



BSI Standards Publication

Engineering data exchange format for use in industrial automation systems engineering — Automation markup language

Part 4: Logic

National foreword

This British Standard is the UK implementation of EN IEC 62714-4:2020. It is identical to IEC 62714-4:2020, incorporating corrigendum November 2020.

The start and finish of text introduced or altered by corrigendum is indicated in the text by tags. Text altered by IEC corrigendum November 2020 is indicated in the text by AC1 AC1.

The UK participation in its preparation was entrusted to Technical Committee GEL/65/3, Industrial communications: process measurement and control, including fieldbus.

A list of organizations represented on this committee can be obtained on request to its committee manager.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2020
Published by BSI Standards Limited 2020

ISBN 978 0 539 16269 1

ICS 25.040.40

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 August 2020.

Amendments/corrigenda issued since publication

Date	Text affected
31 December 2020	Implementation of IEC corrigendum November 2020

English Version

Engineering data exchange format for use in industrial
automation systems engineering - Automation markup language
- Part 4: Logic
(IEC 62714-4:2020)

Format d'échange de données pour une utilisation dans
l'ingénierie des systèmes d'automatisation industrielle -
Automation markup language - Partie 4: Logique
(IEC 62714-4:2020)

Datenaustauschformat für Planungsdaten industrieller
Automatisierungssysteme - Automation Markup Language
(AML) - Teil 4: Logik
(IEC 62714-4:2020)

This European Standard was approved by CENELEC on 2020-07-21. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the CEN-CENELEC Management Centre or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the CEN-CENELEC Management Centre has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Croatia, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and the United Kingdom.



European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

European foreword

The text of document 65E/654/CDV, future edition 1 of IEC 62714-4, prepared by SC 65E "Devices and integration in enterprise systems" of IEC/TC 65 "Industrial-process measurement, control and automation" was submitted to the IEC-CENELEC parallel vote and approved by CENELEC as EN IEC 62714-4:2020.

The following dates are fixed:

- latest date by which the document has to be implemented at national level by publication of an identical national standard or by endorsement (dop) 2021-04-21
- latest date by which the national standards conflicting with the document have to be withdrawn (dow) 2023-07-21

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CENELEC shall not be held responsible for identifying any or all such patent rights.

Endorsement notice

The text of the International Standard IEC 62714-4:2020 was approved by CENELEC as a European Standard without any modification.

In the official version, for Bibliography, the following note has to be added for the standard indicated:

IEC 62881:2018 NOTE Harmonized as EN IEC 62881:2018 (not modified)

Annex ZA

(normative)

Normative references to international publications with their corresponding European publications

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.

NOTE 1 Where an International Publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

NOTE 2 Up-to-date information on the latest versions of the European Standards listed in this annex is available here: www.cenelec.eu.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 61131-3	-	Programmable controllers - Part 3: Programming languages	EN 61131-3	-
IEC 61131-10	-	Programmable controllers - Part 10: PLC open XML exchange format	EN IEC 61131-10	-
IEC 62714-1	2014	Engineering data exchange format for use in industrial automation systems engineering - Automation markup language - Part 1: Architecture and general requirements	EN 62714-1	-
W3C	2004	Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation 04 February 2004	-	-
W3C	2003	Mathematical Markup Language (MathML) Version 2.0 (Second Edition), W3C Recommendation 21 October 2003	-	-
IETF RFC 5646	-	Tags for Identifying Languages	-	-
IETF RFC 4122	-	A Universally Unique Identifier (UUID) URN Namespace	-	-
PLCopen	-	XML 2.01: XML formats for IEC 61131-3	-	-

This page deliberately left blank

CONTENTS

FOREWORD	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	16
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	22
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"	31
7.3.7	Complex type "VariableDecl"	31
7.4	Complex Type "IML"	32
7.4.1	General	32
7.4.2	Attributes	33
7.4.3	Sub-element "Resource"	33
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"	35
7.4.9	Complex type "IMLSimultaneousConvergence"	35
7.5	Complex Type "MathematicalExpression"	35
7.5.1	General	35
7.5.2	Attributes	36
7.5.3	Sub-element "VariableMapping"	36
7.5.4	Sub-element "MathML"	36
7.6	Simple type "LogicModelTypeEnum"	36
7.7	Simple type "TimeUnion"	36
7.8	Simple type "TimeFormatEnum"	36
7.9	Simple type "UuidString"	37
8	Storing logic models	37
8.1	General	37
8.2	Storing Gantt charts in AML logic XML	37
8.2.1	Common rules	37
8.2.2	Storing the start of a Gantt chart	37
8.2.3	Storing bars	38
8.2.4	Storing arrows	38
8.2.5	Storing successor bars	39
8.2.6	Storing predecessor bars	41
8.3	Storing activity-on-node networks in AML logic XML	42
8.3.1	Common rules	42
8.3.2	Storing the start of an activity-on-node network	42
8.3.3	Storing nodes	43
8.3.4	Storing arrows	43
8.3.5	Storing successor nodes	44
8.3.6	Storing predecessor nodes	45
8.4	Storing timing diagrams in AML logic XML	46
8.4.1	Common rules	46
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.....	50
8.5	Storing sequential function charts in AML logic XML	51
8.5.1	Common rules	51
8.5.2	Storing variables.....	51
8.6	Storing function block diagrams in AML logic XML	52
8.6.1	Common rules	52
8.6.2	Storing variables.....	52
8.7	Storing mathematical expressions in AML logic XML.....	53
8.7.1	Common rules	53
8.7.2	Storing variables.....	54
8.7.3	Storing variable mappings	54
8.7.4	Storing mathematical expressions	55
9	Meta information about AML logic XML writer tools.....	55
10	Extensions of AML classes for logic.....	56
10.1	General.....	56
10.2	AutomationMLLogicRoleClassLib	56
10.2.1	General	56
10.2.2	RoleClass InterlockingTargetGroup	57
10.2.3	RoleClass InterlockingSourceGroup	57
10.2.4	RoleClass LogicModelObject	58
10.3	AutomationMLLogicInterfaceClassLib	59
10.3.1	General	59
10.3.2	InterfaceClass LogicModelInterface	59
10.3.3	InterfaceClass SequencingLogicModelInterface	59
10.3.4	InterfaceClass BehaviourLogicModelInterface	60
10.3.5	InterfaceClass InterlockingLogicModelInterface	60
10.3.6	InterfaceClass LogicModelElementInterface	60
10.3.7	InterfaceClass VariableInterface	61
10.3.8	InterfaceClass InterlockingVariableInterface	61
10.4	AutomationMLPLCopenXMLInterfaceClassLib.....	62
10.4.1	General	62
10.4.2	InterfaceClass VariableInterface	62
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.....	63
11.1	General.....	63
11.2	Referencing logic information.....	63
12	Linking AML objects with interlocking information	64
12.1	General.....	64
12.2	Referencing interlocking information	64
Annex A (informative)	Examples for storing logic models in AML logic XML	66
A.1	Example for storing Gantt charts.....	66
A.1.1	General	66
A.1.2	Storing of activities without predecessor and successor relation	66
A.1.3	Storing of an activity sequence	67

A.1.4	Storing of an activity sequence with divergences	69
A.1.5	Storing of an activity sequence with convergences	70
A.2	Example for storing activity-on-node networks	71
A.2.1	General	71
A.2.2	Storing of activities without predecessor and successor relation	71
A.2.3	Storing of an activity sequence	72
A.2.4	Storing of an activity sequence with divergences	73
A.2.5	Storing of an activity sequence with convergences	75
A.3	Example for storing timing diagrams	76
A.3.1	General	76
A.3.2	Example of storing internal signal	77
A.3.3	Example of storing external signal	78
A.3.4	Example of storing signal between two resource states flows	79
A.4	Example for storing sequential function charts	81
A.5	Example for storing function block diagrams	83
A.6	Example for storing mathematical expressions.....	85
Annex B (informative)	Examples for referencing logic information	89
B.1	General.....	89
B.2	Referencing logic information expressed as logic models.....	89
B.2.1	General	89
B.2.2	Referencing logic information stored in one FunctionBlock	89
B.2.3	Referencing logic information, which is composed of several FunctionBlocks	90
B.2.4	Referencing logic information, which is composed of several AML logic XML documents.....	90
B.3	Referencing logic information as a part of logic models.....	92
B.3.1	General	92
B.3.2	Referencing a variable.....	92
B.3.3	Referencing a logic element	93
B.4	Referencing logic information as a part of already referenced logic models.....	93
Annex C (informative)	Examples for referencing interlocking information	96
C.1	General.....	96
C.2	Interlocking information.....	97
C.3	Referencing interlocking information without interlocking condition	97
C.4	Referencing interlocking information with interlocking condition	99
Annex D (normative)	XML representation of AML standard libraries	103
D.1	General.....	103
D.2	AutomationMLLogicRoleClassLib	103
D.3	AutomationMLLogicInterfaceClassLib	104
D.4	AutomationMLPLCOpenXMLInterfaceClassLib.....	105
Annex E (normative)	XML representation of AML logic XML schema.....	106
Bibliography.....		110
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		

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 "AMLLogic".....	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".....	34
Figure 19 – Complex type "IMLTransition"	34
Figure 20 – Complex type "IMLSimultaneousDivergence"	35
Figure 21 – Complex type "IMLSimultaneousConvergence"	35
Figure 22 – Complex type "MathematicalExpression".....	35
Figure 23 – AutomationMLLogicRoleClassLib	57
Figure 24 – AutomationMLLogicInterfaceClassLib.....	59
Figure 25 – AutomationMLPLCopenXMLInterfaceClassLib.....	62
Figure A.1 – Flow rate of valves.....	86
Figure A.2 – Example for storing a mathematical expression	88
Figure B.1 – Referencing logic information (as SFC) stored in one FunctionBlock.....	89
Figure B.2 – XML text of the CAEX file for referencing logic information stored in one FunctionBlock	90
Figure B.3 – Referencing logic information, which is composed of several FunctionBlocks	90
Figure B.4 – Referencing logic information which is composed of several AML logic XML documents.....	91
Figure B.5 – XML text of the CAEX file for referencing logic information, which is composed of several AML logic XML documents.....	91
Figure B.6 – Referencing a variable	92
Figure B.7 – XML text of the CAEX file for referencing a variable.....	92
Figure B.8 – Referencing a logic element.....	93
Figure B.9 – XML text of the CAEX file for referencing a logic element	93
Figure B.10 – Referencing a variable of an already referenced logic model.....	94
Figure B.11 – XML text of the CAEX file for referencing a variable of an already referenced logic model.....	95
Figure C.1 – Example manufacturing system	96
Figure C.2 – Example interlocking source group and interlocking target group.....	97
Figure C.3 – Referencing interlocking information without interlocking condition	98
Figure C.4 – XML text of the CAEX file for referencing interlocking information without interlocking condition	99
Figure C.5 – Referencing interlocking information with interlocking condition	100

Figure C.6 – XML text of the CAEX file for referencing interlocking information with interlocking condition	101
Figure C.7 – Linking logical interface with physical interface (extension to Figure C.5)	102
Figure C.8 – XML text of the CAEX file for linking logical interface with physical interface (extension to Figure C.6)	102
Table 1 – Abbreviated terms	16
Table 2 – Mapping the start of a Gantt chart to IML element	38
Table 3 – Mapping of Gantt chart bars to IML elements	38
Table 4 – Mapping of Gantt chart arrow to IML elements	39
Table 5 – Mapping of Gantt chart bar with one or more successor bars to IML elements	40
Table 6 – Mapping of Gantt chart bar with one and more predecessor bars to IML elements	41
Table 7 – Mapping of the start of an activity-on-node network to IML elements	42
Table 8 – Mapping of activity-on-node network nodes to IML elements	43
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	45
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	50
Table 16 – Storing variable of a sequential function chart in AML logic XML	52
Table 17 – Storing variable of a function block diagram in AML logic XML	53
Table 18 – Storing variable of a mathematical expression in AML logic XML	54
Table 19 – Storing variable mappings of a mathematical expression in AML logic XML	55
Table 20 – Storing a mathematical expression in AML logic XML	55
Table 21 – Meta information about each AML logic XML writer tool	56
Table 22 – RoleClass InterlockingTargetGroup	57
Table 23 – RoleClass InterlockingSourceGroup	58
Table 24 – RoleClass LogicModelObject	58
Table 25 – InterfaceClass LogicModelInterface	59
Table 26 – InterfaceClass SequencingLogicModelInterface	60
Table 27 – InterfaceClass BehaviourLogicModelInterface	60
Table 28 – InterfaceClass InterlockingLogicModelInterface	60
Table 29 – InterfaceClass LogicModelElementInterface	61
Table 30 – InterfaceClass VariableInterface	61
Table 31 – InterfaceClass InterlockingVariableInterface	62
Table 32 – InterfaceClass VariableInterface	62
Table 33 – InterfaceClass InterlockingConnector	63
Table 34 – InterfaceClass PLCOpenXMLInterface	63
Table A.1 – Storing of the Gantt chart example "activities without predecessor and successor relations"	67
Table A.2 – Storing of the Gantt chart example "activity sequence"	68

Table A.3 – Storing of the Gantt chart example "activity sequence with divergence"	69
Table A.4 – Storing of the Gantt chart example "activity sequence with convergences"	70
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"	72
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"	75
Table A.9 – Storing of the timing diagram example "transition from a state change to the subsequent state"	77
Table A.10 – Mapping of the timing diagram example "two external signals fired with delay of three seconds"	78
Table A.11 – Storing of the timing diagram example "signal fired by one resource state and consumed by another"	80
Table A.12 – Example for storing sequential function chart	81
Table A.13 – Example for storing a function block diagram	83

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**ENGINEERING DATA EXCHANGE FORMAT FOR USE
IN INDUSTRIAL AUTOMATION SYSTEMS ENGINEERING –
AUTOMATION MARKUP LANGUAGE –****Part 4: Logic****FOREWORD**

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62714-4 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

The text of this International Standard is based on the following documents:

CDV	Report on voting
65E/654/CDV	65E/692/RVC

Full information on the voting for the approval of this International Standard can be found in the report on voting indicated in the above table.

This document has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts in the IEC 62714 series, published under the general title *Engineering data exchange format for use in industrial systems engineering – Automation Markup Language*, can be found on the IEC website.

The committee has decided that the contents of this document will remain unchanged until the stability date indicated on the IEC website under "<http://webstore.iec.ch>" in the data related to the specific document. At this date, the document will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

The contents of the corrigendum of November 2020 have been included in this copy.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

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.

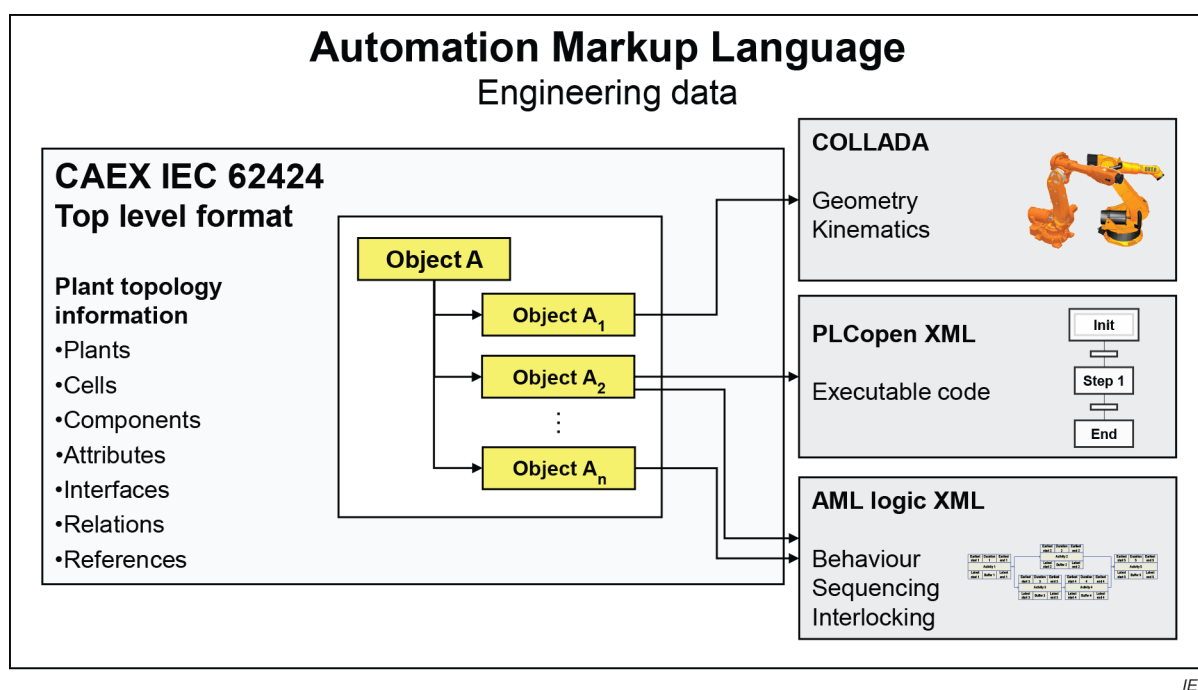


Figure 1 – Overview of the engineering data exchange format AML

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.

AC1 Clause 11 **AC1** 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.

ENGINEERING DATA EXCHANGE FORMAT FOR USE IN INDUSTRIAL AUTOMATION SYSTEMS ENGINEERING – Automation Markup Language –

Part 4: Logic

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 heterogenous 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>

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

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*.

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, AC1 11 AC1 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).

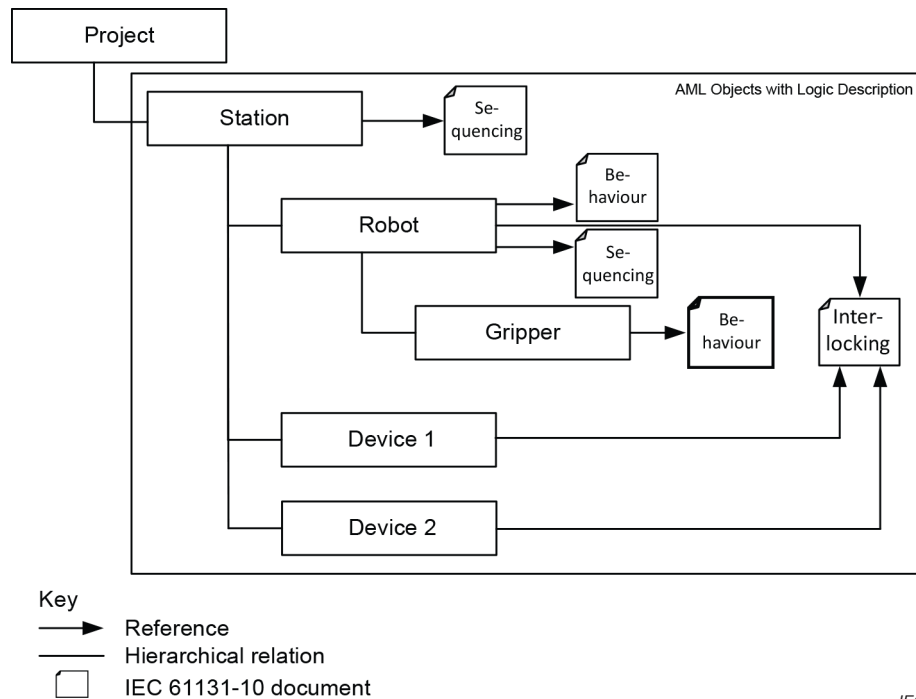


Figure 2 – Example of system representation with roles of information in AML

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).

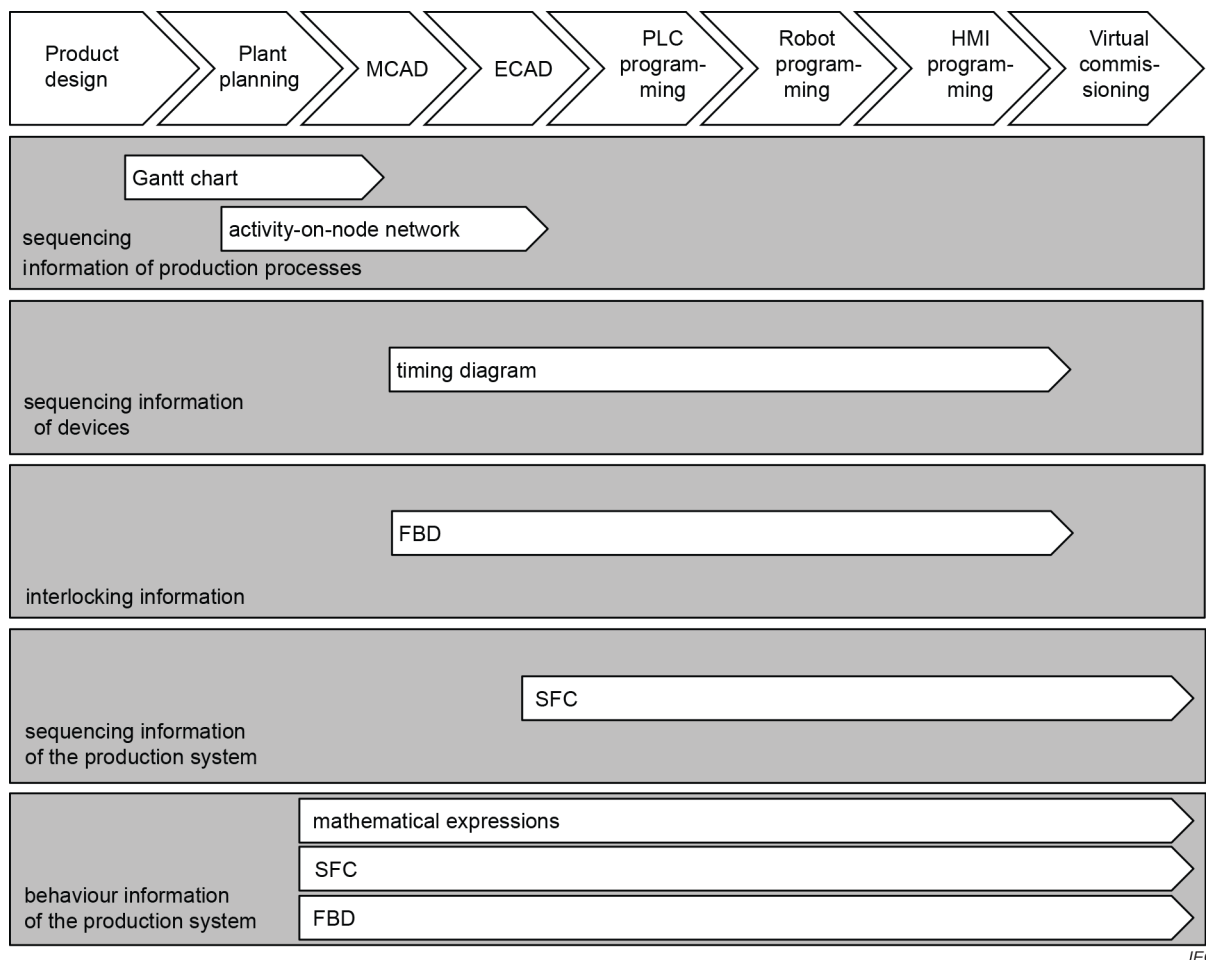


Figure 3 – Logic models in AML

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.

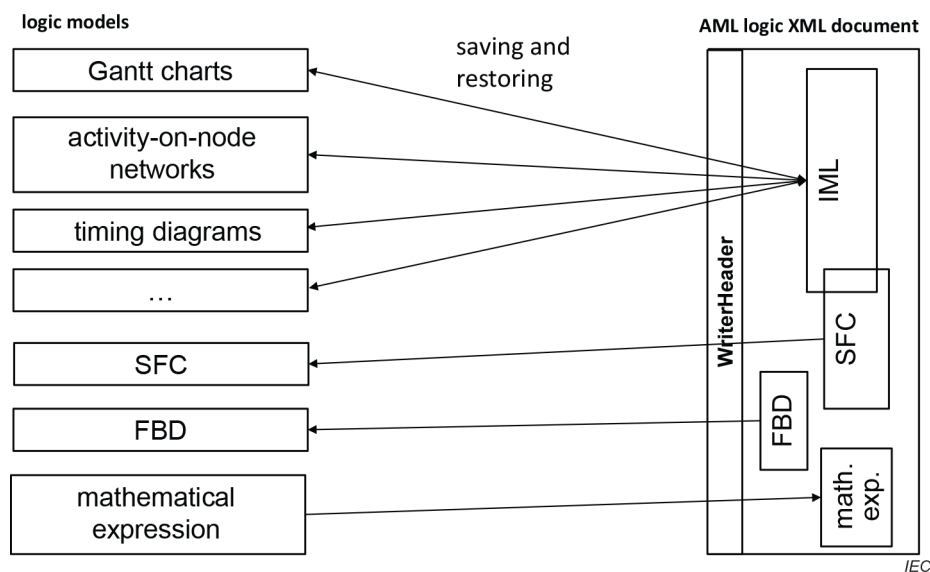


Figure 4 – Storing logic models in AML logic XML

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.

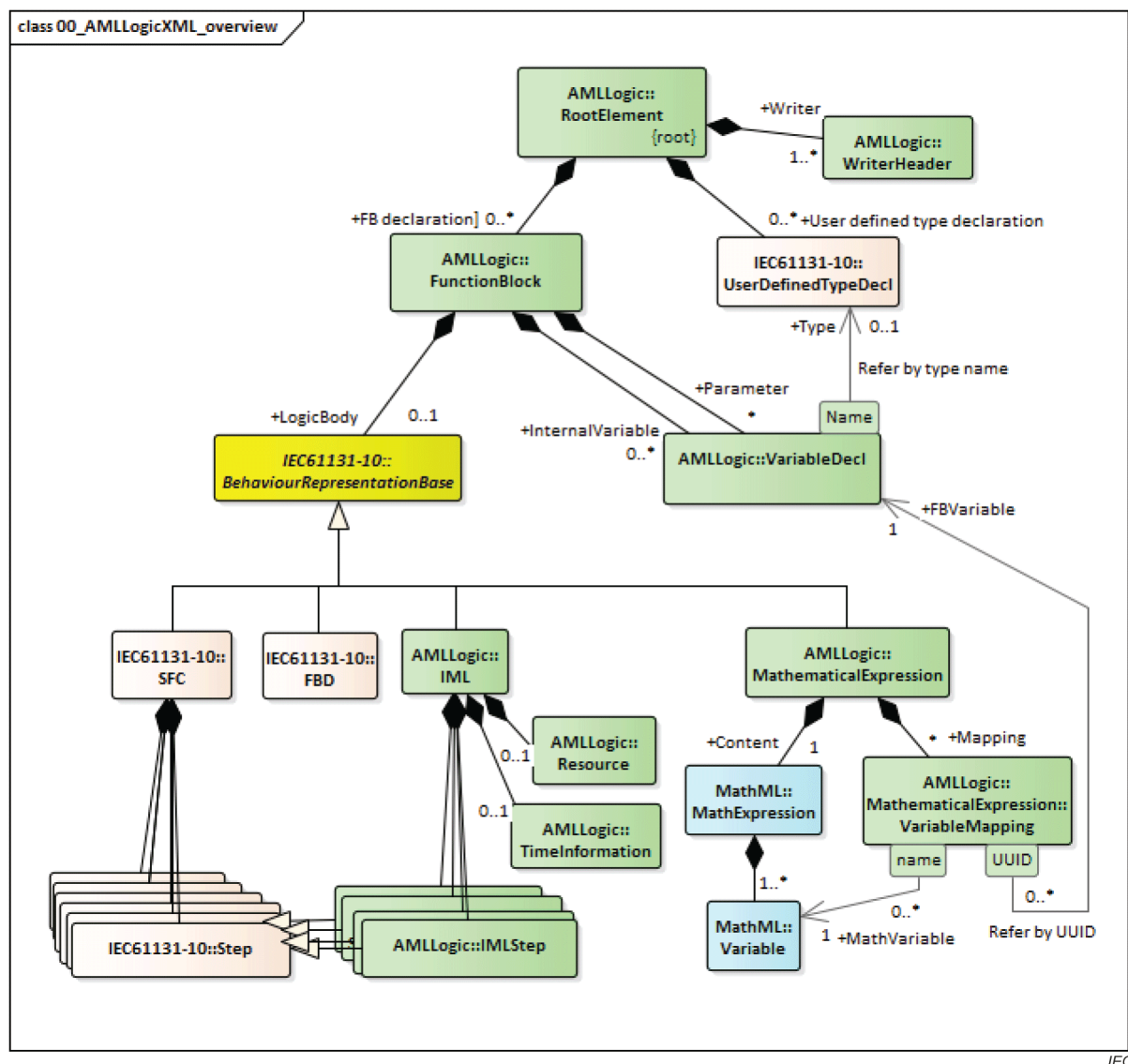


Figure 5 – Modelling elements of the AML logic XML

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 AC1 Clause 11 AC1. 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.

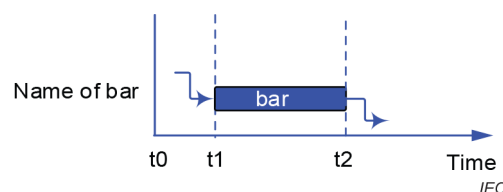


Figure 6 – Model elements of Gantt charts

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.

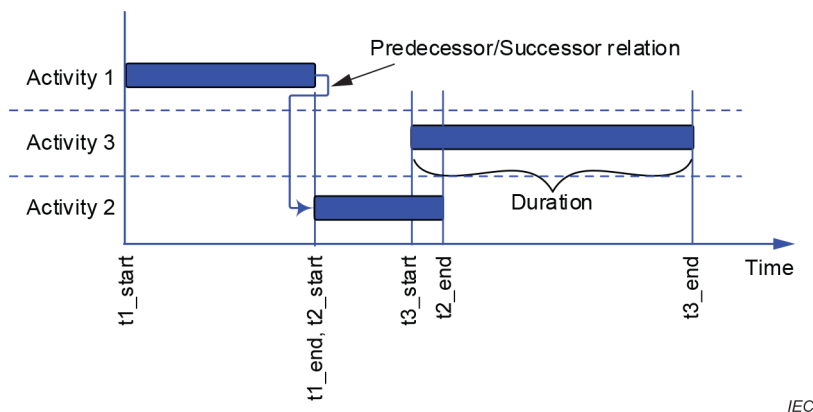


Figure 7 – Information provided by Gantt charts

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.

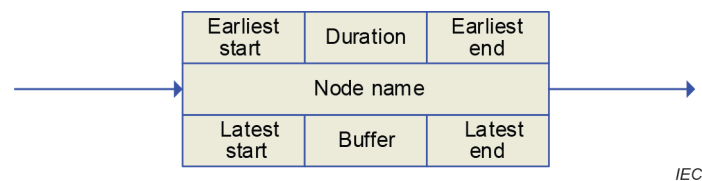


Figure 8 – Model elements of activity-on-node networks

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.

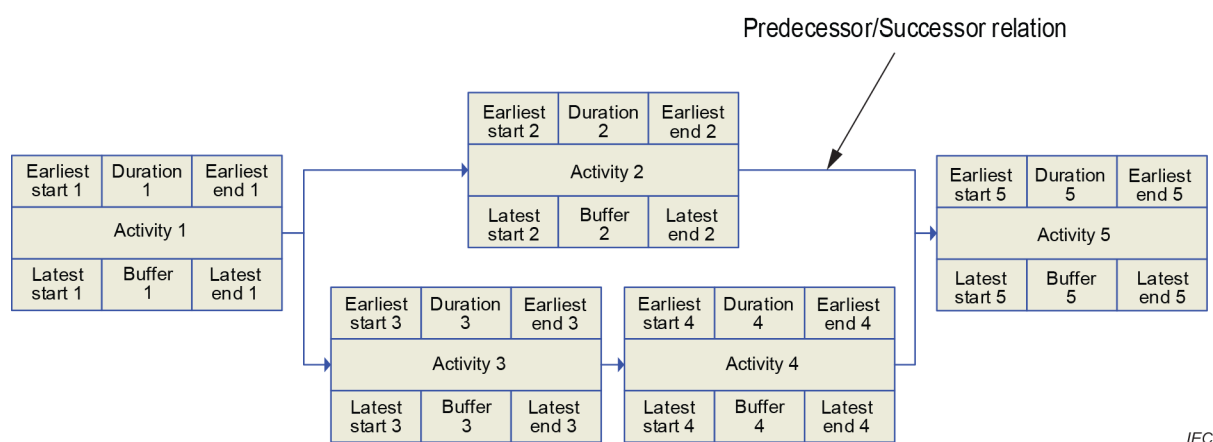


Figure 9 – Information provided by activity-on-node networks

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.

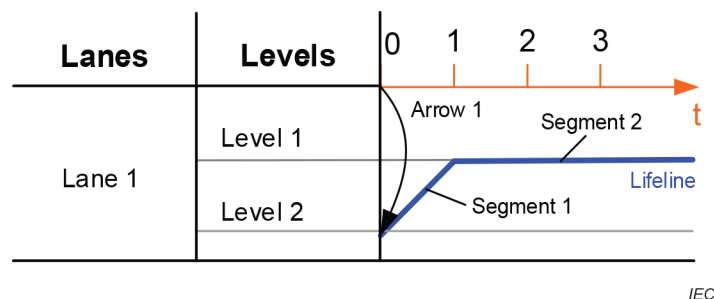


Figure 10 – Model elements of timing diagrams

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

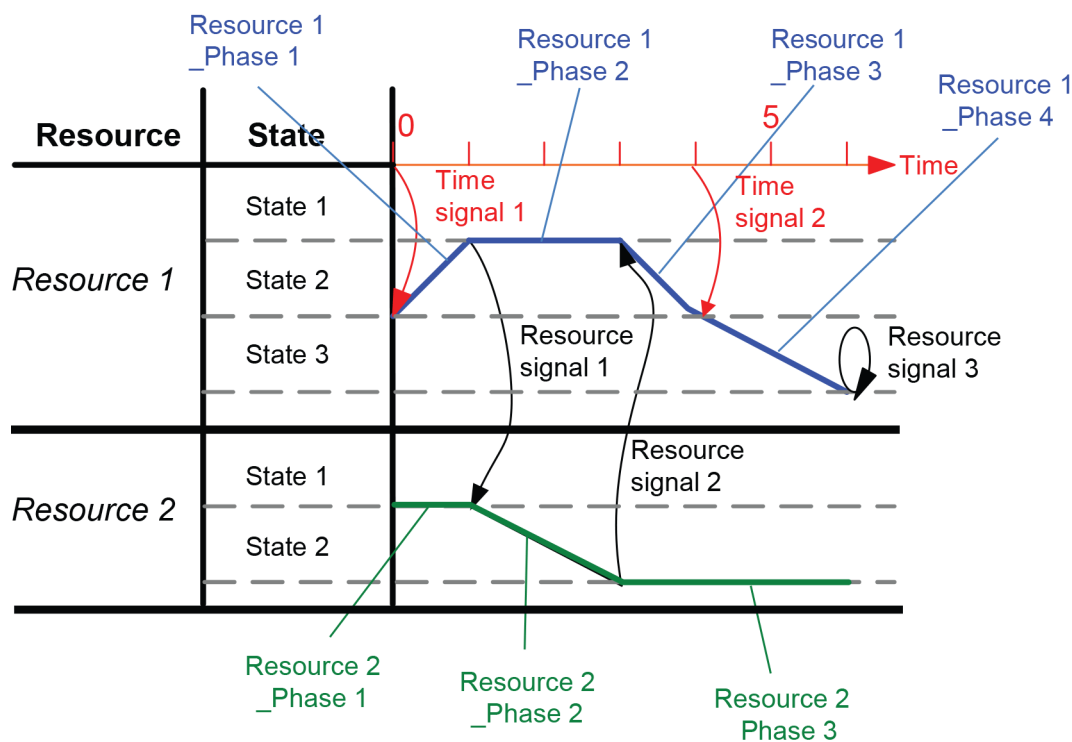
Timing diagrams shall comprise the following types of logic information:

- 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 $\boxed{AC_1}$ SFC $\langle AC_1 \rangle$ elements of IEC 61131-10 to represent Intermediate Modelling Layer (IML) described in 5.3, as shown in Figure 12.

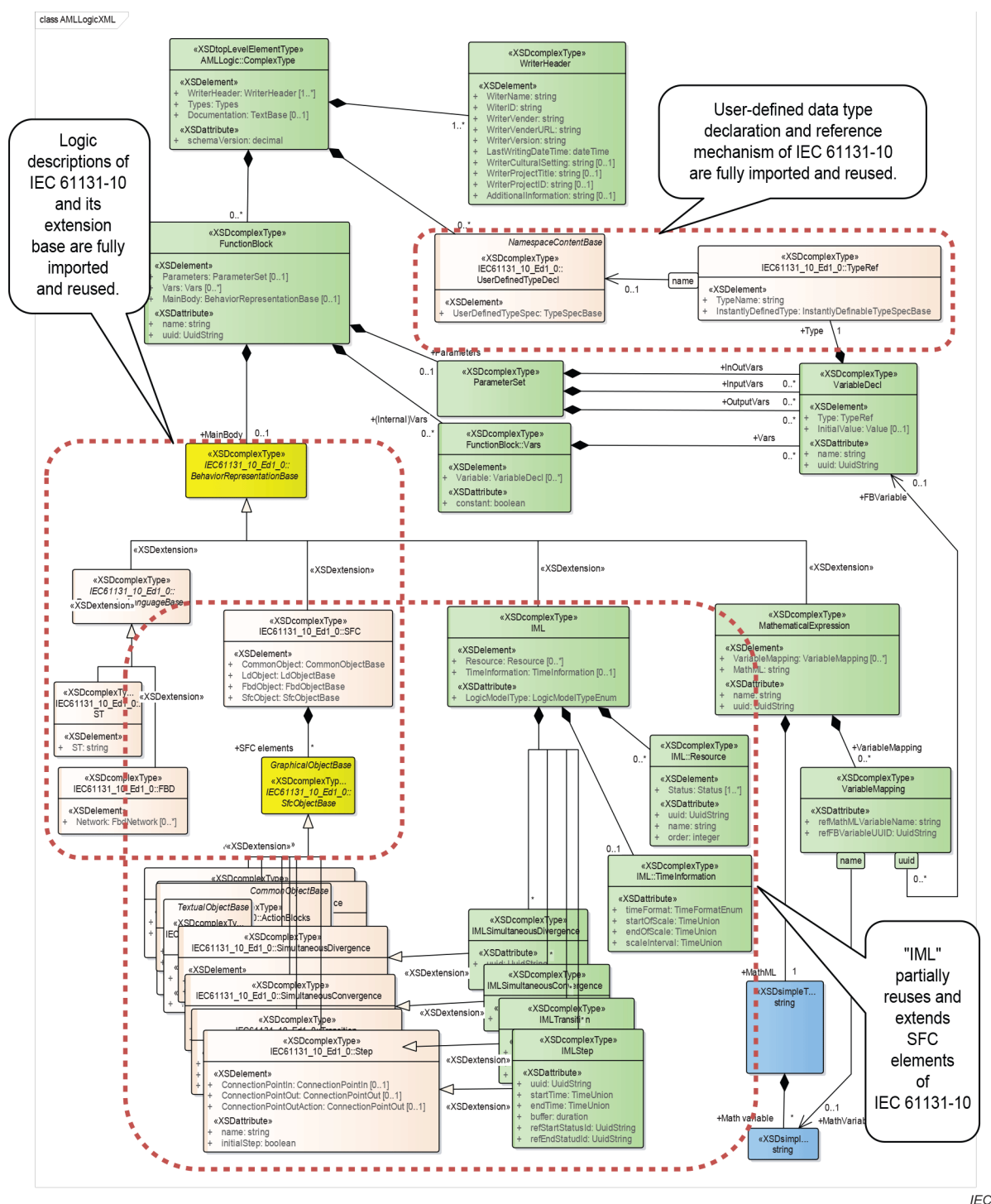


Figure 12 – AML logic schema overview

7.1.2 Schema versioning

The AML logic XML schema definition has a version attribute. Its first number equals to the edition number of IEC 62714-4. The second number is reserved for minor changes within the same edition. The file name of the schema also reflects the edition number.

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 "AMLLLogic"

7.2.1 General

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

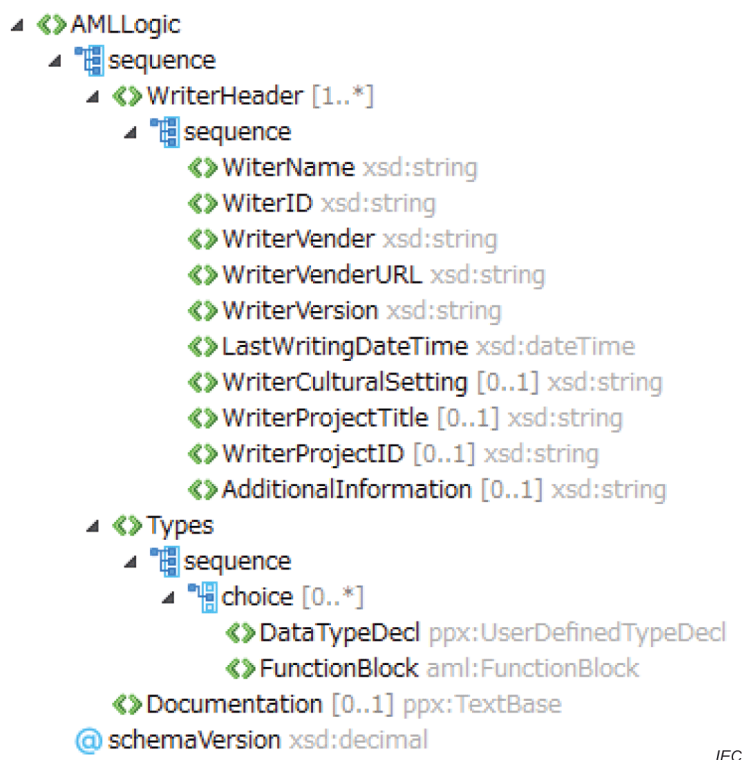


Figure 13 – Root element "AMLLLogic"

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 "ppx: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 "ppx: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.

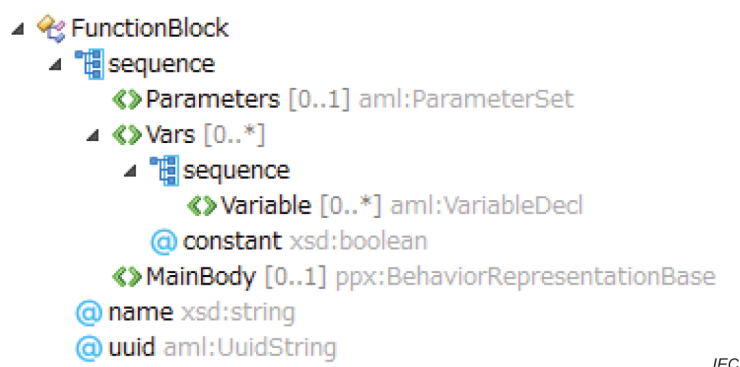
7.2.5 Sub-element "Documentation"

"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.



IEC

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).