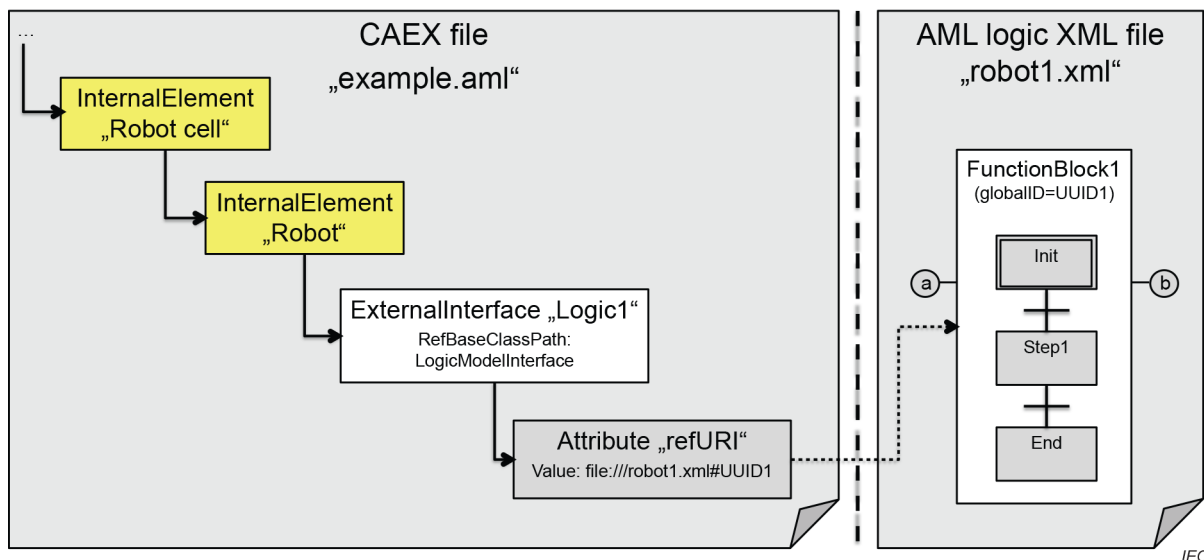


Figure B.1 – Referencing logic information (as SFC) stored in one FunctionBlock

```

<InternalElement Name="Robot cell" ID="GUID100">
  <InternalElement Name="Robot" ID="GUID101">
    <ExternalInterface Name="Logic1" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1.xml#UUID1</Value>
      </Attribute>
    </ExternalInterface>
  </InternalElement>
</InternalElement>

```

Figure B.2 – XML text of the CAEX file for referencing logic information stored in one FunctionBlock

Logic information can also be expressed as FBD as defined in 5.3. The referencing method remains unaffected.

B.2.3 Referencing logic information, which is composed of several FunctionBlocks

Sequencing, behaviour, or interlocking information, stored as a logic model, which is composed of several FunctionBlocks, is referenced by modelling a CAEX ExternalInterface with an AML InterfaceClass "LogicModelInterface" or a derivation of it.

This is depicted in Figure B.3. Here, a FunctionBlock, which contains a SFC, calls other FunctionBlocks containing SFC or FBD by IEC 61131-10 means. Also a FunctionBlock, which contains a FBD, can call other FunctionBlocks. The referencing method remains unaffected. The XML text of the CAEX file for referencing logic information, which is composed of several FunctionBlocks is the same as depicted in Figure B.2.

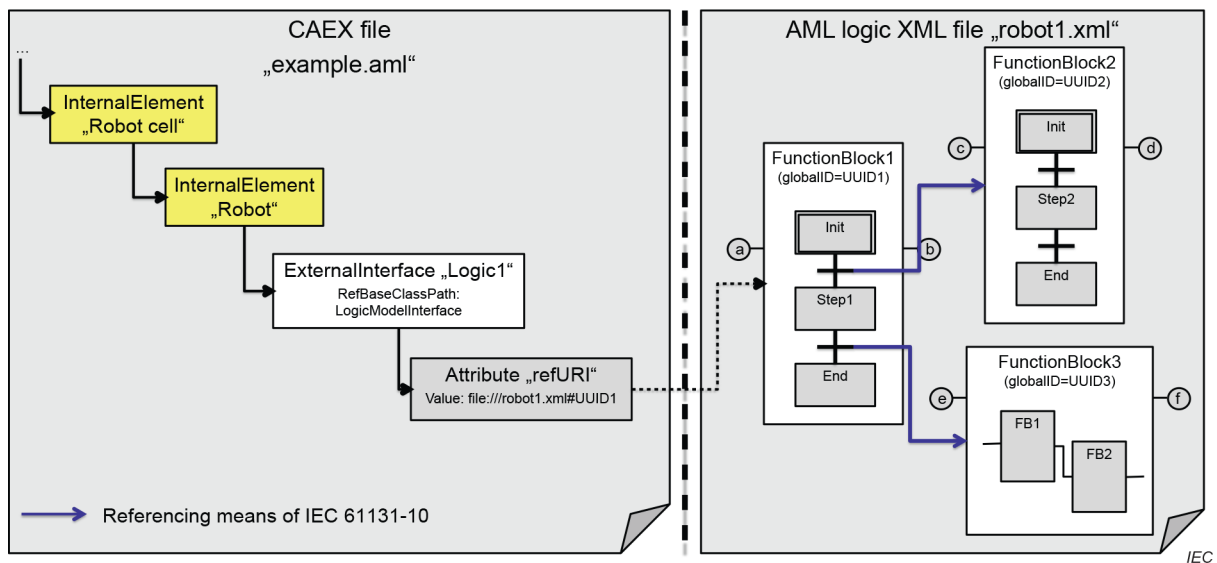


Figure B.3 – Referencing logic information, which is composed of several FunctionBlocks

It is also possible that an AML logic XML document contains several FunctionBlocks which are not interlinked with each other. In that case, the referencing method as specified in B.2.2 is applied.

B.2.4 Referencing logic information, which is composed of several AML logic XML documents

Logic information, stored as logic models, which is composed of several AML logic XML documents, is modelled by a CAEX ExternalInterface for each AML logic XML document resp. FunctionBlock. These CAEX ExternalInterfaces are associated with an AML InterfaceClass "LogicModelInterface", "SequencingLogicModelInterface", "BehaviourLogicModelInterface" or a derivation of them. This is depicted in Figure B.4 and Figure B.5. Here, the logic information is composed of two AML logic XML documents.

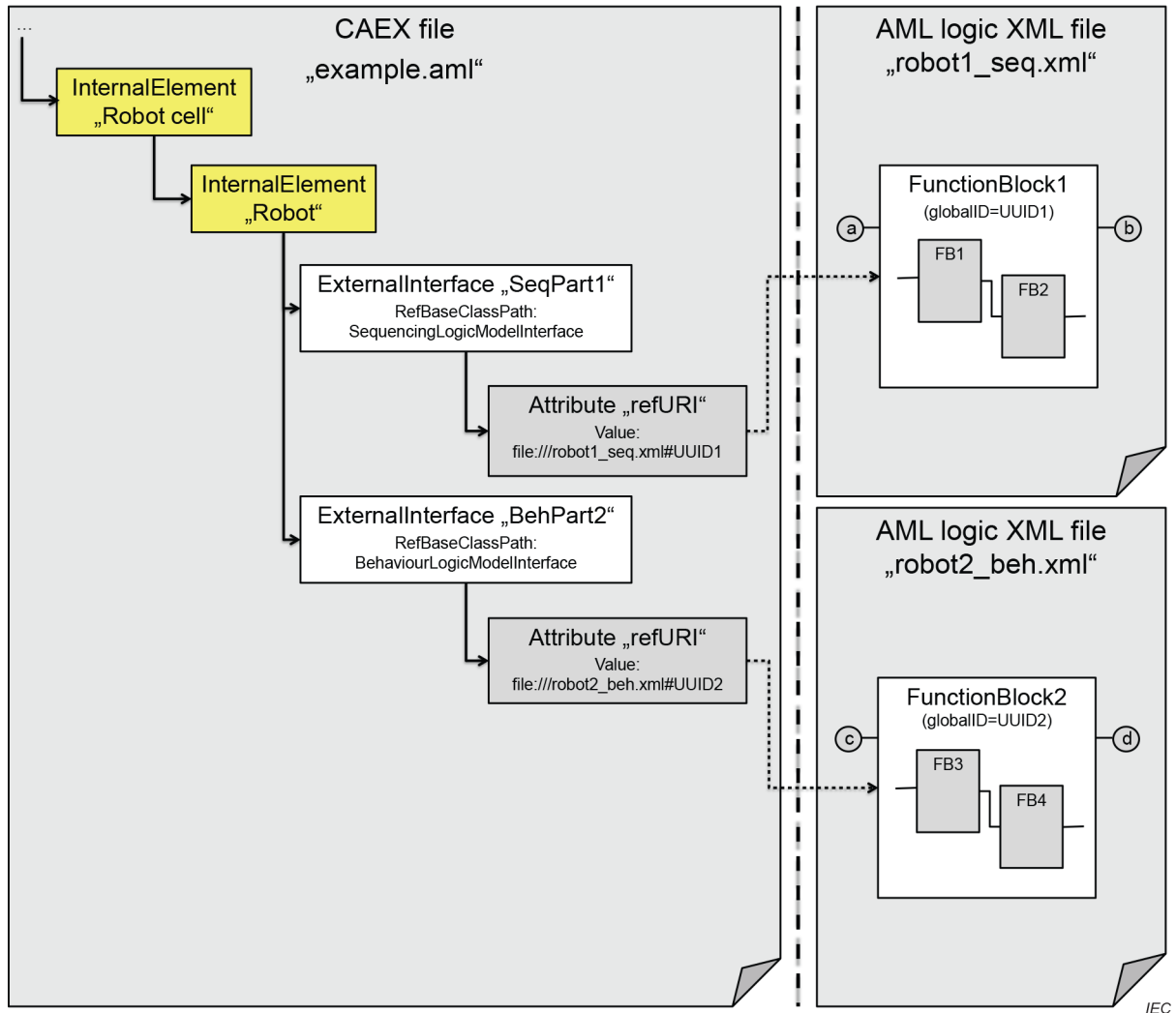


Figure B.4 – Referencing logic information which is composed of several AML logic XML documents

```
<InternalElement Name="Robot cell" ID="GUID100">
  <InternalElement Name="Robot" ID="GUID101">
    <ExternalInterface Name="SeqPart1"
      RefBaseClassPath="AutomationMLLogicInterfaceClassLib/SequencingLogicModelInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1_seq.xml#UUID1</Value>
      </Attribute>
    </ExternalInterface>
```

```

<ExternalInterface Name="BehPart2"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/BehaviourLogicModelInterface">
  <Attribute Name="refURI" AttributeDataType="xs:anyURI">
    <Value>file:///robot2_beh.xml#UUID2</Value>
  </Attribute>
</ExternalInterface>
</InternalElement>
</InternalElement>

```

Figure B.5 – XML text of the CAEX file for referencing logic information, which is composed of several AML logic XML documents

A logic model is always entirely stored in one AML logic XML document.

B.3 Referencing logic information as a part of logic models

B.3.1 General

This subclause describes the referencing of logic information as a part of logic models of an AML object as specified in 11.

B.3.2 Referencing a variable

A variable in a logic model, containing sequencing or behaviour information, is referenced by modelling a CAEX ExternalInterface of the AML InterfaceClass "VariableInterface". This is depicted in Figure B.6 and Figure B.7, in which a variable "b" is referenced. But "FunctionBlock1", which defines the variable "b", needs to be referenced as described in B.2 using an InternalLink.

NOTE Variables can be signals, parameters, internal variables etc.

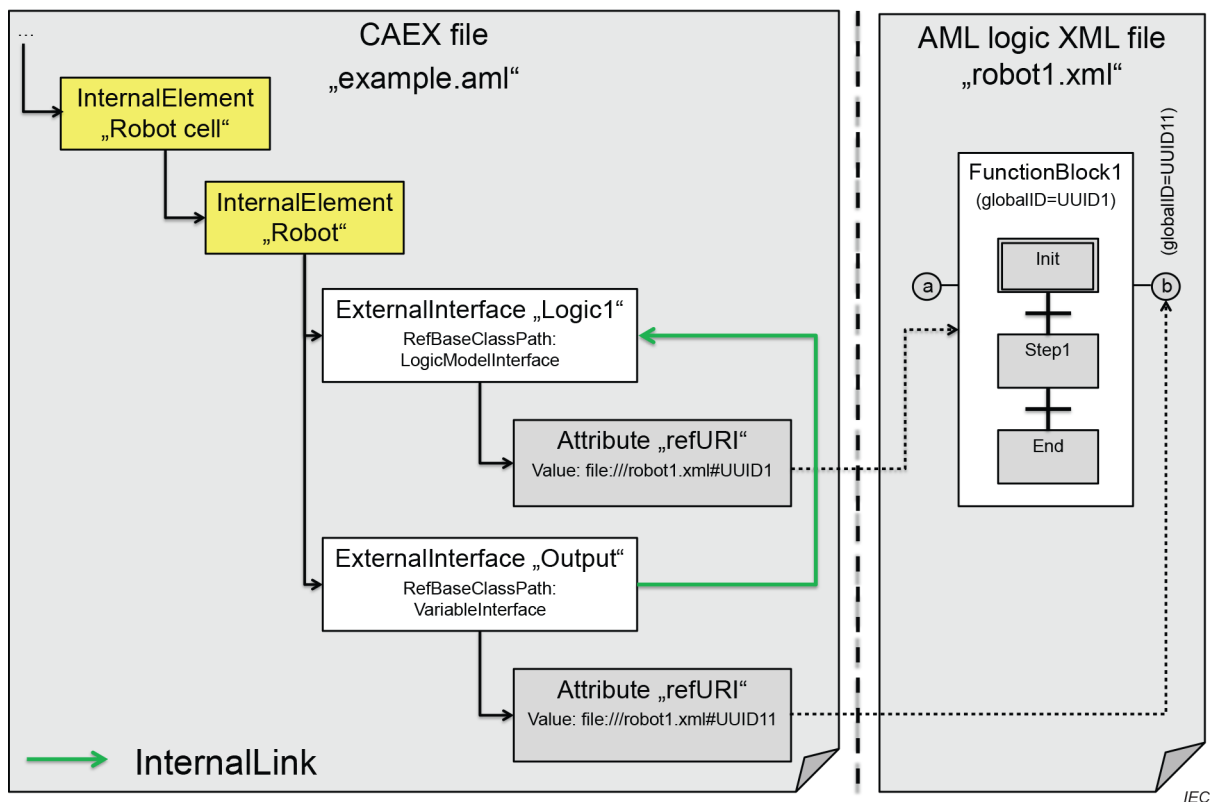


Figure B.6 – Referencing a variable

```

<InternalElement Name="Robot cell" ID="GUID100">
  <InternalElement Name="Robot" ID="GUID101">
    <ExternalInterface Name="Logic1" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1.xml#UUID1</Value>
      </Attribute>
    </ExternalInterface>
    <ExternalInterface Name="Output" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/VariableInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1.xml#UUID11</Value>
      </Attribute>
    <InternalLink Name="Link_VariableToLogic" RefPartnerSideA="GUID101:Output" RefPartnerSideB="GUID101:Logic1" />
  </ExternalInterface>
</InternalElement>
</InternalElement>

```

Figure B.7 – XML text of the CAEX file for referencing a variable

B.3.3 Referencing a logic element

A logic element in a logic model, containing sequencing or behaviour information, is referenced by modelling a CAEX ExternalInterfaces of the AML InterfaceClass "LogicModelElementInterface" or a derivation of it. This is depicted in Figure B.8 and Figure B.9, where the step "Step1" is referenced. But the logic model (stored in "FunctionBlock1") of which the logic element is a part of needs to be referenced as described in Clause B.2 using an InternalLink.

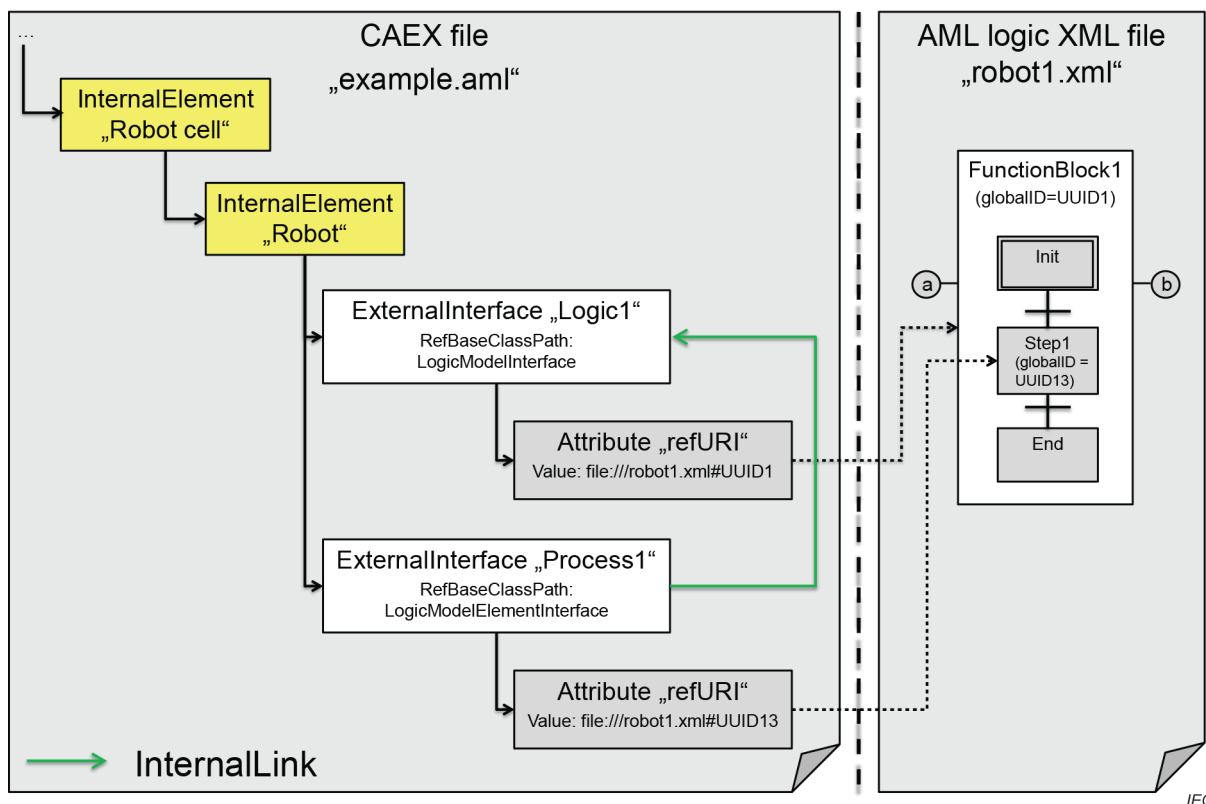


Figure B.8 – Referencing a logic element

```

<InternalElement Name="Robot cell" ID="GUID100">
  <InternalElement Name="Robot" ID="GUID101">
    <ExternalInterface Name="Logic1" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1.xml#UUID1</Value>
      </Attribute>
    </ExternalInterface>
    <ExternalInterface Name="Process1"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelElementInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1.xml#UUID13</Value>
      </Attribute>
      <InternalLink Name="Link_ProcessToLogic" RefPartnerSideA="GUID101:Process1" RefPartnerSideB="GUID101:Logic1"
/>
    </ExternalInterface>
  </InternalElement>
</InternalElement>

```

Figure B.9 – XML text of the CAEX file for referencing a logic element

B.4 Referencing logic information as a part of already referenced logic models

Parts of a logic model, containing sequencing, behaviour, or interlocking information, are referenced from AML objects as described in Clause B.3. But if this logic model is already referenced by another AML objects, it is not necessary to reference this logic model again. Instead a prompt, referencing of the parts of this logic model is possible. This is depicted in Figure B.10 and Figure B.11, in which a variable "b" of an already referenced logic model is referenced.

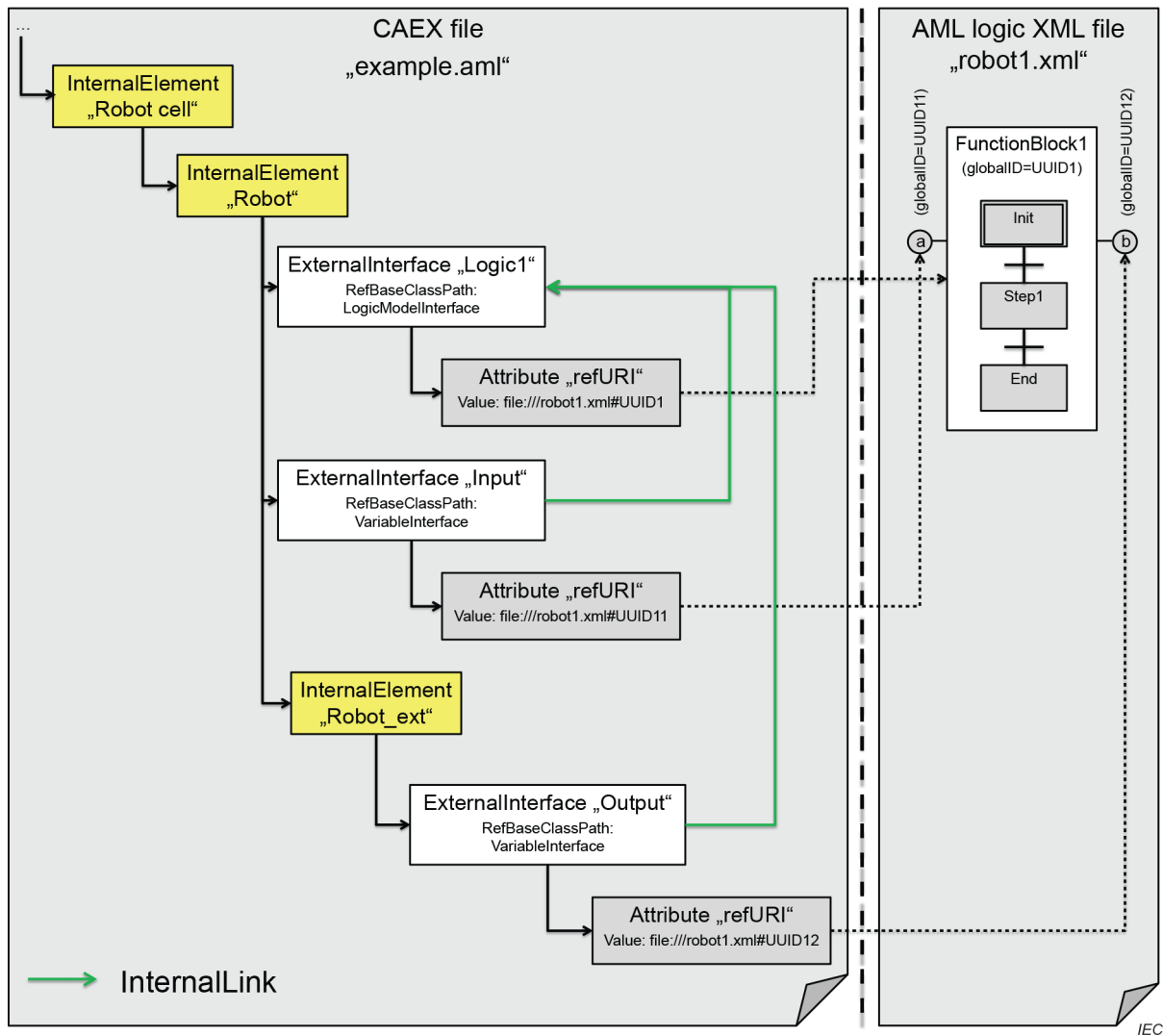


Figure B.10 – Referencing a variable of an already referenced logic model

```

<InternalElement Name="Robot cell" ID="GUID100">
  <InternalElement Name="Robot" ID="GUID101">
    <ExternalInterface Name="Logic1" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1.xml#UUID1</Value>
      </Attribute>
    </ExternalInterface>
    <ExternalInterface Name="Input" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/VariableInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///robot1.xml#UUID11</Value>
      </Attribute>
    </ExternalInterface>
    <InternalElement Name="Robot_ext" ID="GUID102">
      <ExternalInterface Name="Output" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/VariableInterface">
        <Attribute Name="refURI" AttributeDataType="xs:anyURI">
          <Value>file:///robot1.xml#UUID12</Value>
        </Attribute>
      </ExternalInterface>
      <InternalLink Name="Link_OutputToLogic1" RefPartnerSideA="GUID102:Output" RefPartnerSideB="GUID101:Logic1" />
      <InternalLink Name="Link_InputToLogic1" RefPartnerSideA="GUID101:Input" RefPartnerSideB="GUID101:Logic1" />
    </InternalElement>
  </InternalElement>
</InternalElement>

```

Figure B.11 – XML text of the CAEX file for referencing a variable of an already referenced logic model

Annex C (informative)

Examples for referencing interlocking information

C.1 General

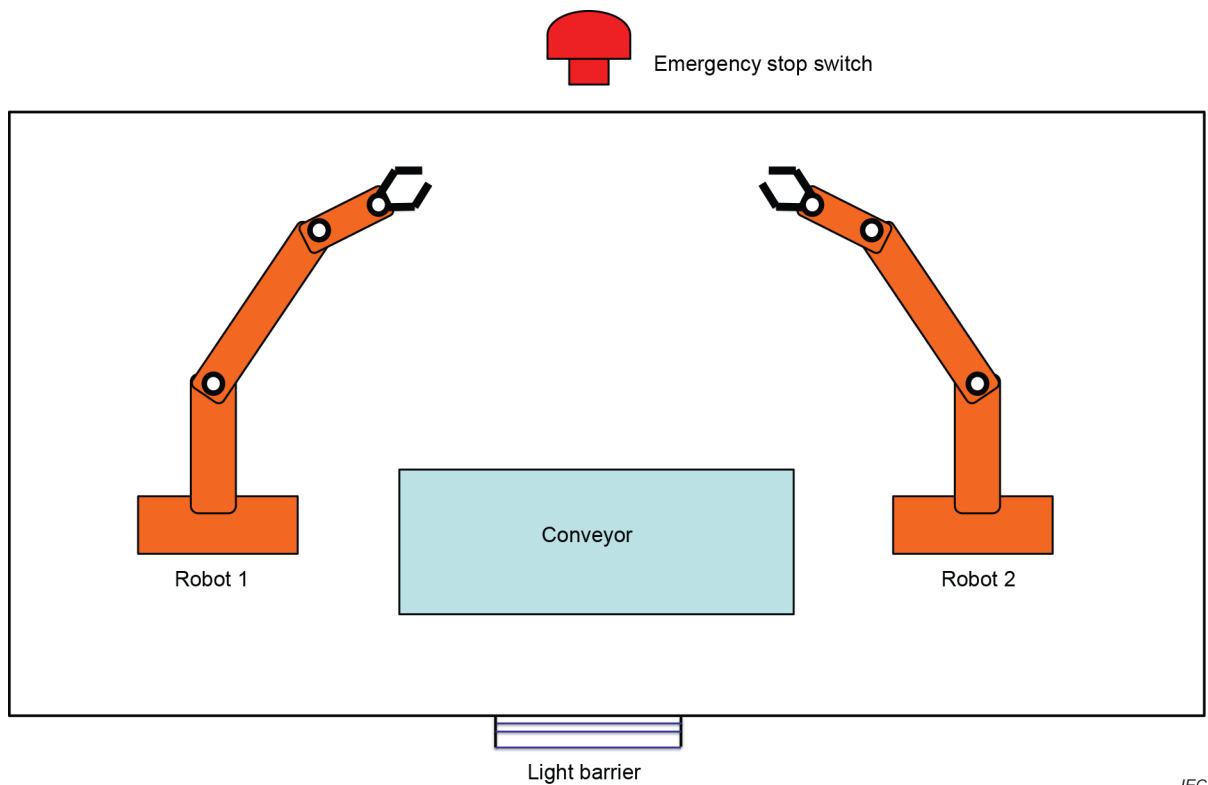
This annex describes the two mechanisms for referencing interlocking information, which is stored in AML logic XML documents. This comprises the modelling and storage of interlocking information without and with an explicitly modelled interlocking condition. Both are explained consecutively using one and the same example.

Normative provisions are given in Clause 12.

NOTE 1 In the CAEX file (see Figure C.4, Figure C.6, and Figure C.8) GUIDs are presented in a short form such as "GUID1", "GUID100" etc. This serves the readability and acts as a real GUID.

NOTE 2 The role class "LogicModelObject" is not used in the following examples.

The example depicts a manufacturing system containing different manufacturing equipment and safety devices (see Figure C.1).



IEC

Figure C.1 – Example manufacturing system

The robot cell contains the objects respectively manufacturing resources "Robot 1", "Robot 2" and "Conveyor". It also contains the safety devices "Light barrier" and "Emergency stop switch". Figure C.2 shows the exemplary aggregation of the manufacturing resources into an interlocking source group and an interlocking target group. It is allowed to have more than one interlocking source group and interlocking target group. In this example, there is one of each.

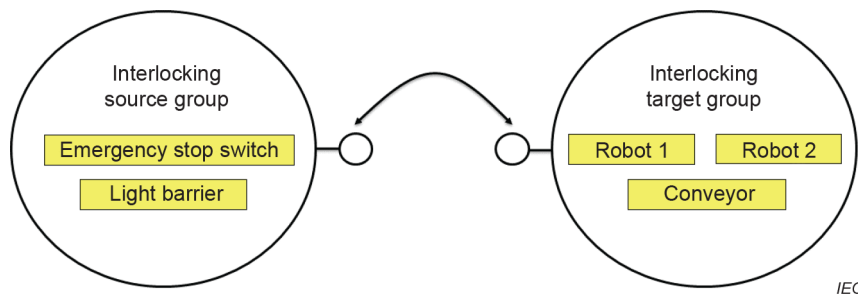


Figure C.2 – Example interlocking source group and interlocking target group

C.2 Interlocking information

The third, complementary concept of logic information covered by AML is the interlocking information expressed on two different levels of detail belonging to an industrial production system or to single components. It reflects the two main concepts:

- The causes of an interlocking condition, and
- The resulting effects of an interlocking condition.

Hereby, the cause represents the situation resulting in the need to interlock something. This case is covered by the definition and description of interlocking source groups. The effect to other groups of objects is expressed by the definition, description of and assignment to interlocking target groups.

These relations between interlocking source group and interlocking target group express the relation between cause and effect within interlocking information. In AML different levels of detail are used to exchange these interlocking conditions. Therefore, AML provides two mechanisms to reference interlocking information.

To describe interlocking source groups and interlocking target groups the basic AML group concept is used. This concept exploits fundamental CAEX capabilities and enables an integration of the necessary information within the AML top-level documents (see Clause C.3).

To express the relation between interlocking source group and interlocking target group and to express the internal logical relation within the groups, Function Block Diagrams (FBD) of the IEC 61131-3 are used (see Clause C.4). They are stored in AML logic XML documents similar to sequencing and behaviour information.

NOTE A detailed specification of cause and effect tables can be found in IEC 62881. This can be a possible implementation for an interlocking logic model.

C.3 Referencing interlocking information without interlocking condition

On the first level of detail, functional dependencies among groups of objects are modelled – without explicitly modelling an interlocking condition that would further specify that functional dependency. Objects that indicate unsafe states within the manufacturing system are aggregated to the interlocking source group. These are the "Light barrier" and "Emergency stop switch". Objects that are influenced in their behaviour by the state of the interlocking source group are aggregated to the interlocking target group. These are the "Robot 1", "Robot 2", and "Conveyor". They then need to execute specific activities to re-establish safe system behaviour, e.g. stop any movement of the objects of the interlocking target group.

For each group, an additional CAEX InternalElement, with either an AML RoleClass "InterlockingSourceGroup" or "InterlockingTargetGroup", is modelled to separate structure information from instance information. Each group contains a CAEX ExternalInterface of the AML InterfaceClass "InterlockingConnector" to relate the interlocking source group and interlocking target group to each other. CAEX Objects which belong to one of those groups are modelled as mirror objects (see Figure C.3 and Figure C.4).

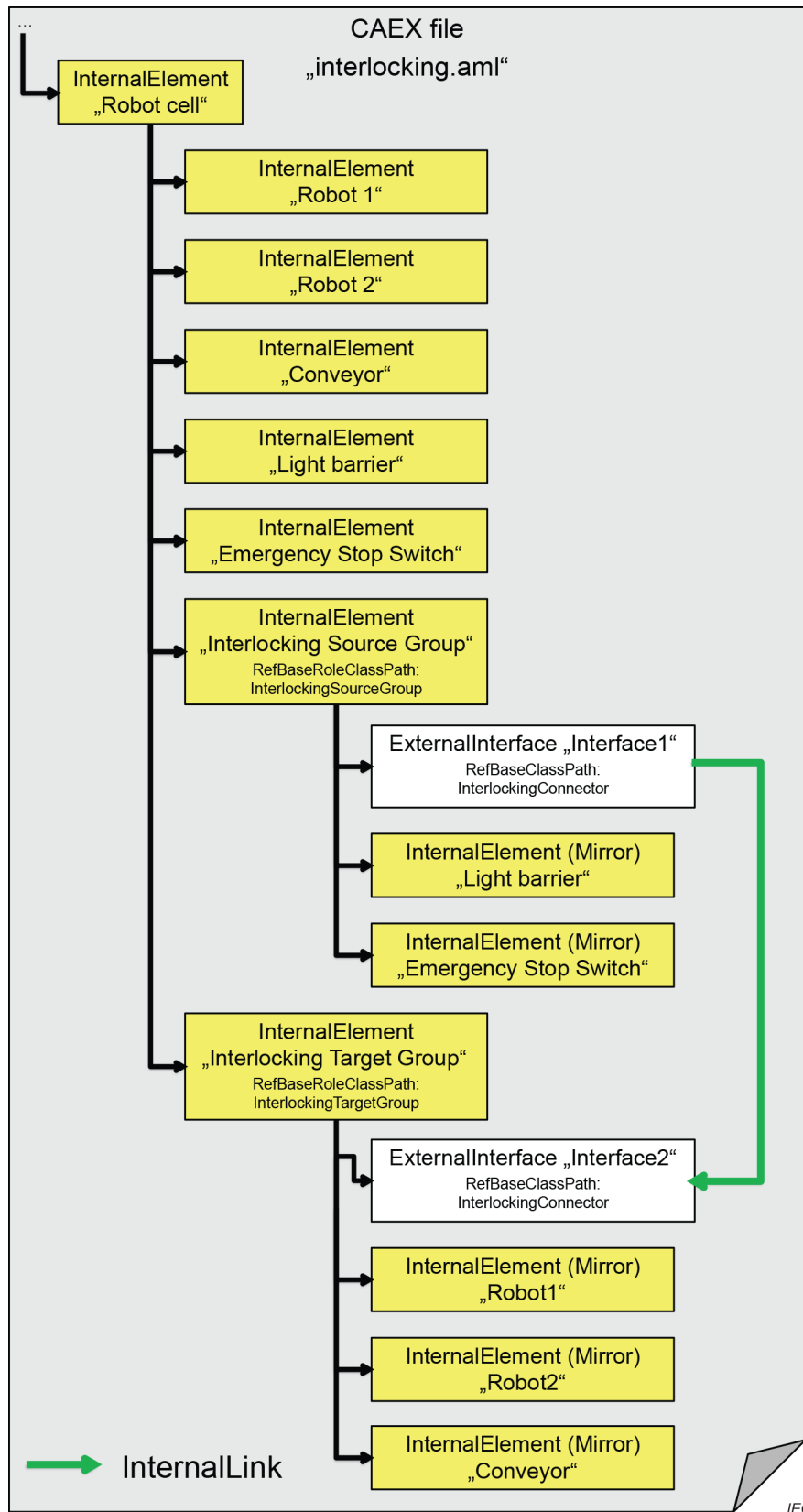


Figure C.3 – Referencing interlocking information without interlocking condition

```

<InternalElement Name="Robot cell" ID="GUID200">
  <InternalElement Name="Robot 1" ID="GUID201" />
  <InternalElement Name="Robot 2" ID="GUID202" />
  <InternalElement Name="Conveyor" ID="GUID203" />
  <InternalElement Name="Light barrier" ID="GUID204" />
  <InternalElement Name="Emergency Stop Switch" ID="GUID205" />
  <InternalElement Name="Interlocking Source Group" ID="GUID206">
    <ExternalInterface Name="Interface1"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector" />
    <InternalElement Name="Light barrier" ID="GUID207" RefBaseSystemUnitPath="GUID204" />
    <InternalElement Name="Emergency Stop Switch" ID="GUID208" RefBaseSystemUnitPath="GUID205" />
    <RoleRequirements RefBaseRoleClassPath="AutomationMLLogicRoleClassLib/InterlockingSourceGroup" />
  </InternalElement>
  <InternalElement Name="Interlocking Target Group" ID="GUID209">
    <ExternalInterface Name="Interface2"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector" />
    <InternalElement Name="Robot 1" ID="GUID210" RefBaseSystemUnitPath="GUID201" />
    <InternalElement Name="Robot 2" ID="GUID211" RefBaseSystemUnitPath="GUID202" />
    <InternalElement Name="Conveyor" ID="GUID212" RefBaseSystemUnitPath="GUID203" />
    <RoleRequirements RefBaseRoleClassPath="AutomationMLLogicRoleClassLib/InterlockingTargetGroup" />
  </InternalElement>
  <InternalLink Name="Link_InterlockingGroups" RefPartnerSideA="GUID206:Interface1"
RefPartnerSideB="GUID209:Interface2" />
</InternalElement>

```

Figure C.4 – XML text of the CAEX file for referencing interlocking information without interlocking condition

C.4 Referencing interlocking information with interlocking condition

On the second level of detail, a function describing the interlocking condition (modelled as FBD) is associated to the interlocking source group. This level extends the first level of detail described in Clause C.3.

Within the interlocking source group, two additional CAEX ExternalInterfaces are modelled: one with an AML InterfaceClass "InterlockingLogicModelInterface" following the rules of 11 and one CAEX ExternalInterface with an AML InterfaceClass "InterlockingVariableInterface". This references the unique Boolean variable that describes the output or the evaluation result of the interlocking information (see variable "z" in Figure C.5). And this result represents, whether the manufacturing system is in a safe state or not, indicated by the value of the attribute "SafeConditionEquals".

The interlocking information also needs input variables of the objects of the interlocking source group respectively the safety devices (see variable "a" and "b" in Figure C.5 and Figure C.6). For this, "Light barrier" and "Emergency stop switch" have a CAEX ExternalInterface of the AML InterfacesClass "VariableInterface".

To link the logical interface "VariableInterface" to the physical interface of the safety device, a CAEX InternalLink is used. Figure C.7 and Figure C.8 depict this exemplarily for the light barrier.

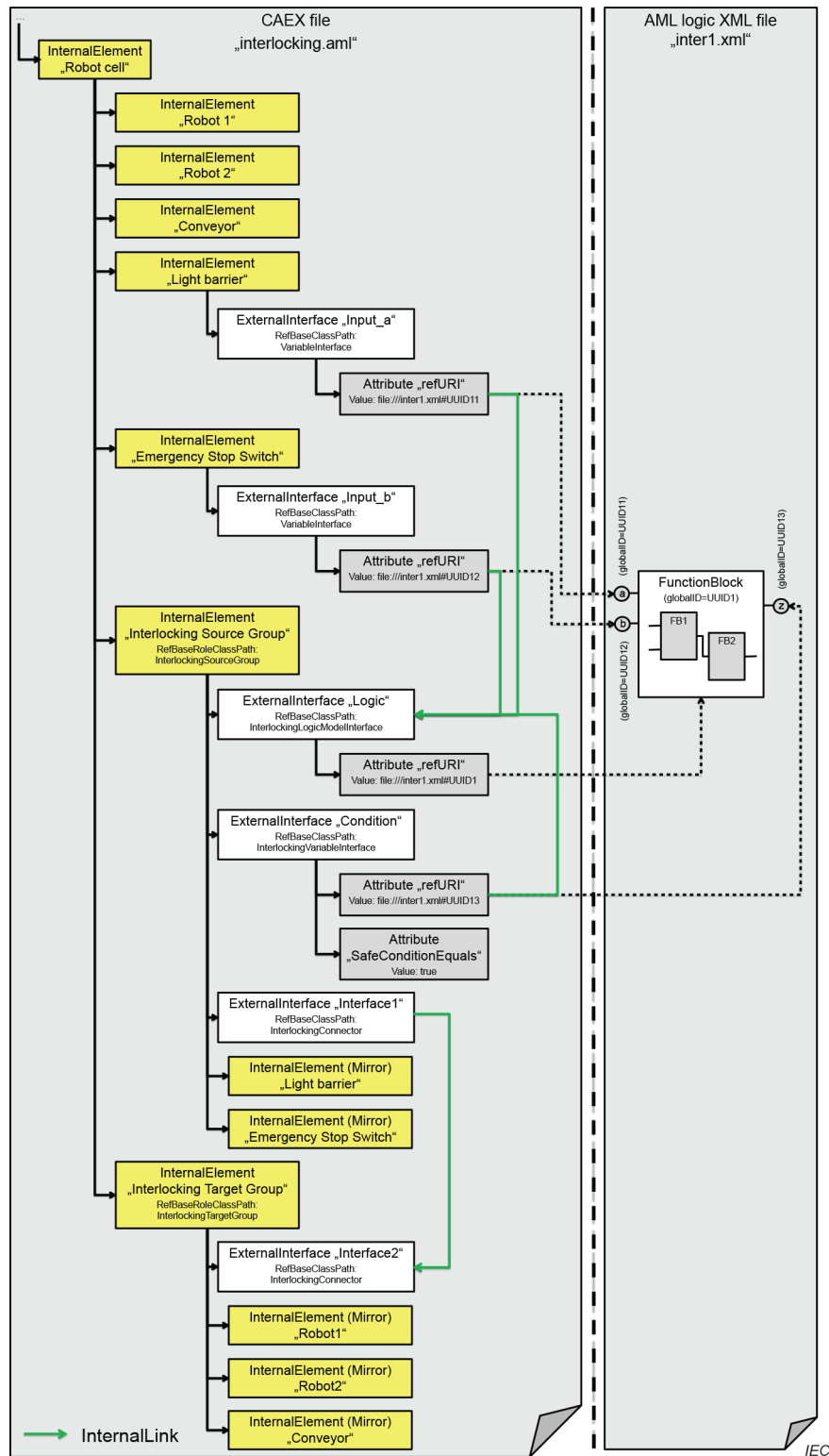


Figure C.5 – Referencing interlocking information with interlocking condition

```

<InternalElement Name="Robot cell" ID="GUID200">
  <InternalElement Name="Robot 1" ID="GUID201" />
  <InternalElement Name="Robot 2" ID="GUID202" />
  <InternalElement Name="Conveyor" ID="GUID203" />
  <InternalElement Name="Light barrier" ID="GUID204">
    <ExternalInterface Name="Input_a" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/VariableInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///inter1.xml#UUID11</Value>
      </Attribute>
    </ExternalInterface>
  </InternalElement>
  <InternalElement Name="Emergency Stop Switch" ID="GUID205">
    <ExternalInterface Name="Input_b" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/VariableInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///inter1.xml#UUID12</Value>
      </Attribute>
    </ExternalInterface>
  </InternalElement>
  <InternalElement Name="Interlocking Source Group" ID="GUID206">
    <ExternalInterface Name="Logic"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/InterlockingLogicModelInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///inter1.xml#UUID1</Value>
      </Attribute>
    </ExternalInterface>
    <ExternalInterface Name="Condition"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/InterlockingVariableInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI">
        <Value>file:///inter1.xml#UUID13</Value>
      </Attribute>
      <Attribute Name="SafeConditionEquals" AttributeDataType="xs:boolean">
        <DefaultValue>true</DefaultValue>
        <Value>true</Value>
      </Attribute>
    </ExternalInterface>
    <ExternalInterface Name="Interface1"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector" />
      <InternalElement Name="Light barrier" ID="GUID207" RefBaseSystemUnitPath="GUID204" />
      <InternalElement Name="Emergency Stop Switch" ID="GUID208" RefBaseSystemUnitPath="GUID205" />
      <InternalLink Name="Link_FunctionBlockOutputToFunctionBlock" RefPartnerSideA="GUID206:Condition"
RefPartnerSideB="GUID206:Logic" />
      <RoleRequirements RefBaseRoleClassPath="AutomationMLLogicRoleClassLib/InterlockingSourceGroup" />
    </InternalElement>
  <InternalElement Name="Interlocking Target Group" ID="GUID209">
    <ExternalInterface Name="Interface2"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector" />

```

```

<InternalElement Name="Robot 1" ID="GUID210" RefBaseSystemUnitPath="GUID201" />
<InternalElement Name="Robot 2" ID="GUID211" RefBaseSystemUnitPath="GUID202" />
<InternalElement Name="Conveyor" ID="GUID212" RefBaseSystemUnitPath="GUID203" />
<RoleRequirements RefBaseRoleClassPath="AutomationMLLogicRoleClassLib/InterlockingTargetGroup" />
</InternalElement>
<InternalLink Name="Link_InterlockingGroups" RefPartnerSideA="GUID206:Interface1"
RefPartnerSideB="GUID209:Interface2" />
<InternalLink Name="Link_LightBarrierToFunctionBlock" RefPartnerSideA="GUID204:Input_a"
RefPartnerSideB="GUID206:Logic" />
<InternalLink Name="Link_EmergencyStopToFunctionBlock" RefPartnerSideA="GUID205:Input_b"
RefPartnerSideB="GUID206:Logic" />
</InternalElement>

```

Figure C.6 – XML text of the CAEX file for referencing interlocking information with interlocking condition

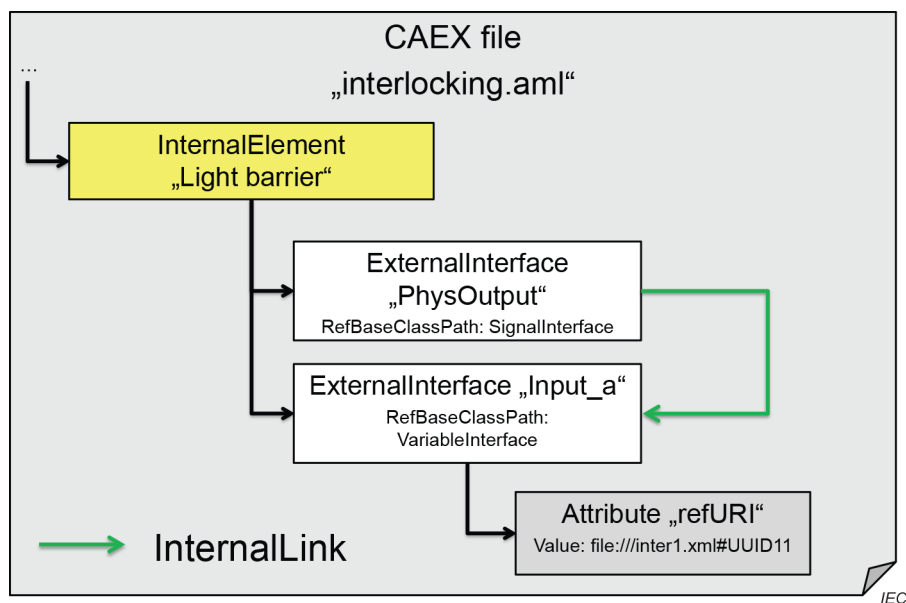


Figure C.7 – Linking logical interface with physical interface (extension to Figure C.5)

```

<InternalElement Name="Light barrier" ID="GUID204">
  <ExternalInterface Name="PhysOutput"
  RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Communication/SignalInterface" />
  <ExternalInterface Name="Input_a" RefBaseClassPath="AutomationMLLogicInterfaceClassLib/VariableInterface">
    <Attribute Name="refURI" AttributeDataType="xs:anyURI">
      <Value>file:///inter1.xml#UUID11</Value>
    </Attribute>
  </ExternalInterface>
  <InternalLink Name="Link1-LinkLogicPhysicalInterface" RefPartnerSideA="GUID204:PhysOutput"
  RefPartnerSideB="GUID204:Input_a" />
</InternalElement>

```

Figure C.8 – XML text of the CAEX file for linking logical interface with physical interface (extension to Figure C.6)

Annex D (normative)

XML representation of AML standard libraries

D.1 General

Annex D gives a normative XML representation of the libraries defined in this document.

D.2 AutomationMLLogicRoleClassLib

```
<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="AutomationMLLogicRoleClassLib.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>AutomationML Editor</WriterName>
      <WriterID>916578CA-FE0D-474E-A4FC-9E1719892369</WriterID>
      <WriterVendor>AutomationML e.V.</WriterVendor>
      <WriterVendorURL>www.AutomationML.org</WriterVendorURL>
      <WriterVersion>4.7.0.0</WriterVersion>
      <WriterRelease>4.7.0.0</WriterRelease>
      <LastWritingDateTime>2018-06-03T20:57:19.7275022+02:00</LastWritingDateTime>
      <WriterProjectTitle>unspecified</WriterProjectTitle>
      <WriterProjectID>unspecified</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0"/>
  <RoleClassLib Name="AutomationMLLogicRoleClassLib">
    <Description>AutomationMLLogicRoleClassLib specifies all logic related role classes.</Description>
    <Version>1.0.0</Version>
    <RoleClass Name="InterlockingTargetGroup">
      <ExternalInterface Name="ConnectionInterlockingSourceGroup" ID="1d3bfa87-5538-40e2-b36a-6b31c76208db"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector"/>
      <ExternalInterface Name="ReferenceInterlockingLogicModel" ID="b1fd5920-a553-4e68-b571-c672bf00592a"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInter
face/LogicModelInterface/InterlockingLogicModelInterface">
        <Attribute Name="refURI" AttributeDataType="xs:anyURI"/>
      </ExternalInterface>
    </RoleClass>
    <RoleClass Name="InterlockingSourceGroup">
      <ExternalInterface Name="ConnectionInterlockingTargetGroup" ID="3cc169e6-1e34-4d50-907f-df361c9a6d03"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector"/>
      <ExternalInterface Name="ReferenceInterlockingLogicModel" ID="59fcfe8e-7531-4f35-9b13-bd1eb9fecc88"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInter
face/LogicModelInterface/InterlockingLogicModelInterface">
        <Attribute Name="refURI" AttributeDataType="xs:anyURI"/>
      </ExternalInterface>
    </RoleClass>
  </RoleClassLib>
</CAEXFile>
```

```
<ExternalInterface Name="ReferenceInterlockingVariable" ID="09546c38-cca7-4eec-85b1-a5f33f11f8bd"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInter
face/LogicModelElementInterface/VariableInterface/InterlockingVariableInterface">
  <Attribute Name="refURI" AttributeDataType="xs:anyURI"/>
  <Attribute Name="Direction" AttributeDataType="xs:string"/>
  <Attribute Name="SafeConditionEquals" AttributeDataType="xs:boolean"/>
</ExternalInterface>
</RoleClass>
<RoleClass Name="LogicModelObject">
  <ExternalInterface Name="ReferenceSequence" ID="dbbb8615-3003-43ea-81e5-5b78e0654a79"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInter
face/LogicModelInterface/SequencingLogicModelInterface">
  <Attribute Name="refURI" AttributeDataType="xs:anyURI"/>
</ExternalInterface>
  <ExternalInterface Name="ReferenceBehaviour" ID="b34a2b98-dd40-4d94-b5ad-110d46729650"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInter
face/LogicModelInterface/BehaviourLogicModelInterface">
  <Attribute Name="refURI" AttributeDataType="xs:anyURI"/>
</ExternalInterface>
</RoleClass>
</RoleClassLib>
</CAEXFile>
```

D.3 AutomationMLLogicInterfaceClassLib

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="AutomationMLLogicInterfaceClassLib.xml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>AutomationML Editor</WriterName>
      <WriterID>916578CA-FE0D-474E-A4FC-9E1719892369</WriterID>
      <WriterVendor>AutomationML e.V.</WriterVendor>
      <WriterVendorURL>www.AutomationML.org</WriterVendorURL>
      <WriterVersion>4.7.0.0</WriterVersion>
      <WriterRelease>4.7.0.0</WriterRelease>
      <LastWritingDateTime>2018-06-03T20:57:19.7275022+02:00</LastWritingDateTime>
      <WriterProjectTitle>unspecified</WriterProjectTitle>
      <WriterProjectID>unspecified</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0"/>
  <InterfaceClassLib Name="AutomationMLLogicInterfaceClassLib">
    <Description>AutomationMLLogicInterfaceClassLib specifies all logic related interface classes for referencing AML logic
XML documents.</Description>
    <Version>1.0.0</Version>
    <InterfaceClass Name="LogicModelInterface">
      </InterfaceClass>
    <InterfaceClass Name="SequencingLogicModelInterface"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelInterface"/>
    <InterfaceClass Name="BehaviourLogicModelInterface"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelInterface"/>
    <InterfaceClass Name="InterlockingLogicModelInterface"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelInterface"/>
    <InterfaceClass Name="LogicModelElementInterface">
      </InterfaceClass>
    <InterfaceClass Name="VariableInterface"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/LogicModelElementInterface">
      <Attribute Name="Direction" AttributeDataType="xs:string"/>
      </InterfaceClass>
    <InterfaceClass Name="InterlockingVariableInterface"
RefBaseClassPath="AutomationMLLogicInterfaceClassLib/VariableInterface">
      <Attribute Name="SafeConditionEquals" AttributeDataType="xs:boolean"/>
      </InterfaceClass>
    </InterfaceClassLib>
  </CAEXFile>

```

D.4 AutomationMLPLCopenXMLInterfaceClassLib

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="AutomationMLPLCopenXMLInterfaceClassLib.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>AutomationML Editor</WriterName>
      <WriterID>916578CA-FE0D-474E-A4FC-9E1719892369</WriterID>
      <WriterVendor>AutomationML e.V.</WriterVendor>
      <WriterVendorURL>www.AutomationML.org</WriterVendorURL>
      <WriterVersion>4.7.0.0</WriterVersion>
      <WriterRelease>4.7.0.0</WriterRelease>
      <LastWritingDateTime>2018-06-03T20:57:19.7275022+02:00</LastWritingDateTime>
      <WriterProjectTitle>unspecified</WriterProjectTitle>
      <WriterProjectID>unspecified</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0"/>
  <InterfaceClassLib Name="AutomationMLPLCopenXMLInterfaceClassLib">
    <Description>AutomationMLPLCopenXMLInterfaceClassLib specifies the logic related interface class for referencing
    IEC 61131-10 documents</Description>
    <Version>1.0.0</Version>
    <InterfaceClass Name="VariableInterface"/>
  </InterfaceClassLib>
</CAEXFile>

```

Annex E (normative)

XML representation of AML logic XML schema

```
<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema xmlns:aml="http://www.automationml.org/IEC62714-4Ed1" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xhtml="http://www.w3.org/1999/xhtml" xmlns:ppx="www.iec.ch/public/TC65SC65BWG7TF10"
  targetNamespace="http://www.automationml.org/IEC62714-4Ed1" elementFormDefault="qualified"
  attributeFormDefault="unqualified" version="1.0">

  <xsd:import namespace="www.iec.ch/public/TC65SC65BWG7TF10" schemaLocation="IEC61131_10_Ed1_0.xsd"/>

  <xsd:annotation>
    <xsd:documentation>This is an example for AutomationML IML to extend IEC 61131-10 to represent vendor-specific
    extension.</xsd:documentation>
  </xsd:annotation>

  <xsd:element name="AMLLogic">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="WriterHeader" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="WiterName" type="xsd:string"/>
              <xsd:element name="WiterID" type="xsd:string"/>
              <xsd:element name="WriterVender" type="xsd:string"/>
              <xsd:element name="WriterVenderURL" type="xsd:string"/>
              <xsd:element name="WriterVersion" type="xsd:string"/>
              <xsd:element name="LastWritingDateTime" type="xsd:dateTime"/>
              <xsd:element name="WriterCulturalSetting" type="xsd:string" minOccurs="0"/>
              <xsd:element name="WriterProjectTitle" type="xsd:string" minOccurs="0"/>
              <xsd:element name="WriterProjectID" type="xsd:string" minOccurs="0"/>
              <xsd:element name="AdditionalInformation" type="xsd:string" minOccurs="0"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="Types">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:choice minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="DataTypeDecl" type="ppx:UserDefinedTypeDecl"/>
                <xsd:element name="FunctionBlock" type="aml:FunctionBlock"/>
              </xsd:choice>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="Documentation" type="ppx:TextBase" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

```

```

<xsd:attribute name="schemaVersion" type="xsd:decimal" use="required"/>
</xsd:complexType>
</xsd:element>
<xsd:complexType name="FunctionBlock">
  <xsd:annotation>
    <xsd:documentation>Conceptual POU where unnecessary elements for AML are removed from IEC 61131-10
    FunctionBlock.</xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="Parameters" type="aml:ParameterSet" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="Vars" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="Variable" type="aml:VariableDecl" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="constant" type="xsd:boolean" use="optional" default="false"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="MainBody" type="ppx:BehaviorRepresentationBase" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required"/>
  <xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
</xsd:complexType>
<!-- Extended Variable with required uuid -->
<xsd:complexType name="ParameterSet">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="InoutVars">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded">
              <xsd:complexType>
                <xsd:complexContent>
                  <xsd:extension base="aml:VariableDecl">
                    <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/>
                  </xsd:extension>
                </xsd:complexContent>
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:choice>
  </xsd:sequence>
  <xsd:element name="InputVars">
    <xsd:complexType>

```

```

<xsd:sequence>
  <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="aml:VariableDecl">
          <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/>
          <xsd:attribute name="edgeDetection" type="ppx:EdgeModifierType" use="optional" default="none"/>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:element name="OutputVars">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="aml:VariableDecl">
              <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/>
            </xsd:extension>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="VariableDecl">
  <xsd:sequence>
    <xsd:element name="Documentation" type="ppx:TextBase" minOccurs="0"/>
    <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
    <xsd:element name="Type" type="ppx:TypeRef"/>
    <xsd:element name="InitialValue" type="ppx:Value" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required"/>
  <xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
</xsd:complexType>
<xsd:complexType name="StructTypeSpec">
  <xsd:complexContent>

```

```

<xsd:extension base="ppx:TypeSpecBase">
  <xsd:sequence>
    <xsd:element name="Member" type="aml:VariableDecl" minOccurs="1" maxOccurs="unbounded"/>
    <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="overlap" type="xsd:boolean" use="optional" default="false"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Intermediate Modelling Layer -->
<xsd:complexType name="IML">
  <xsd:complexContent>
    <xsd:extension base="ppx:BehaviorRepresentationBase">
      <xsd:sequence>
        <xsd:element name="Resource" minOccurs="0" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="Status" maxOccurs="unbounded">
                <xsd:complexType>
                  <xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
                  <xsd:attribute name="name" type="xsd:string" use="required"/>
                  <xsd:attribute name="order" type="xsd:integer" use="optional"/>
                </xsd:complexType>
              </xsd:element>
            </xsd:sequence>
            <xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
            <xsd:attribute name="name" type="xsd:string" use="required"/>
            <xsd:attribute name="order" type="xsd:integer" use="optional"/>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
      <xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
      <xsd:attribute name="name" type="xsd:string" use="required"/>
      <xsd:attribute name="order" type="xsd:integer" use="optional"/>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="TimeInformation" minOccurs="0">
    <xsd:complexType>
      <xsd:attribute name="timeFormat" type="aml:TimeFormatEnum" use="required"/>
      <xsd:attribute name="startOfScale" type="aml:TimeUnion" use="optional" default="PT0S"/>
      <xsd:attribute name="endOfScale" type="aml:TimeUnion" use="optional"/>
      <xsd:attribute name="scaleInterval" type="aml:TimeUnion" use="optional"/>
    </xsd:complexType>
  </xsd:element>
  <xsd:choice minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="IMLStep" type="aml:IMLStep"/>
    <xsd:element name="IMLTransition" type="aml:IMLTransition"/>
    <xsd:element name="IMLSimultaneousDivergence" type="aml:IMLSimultaneousDivergence"/>
    <xsd:element name="IMLSimultaneousConvergence" type="aml:IMLSimultaneousConvergence"/>
  </xsd:choice>

```

```

</xsd:sequence>
<xsd:attribute name="logicModelType" type="aml:LogicModelTypeEnum" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="IMLStep">
<xsd:complexContent>
<xsd:extension base="ppx:Step">
<xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
<xsd:attribute name="startTime" type="aml:TimeUnion" use="required"/>
<xsd:attribute name="endTime" type="aml:TimeUnion" use="required"/>
<xsd:attribute name="buffer" type="xsd:duration" use="optional"/>
<xsd:attribute name="refStartStatusId" type="aml:UuidString" use="optional"/>
<xsd:attribute name="refEndStatudId" type="aml:UuidString" use="optional"/>
</xsd:extension>
<!--For Activity on Node Network-->
<!--For Timing Diagram-->
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="IMLTransition">
<xsd:complexContent>
<xsd:extension base="ppx:Transition">
<xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
<xsd:attribute name="refTime" type="aml:TimeUnion" use="optional"/>
<xsd:attribute name="refEndSegmentId" type="aml:UuidString" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="IMLSimultaneousDivergence">
<xsd:complexContent>
<xsd:extension base="ppx:SimultaneousDivergence">
<xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="IMLSimultaneousConvergence">
<xsd:complexContent>
<xsd:extension base="ppx:SimultaneousConvergence">
<xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- MathML Integration -->
<xsd:complexType name="MathematicalExpression">

```

```

<xsd:complexContent>
  <xsd:extension base="ppx:BehaviorRepresentationBase">
    <xsd:sequence>
      <xsd:element name="VariableMapping" minOccurs="0" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:attribute name="refMathMLVariableName" type="xsd:string" use="required"/>
          <xsd:attribute name="refFBVariableUUID" type="aml:UuidString" use="required"/>
          <xsd:attribute name="direction" use="required">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:enumeration value="In"/>
                <xsd:enumeration value="Out"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:attribute>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="MathML">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:any namespace="##any" processContents="lax"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="optional"/>
    <xsd:attribute name="uuid" type="aml:UuidString" use="required"/>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- String representation for Universal Unique Identifier -->
<xsd:simpleType name="LogicModelTypeEnum">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="GanttChart"/>
    <xsd:enumeration value="ActivityOnNodeNetwork"/>
    <xsd:enumeration value="TimingDiagram"/>
    <xsd:enumeration value="Unknown"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="TimeFormatEnum">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="xsd:duration"/>
    <xsd:enumeration value="xsd:time"/>
    <xsd:enumeration value="xsd:dateTime"/>
  </xsd:restriction>

```

```
<xsd:enumeration value="xsd:date"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="TimeUnion">
  <xsd:union memberTypes="xsd:duration xsd:time xsd:dateTime xsd:date"/>
</xsd:simpleType>
<xsd:simpleType name="UuidString">
  <xsd:restriction base="xsd:string">
    <xsd:annotation>
      <xsd:documentation>This shall be some textual representation of UUID, which can be GUID.</xsd:documentation>
    </xsd:annotation>
  </xsd:restriction>
</xsd:simpleType>
</xsd:schema>
```

Bibliography

IEC 62424:2008², *Representation of process control engineering – Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools*

IEC 62881:2018, *Cause and effect matrix*

² First edition. This first edition has been replaced in 2016 by a second edition IEC 62424:2016, *Representation of process control engineering – Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools*.