

BS EN 62714-1:2014



BSI Standards Publication

# Engineering data exchange format for use in industrial automation systems engineering

Part 1: Architecture and  
General Requirements

**bsi.**

...making excellence a habit.  
**AVA** Documents™  
Document Sharing Hub

**National foreword**

This British Standard is the UK implementation of EN 62714-1:2014. It is identical to IEC 62714-1:2014.

The UK participation in its preparation was entrusted to Technical Committee AMT/7, Industrial communications: process measurement and control, including fieldbus.

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

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 2014.

Published by BSI Standards Limited 2014

ISBN 978 0 580 73983 5

ICS 25.040.40; 35.060; 35.240.50

**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 October 2014.

**Amendments/corrigenda issued since publication**

| Date | Text affected |
|------|---------------|
|------|---------------|

English Version

Engineering data exchange format for use in industrial  
automation systems engineering - Part 1: Architecture and  
General Requirements  
(IEC 62714-1:2014)

Format d'échange de données techniques pour une  
utilisation dans l'ingénierie des systèmes d'automatisation  
industrielle - AutomatisationML - Partie 1: Architecture et  
exigences générales  
(CEI 62714-1:2014)

Datenaustauschformat für Planungsdaten industrieller  
Automatisierungssysteme (AutomationML) - Teil 1:  
Architektur und allgemeine Festlegungen  
(IEC 62714-1:2014)

This European Standard was approved by CENELEC on 2014-07-31. 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, Former Yugoslav Republic of Macedonia, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Romania, 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: Avenue Marnix 17, B-1000 Brussels

## Foreword

The text of document 65E/385/FDIS, future edition 1 of IEC 62714-1, 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 62714-1:2014.

The following dates are fixed:

- latest date by which the document has to be (dop) 2015-05-01  
implemented at national level by  
publication of an identical national  
standard or by endorsement
- latest date by which the national (dow) 2017-07-31  
standards conflicting with the  
document have to be withdrawn

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

This document has been prepared under a mandate given to CENELEC by the European Commission and the European Free Trade Association.

## Endorsement notice

The text of the International Standard IEC 62714-1:2014 was approved by CENELEC as a European Standard without any modification.

In the official version, for Bibliography, the following notes have to be added for the standards indicated:

|                    |      |                                  |
|--------------------|------|----------------------------------|
| IEC 60027 (Series) | NOTE | Harmonized as EN 60027 (Series). |
| IEC 62264-1        | NOTE | Harmonized as EN 62264-1.        |
| IEC 62714-2        | NOTE | Harmonized as EN 62714-2         |
| ISO 80000-1        | NOTE | Harmonized as EN ISO 80000-1.    |

## Annex ZA (normative)

### Normative references to international publications with their corresponding European publications

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

NOTE 1 When 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](http://www.cenelec.eu).

| <u>Publication</u> | <u>Year</u> | <u>Title</u>                                                                                                                                                                     | <u>EN/HD</u> | <u>Year</u> |
|--------------------|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|-------------|
| IEC 62424          | 2008        | Representation of process control engineering - Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools                                                  | EN 62424     | 2009        |
| IEC 62714          | series      | Engineering data exchange format for use in industrial automation systems engineering                                                                                            | EN 62714     | series      |
| ISO/IEC 9834-8     | -           | Information technology - Procedures for the operation of object identifier registration authorities: General procedures and top arcs of the international object identifier tree | -            | -           |
| ISO/PAS 17506      | -           | Industrial automation systems and integration - COLLADA digital asset schema specification for 3D visualization of industrial data                                               | -            | -           |

## CONTENTS

|                                                                       |    |
|-----------------------------------------------------------------------|----|
| INTRODUCTION.....                                                     | 9  |
| 1 Scope.....                                                          | 11 |
| 2 Normative references .....                                          | 11 |
| 3 Terms, definitions and abbreviations .....                          | 11 |
| 3.1 Terms and definitions.....                                        | 11 |
| 3.2 Abbreviations .....                                               | 14 |
| 4 Conformity.....                                                     | 14 |
| 5 AML architecture specification .....                                | 15 |
| 5.1 General.....                                                      | 15 |
| 5.2 General AML architecture .....                                    | 15 |
| 5.3 AML document versions .....                                       | 16 |
| 5.4 Meta information about the AML source tool .....                  | 17 |
| 5.5 Object identification .....                                       | 18 |
| 5.6 AML relations specification .....                                 | 19 |
| 5.6.1 General .....                                                   | 19 |
| 5.6.2 Parent-child-relations between AML objects .....                | 19 |
| 5.6.3 Parent-child-relations between AML classes .....                | 20 |
| 5.6.4 Inheritance relations .....                                     | 21 |
| 5.6.5 Class-instance-relations .....                                  | 21 |
| 5.6.6 Instance-instance-relations .....                               | 23 |
| 5.7 AML document reference specification .....                        | 25 |
| 5.7.1 General .....                                                   | 25 |
| 5.7.2 Referencing COLLADA documents .....                             | 25 |
| 5.7.3 Referencing PLCopen XML documents .....                         | 25 |
| 5.7.4 Referencing additional documents .....                          | 25 |
| 6 AML base libraries .....                                            | 25 |
| 6.1 General.....                                                      | 25 |
| 6.2 General provisions .....                                          | 25 |
| 6.3 AML interface class library – AutomationMLInterfaceClassLib ..... | 26 |
| 6.3.1 General .....                                                   | 26 |
| 6.3.2 InterfaceClass AutomationMLBaseInterface .....                  | 28 |
| 6.3.3 InterfaceClass Order .....                                      | 28 |
| 6.3.4 InterfaceClass PortConnector .....                              | 29 |
| 6.3.5 InterfaceClass PPRConnector .....                               | 29 |
| 6.3.6 InterfaceClass ExternalDataConnector .....                      | 29 |
| 6.3.7 InterfaceClass COLLADAInterface .....                           | 30 |
| 6.3.8 InterfaceClass PLCopenXMLInterface .....                        | 30 |
| 6.3.9 InterfaceClass Communication .....                              | 30 |
| 6.3.10 InterfaceClass SignalInterface .....                           | 31 |
| 6.4 AML basic role class library – AutomationMLBaseRoleClassLib ..... | 31 |
| 6.4.1 General .....                                                   | 31 |
| 6.4.2 RoleClass AutomationMLBaseRole .....                            | 33 |
| 6.4.3 RoleClass Group .....                                           | 33 |
| 6.4.4 RoleClass Facet .....                                           | 34 |

|                       |                                                                |    |
|-----------------------|----------------------------------------------------------------|----|
| 6.4.5                 | RoleClass Port .....                                           | 34 |
| 6.4.6                 | RoleClass Resource .....                                       | 36 |
| 6.4.7                 | RoleClass Product .....                                        | 36 |
| 6.4.8                 | RoleClass Process .....                                        | 37 |
| 6.4.9                 | RoleClass Structure .....                                      | 37 |
| 6.4.10                | RoleClass ProductStructure .....                               | 37 |
| 6.4.11                | RoleClass ProcessStructure .....                               | 38 |
| 6.4.12                | RoleClass ResourceStructure .....                              | 38 |
| 6.4.13                | RoleClass PropertySet .....                                    | 38 |
| 7                     | Modelling of user-defined data .....                           | 39 |
| 7.1                   | General .....                                                  | 39 |
| 7.2                   | User-defined attributes .....                                  | 39 |
| 7.3                   | User-defined InterfaceClasses .....                            | 39 |
| 7.4                   | User-defined RoleClasses .....                                 | 40 |
| 7.5                   | User-defined SystemUnitClasses .....                           | 41 |
| 7.6                   | User-defined InstanceHierarchies .....                         | 41 |
| 8                     | Extended AML concepts .....                                    | 42 |
| 8.1                   | General overview .....                                         | 42 |
| 8.2                   | AML Port object .....                                          | 42 |
| 8.3                   | AML Facet object .....                                         | 43 |
| 8.4                   | AML Group object .....                                         | 43 |
| 8.5                   | AML PropertySet .....                                          | 44 |
| 8.6                   | Support of multiple roles .....                                | 46 |
| 8.7                   | Splitting of AML top-level data into different documents ..... | 47 |
| 8.8                   | Internationalization .....                                     | 47 |
| 8.9                   | Version information of AML objects .....                       | 47 |
| Annex A (informative) | General introduction into the Automation Markup Language ..... | 48 |
| A.1                   | General Automation Markup Language concepts .....              | 48 |
| A.1.1                 | The Automation Markup Language architecture .....              | 48 |
| A.1.2                 | Modelling of plant topology information .....                  | 50 |
| A.1.3                 | Referencing geometry and kinematics information .....          | 51 |
| A.1.4                 | Referencing logic information .....                            | 51 |
| A.1.5                 | Modelling of relations .....                                   | 52 |
| A.2                   | Extended AML concepts and examples .....                       | 55 |
| A.2.1                 | General overview .....                                         | 55 |
| A.2.2                 | AML Port concept .....                                         | 55 |
| A.2.3                 | AML Facet concept .....                                        | 59 |
| A.2.4                 | AML Group concept .....                                        | 61 |
| A.2.5                 | PropertySet concept .....                                      | 65 |
| A.2.6                 | Process-Product-Resource concept .....                         | 68 |
| A.2.7                 | Support of multiple roles .....                                | 76 |
| Annex B (informative) | XML Representation of AML Libraries .....                      | 80 |
| B.1                   | AutomationMLBaseRoleClassLib .....                             | 80 |
| B.2                   | AutomationMLInterfaceClassLib .....                            | 81 |
| Bibliography          | .....                                                          | 82 |

Figure 1 – Overview of the engineering data exchange format AML ..... 9

Figure 2 – AML document version information ..... 9

|                                                                                                   |    |
|---------------------------------------------------------------------------------------------------|----|
| Figure 3 – XML text of the AML source tool information .....                                      | 18 |
| Figure 4 – Object identification example of an AML class.....                                     | 19 |
| Figure 5 – Object identification example of an AML object instance .....                          | 19 |
| Figure 6 – Example of a parent-child-relation between AML objects.....                            | 20 |
| Figure 7 – Example of a parent-child-relation between classes .....                               | 20 |
| Figure 8 – Example of an inheritance relation between two classes .....                           | 21 |
| Figure 9 – Example of a class-instance-relation .....                                             | 22 |
| Figure 10 – Example of a relation as block diagram and as object tree .....                       | 23 |
| Figure 11 – Example relation between the objects “PLC1” and “Rob1” .....                          | 24 |
| Figure 12 – AML basic interface class library .....                                               | 27 |
| Figure 13 – XML description of the AML basic interface class library .....                        | 28 |
| Figure 14 – AML basic role class library.....                                                     | 32 |
| Figure 15 – AutomationMLBaseRoleClassLib.....                                                     | 32 |
| Figure 16 – XML text of the AutomationMLBaseRoleClassLib .....                                    | 33 |
| Figure 17 – Example of a user-defined attribute.....                                              | 39 |
| Figure 18 – Example of a user-defined InterfaceClass in a user-defined<br>InterfaceClassLib ..... | 40 |
| Figure 19 – Example of a user-defined RoleClass in a user-defined RoleClassLib .....              | 41 |
| Figure 20 – Examples for different user-defined SystemUnitClasses .....                           | 41 |
| Figure 21 – Example of a user-defined InstanceHierarchy.....                                      | 42 |
| Figure 22 – AML representation of a user-defined InstanceHierarchy.....                           | 42 |
| Figure 23 – Example illustrating the PropertySet concept .....                                    | 45 |
| Figure 24 – XML text of the PropertySet example .....                                             | 46 |
| Figure A.1 – AML general architecture .....                                                       | 48 |
| Figure A.2 – Plant topology with AML .....                                                        | 50 |
| Figure A.3 – Reference from CAEX to a COLLADA document.....                                       | 51 |
| Figure A.4 – Reference from a CAEX to a PLCopen XML document .....                                | 52 |
| Figure A.5 – Relations in AML.....                                                                | 53 |
| Figure A.6 – XML description of the relations example.....                                        | 54 |
| Figure A.7 – XML text of the SystemUnitClassLib of the relations example .....                    | 54 |
| Figure A.8 – XML text of the InstanceHierarchy of the relations example .....                     | 54 |
| Figure A.9 – Port concept .....                                                                   | 55 |
| Figure A.10 – Example describing the AML Port concept .....                                       | 56 |
| Figure A.11 – XML description of the AML Port concept.....                                        | 57 |
| Figure A.12 – XML text describing the AML Port concept.....                                       | 58 |
| Figure A.13 – Definition of a user-defined AML Port class “myPortClass”.....                      | 58 |
| Figure A.14 – AML Facet example .....                                                             | 60 |
| Figure A.15 – XML text of the AML Facet example.....                                              | 60 |
| Figure A.16 – AML Group example .....                                                             | 61 |
| Figure A.17 – XML text for the AML Group example.....                                             | 62 |
| Figure A.18 – Combination of the Facet and Group concept.....                                     | 63 |
| Figure A.19 – XML text view for the combined Facet-Group example .....                            | 64 |
| Figure A.20 – Generic HMI template “B” visualizing a process variable “Y” of a<br>conveyor.....   | 65 |

|                                                                                                           |    |
|-----------------------------------------------------------------------------------------------------------|----|
| Figure A.21 – Generated HMI result “B” visualizing both conveyors with individual process variables ..... | 65 |
| Figure A.22 – PropertySet example.....                                                                    | 66 |
| Figure A.23 – PropertySet example.....                                                                    | 66 |
| Figure A.24 – XML text for the instance hierarchy .....                                                   | 67 |
| Figure A.25 – PropertySet example AML library as XML code .....                                           | 68 |
| Figure A.26 – Base elements of the Product-Process-Resource concept .....                                 | 69 |
| Figure A.27 – PPRConnector interface .....                                                                | 70 |
| Figure A.28 – Example for the Product-Process-Resource concept.....                                       | 70 |
| Figure A.29 – AML roles required for the Process-Product-Resource concept.....                            | 71 |
| Figure A.30 – Elements of the example.....                                                                | 71 |
| Figure A.31 – Links within the example.....                                                               | 72 |
| Figure A.32 – Links of the resource centric view on the example .....                                     | 73 |
| Figure A.33 – InstanceHierarchy of the example in AML .....                                               | 74 |
| Figure A.34 – InternalElements of the example .....                                                       | 75 |
| Figure A.35 – InternalLinks of the example .....                                                          | 75 |
| Figure A.36 – InstanceHierarchy of the example in XML .....                                               | 76 |
| Figure A.37 – Example of a user-defined instance supporting multiple roles .....                          | 77 |
| Figure A.38 – XML text of the AML representation of multiple role support.....                            | 78 |
| Figure A.39 – AML Role class library corresponding to the multiple role definition example.....           | 78 |
| Figure A.40 – XML text of the AML role class library .....                                                | 79 |
| Table 1 – Abbreviations .....                                                                             | 14 |
| Table 2 – Meta information about the AML source tool.....                                                 | 17 |
| Table 3 – Interface classes of the AutomationMLInterfaceClassLib .....                                    | 26 |
| Table 4 – InterfaceClass AutomationMLBaseInterface .....                                                  | 28 |
| Table 5 – InterfaceClass Order .....                                                                      | 28 |
| Table 6 – InterfaceClass PortConnector.....                                                               | 29 |
| Table 7 – InterfaceClass PPRConnector .....                                                               | 29 |
| Table 8 – InterfaceClass ExternalDataConnector .....                                                      | 29 |
| Table 9 – InterfaceClass COLLADAInterface .....                                                           | 30 |
| Table 10 – InterfaceClass PLCopenXMLInterface .....                                                       | 30 |
| Table 11 – InterfaceClass Communication .....                                                             | 31 |
| Table 12 – InterfaceClass SignalInterface.....                                                            | 31 |
| Table 13 – RoleClass AutomationMLBaseRole .....                                                           | 33 |
| Table 14 – RoleClass Group .....                                                                          | 34 |
| Table 15 – RoleClass Facet.....                                                                           | 34 |
| Table 16 – Optional attributes for AML Port objects .....                                                 | 35 |
| Table 17 – Sub-attributes of the attribute “Cardinality” .....                                            | 35 |
| Table 18 – Interface of the AML Port class.....                                                           | 36 |
| Table 19 – RoleClass Resource.....                                                                        | 36 |
| Table 20 – RoleClass Product.....                                                                         | 36 |
| Table 21 – RoleClass Process .....                                                                        | 37 |

|                                                           |    |
|-----------------------------------------------------------|----|
| Table 22 – RoleClass Structure .....                      | 37 |
| Table 23 – RoleClass ProductStructure .....               | 37 |
| Table 24 – RoleClass ProcessStructure .....               | 38 |
| Table 25 – RoleClass ResourceStructure .....              | 38 |
| Table 26 – RoleClass PropertySet .....                    | 38 |
| Table A.1 – Overview of major extended AML concepts ..... | 55 |

## INTRODUCTION

IEC 62714 is a solution for data exchange focusing on the domain of automation engineering.

The data exchange format defined in the IEC 62714 series (Automation Markup Language, AML) is an XML schema based data format and has been developed in order to support the data exchange in a heterogeneous engineering tools landscape.

The goal of AML is to interconnect engineering tools 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. Therefore, an important focus in the data exchange in engineering is the exchange of object oriented data structures, geometry, kinematics and logic.

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 an “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, kinematics and logic information.

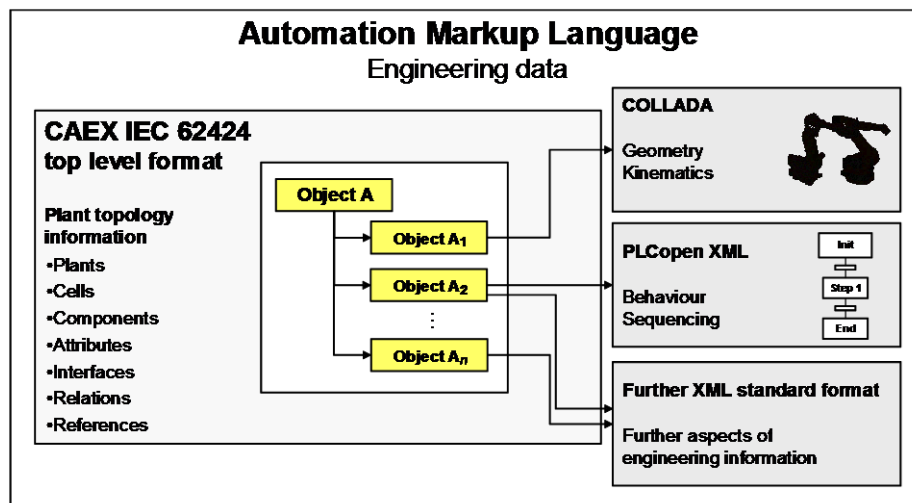


Figure 1 – Overview of the engineering data exchange format AML

Due to the different aspects of AML, the IEC 62714 series 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. It is the basis of all future parts, and it provides mechanisms to reference other sub formats.

- IEC 62714-2: Role class libraries  
This part is intended to specify additional AML libraries.
- IEC 62714-3: Geometry and kinematics  
This part is intended to specify the modelling of geometry and kinematics information.
- IEC 62714-4: Logic  
This part is intended to specify the modelling of logics, sequencing, behaviour and control related information.

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

As long as no further parts describe the integration of further standards, it is important to focus on a limited set of sub data formats. Otherwise it would open up the usage of any data format and data exchange would not work.

Annex A gives an informative introduction, use cases and examples regarding AML.

Annex B gives an informative XML representation of the libraries defined in this part of IEC 62714.

# ENGINEERING DATA EXCHANGE FORMAT FOR USE IN INDUSTRIAL AUTOMATION SYSTEMS ENGINEERING – AUTOMATION MARKUP LANGUAGE –

## Part 1: Architecture and general requirements

### 1 Scope

This part of IEC 62714 specifies general requirements and the architecture of AML for the modelling of engineering information which is exchanged between engineering tools for industrial automation and control systems. Its provisions apply to the export/import applications of related tools.

This part of IEC 62714 does not define details of the data exchange procedure or implementation requirements for the import/export tools.

### 2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 62424:2008, *Representation of process control engineering – Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools*

IEC 62714 (all parts), *Engineering data exchange format for use in industrial automation systems engineering – Automation Markup Language*

ISO/IEC 9834-8, *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components*

ISO/PAS 17506, *Industrial automation systems and integration — COLLADA digital asset schema specification for 3D visualization of industrial data*

COLLADA 1.4.1:March 2008, COLLADA – Digital Asset Schema Release 1.4.1  
(available at <[http://www.khronos.org/files/collada\\_spec\\_1\\_4.pdf](http://www.khronos.org/files/collada_spec_1_4.pdf)>)

Extensible Markup Language (XML) 1.0 1.0:2004, W3C Recommendation  
(available at <<http://www.w3.org/TR/2004/REC-xml-20040204/>>)

PLCopen XML 2.0:December 3rd 2008 and PLCopen XML 2.0.1:May 8th 2009, XML formats for IEC 61131-3  
(available at <<http://www.plcopen.org/>>)

### 3 Terms, definitions and abbreviations

#### 3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

**3.1.1****AML**

XML based data exchange format for plant engineering data following IEC 62714

**3.1.2****automation object**

physical or logical entity in the automated system

Note 1 to entry: An example of an automation object is an automation component, a valve or a signal.

**3.1.3****AML object**

data representation of an automation object with a relation to an AML role class

Note 1 to entry: The AML objects are the core elements of AML. They represent instances and may contain administration items, attributes, interfaces, relations and references.

**3.1.4****AML class**

predefined AML object type

Note 1 to entry: AML classes are stored within AML libraries.

Note 2 to entry: AML classes define reusable sample solutions, characterized by attributes, interfaces and aggregated objects.

Note 3 to entry: AML classes can be used for multiple instantiations.

**3.1.5****AML attribute**

property which belongs to an AML object

Note 1 to entry: AML attributes are described as an XML element corresponding to IEC 62424:2008, A.2.4.

**3.1.6****AML document**

certain CAEX document following IEC 62714 including all referenced sub documents

Note 1 to entry: AML documents may be stored as files, but also e.g. as string or data streams.

**3.1.7****AML file**

certain CAEX file following IEC 62714-1 with the extension .aml excluding all referenced sub files

**3.1.8****AML interface**

single connection point that belongs to an AML object and can be linked with another interface

Note 1 to entry: Interfaces allow the description of relations between objects by the definition of CAEX Internal-Links. Examples are a signal interface, a device interface or a power interface.

**3.1.9****AML library**

library containing AML classes

**3.1.10****AML Port**

AML object that represents a container for a group of interfaces characterized by additional properties

Note 1 to entry: Ports belong to a parent AML object and describe complex interfaces of this object. Ports can be connected to each other on a higher abstraction level.

### 3.1.11

#### **AML Group**

AML object providing a certain view on AML objects

### 3.1.12

#### **AML Facet**

AML object providing a certain view on AML attributes or interfaces of one AML object

### 3.1.13

#### **CAEX**

neutral XML based data format

Note 1 to entry: CAEX is a neutral data format according to IEC 62424:2008, Clause 7, Annex A and Annex C

### 3.1.14

#### **copy-instance-relation**

relation between the instance and the corresponding class where the instance is created by copying the class data structures

Note 1 to entry: The instance receives a copy of all features and properties of the source AML class. Modifications of the class do not lead to modifications of the instance. Within the instance, class properties are individualized. Further copies are possible due to the knowledge of the source AML class.

### 3.1.15

#### **universal unique identifier**

#### **UUID**

unique identifier for AML objects

Note 1 to entry: This note applies to the French language only.

### 3.1.16

#### **global unique identifier**

#### **GUID**

implementation of a UUID

Note 1 to entry: Real GUID example: "{AC76BA86-7AD7-1033-7B44-A70000000000}".

Note 2 to entry: In IEC 62714, GUIDs are also presented in a short form such as "GUID1", "GUID2" etc. This serves the readability and acts as a real GUID.

Note 3 to entry: This note applies to the French language only.

### 3.1.17

#### **inheritance relation**

relation between two AML classes

Note 1 to entry: The derived class inherits all attributes and features of the parent class.

### 3.1.18

#### **instance**

data representation of an individual physical or logical item

Note 1 to entry: Instances can be extended, e.g. by aggregated objects or attributes.

### 3.1.19

#### **PropertySet**

AML standard role class containing a set of semantically predefined attributes

**3.1.20****topology**

hierarchical structure of a system, visualizable as object tree

Note 1 to entry: Multiple hierarchies, crossed structures and object networks are included.

**3.1.21****plant topology**

hierarchical structure of a plant, visualizable as object tree

**3.1.22****publish, verb**

to model a data structure of an external document for usage within CAEX

Note 1 to entry: This allows definition of relations between data structures of independent external documents.

**3.1.23****relation**

association between CAEX objects

Note 1 to entry: Examples for relations are parent-child-relations and class-instance-relations.

**3.1.24****link**

connection between objects of type CAEX ExternalInterface

Note 1 to entry: A link is modelled by means of CAEX InternalLink.

**3.1.25****reference**

association between a CAEX InternalElement and externally stored information

**3.2 Abbreviations****Table 1 – Abbreviations**

|         |                                     |
|---------|-------------------------------------|
| AML     | Automation Markup Language          |
| CAE     | Computer Aided Engineering          |
| CAEX    | Computer Aided Engineering eXchange |
| COLLADA | Collaborative design activity       |
| GUID    | Global unique identifier            |
| HMI     | Human machine interface             |
| ID      | Identifier                          |
| MES     | Manufacturing execution system      |
| PLC     | Programmable logic controller       |
| URL     | Uniform resource locator            |
| URI     | Uniform resource identifier         |
| UUID    | Universal unique identifier         |
| XML     | Extensible Markup Language          |

**4 Conformity**

To claim conformity to this part of IEC 62714 with respect to the support of AML, the requirements of Clauses 5, 6, 7 and 8 shall be fulfilled.

## 5 AML architecture specification

### 5.1 General

The centre of AML is the top-level data format CAEX, a neutral data format according to IEC 62424:2008, Clause 7, Annex A and Annex C, that interconnects established data formats for the engineering aspects for topology, geometry, kinematics, behaviour and sequencing information. Therefore, a basic characteristic of AML is an inherent distributed document architecture focussing on the above mentioned engineering aspects.

Figures are illustrative only. The graphical representation is not normative.

### 5.2 General AML architecture

Regarding the general AML architecture, the following provisions apply:

**Plant topology information:** The plant topology acts as the top-level data structure of the plant engineering information and shall be modelled by means of the data format CAEX according to IEC 62424:2008, Clause 7, Annex A and Annex C. Semantic extensions of CAEX are described separately. Multiple and crossed hierarchy structures shall be used by means of the mirror object concept according to IEC 62424:2008, A.2.14. Mirror objects shall not be modified; all changes shall be done at the master object.

NOTE 1 According to IEC 62424:2008, A.2.14, an AML object with a relation to another AML object is called "mirror object" whereas the related AML object is called "master object". The mirror object is considered to be identical to the master object. This enables placing one object instance into different plant hierarchies and thus allows modelling of complex object networks with crossed structures.

NOTE 2 IEC 62714 does not syntactically modify the CAEX data format. An informative overview and additional examples regarding the plant topology are provided in A.1.2 and in IEC 62424:2008, Annex D.

**Reference and relation information:** References and relations shall be stored according to 5.6 and 5.7. Relations between externally stored information shall be stored with CAEX mechanisms. If required, the related link partners shall be published in the CAEX plant topology description by means of CAEX ExternalInterfaces. They shall be derived from AML standard interface classes specified in 6.3.

NOTE 3 References depict links from CAEX objects to externally stored information. An informative overview about relations is provided in A.1.5. References and publishing of interfaces are described in additional parts of IEC 62714.

NOTE 4 Relations depict associations between CAEX objects.

**Geometry and kinematics information:** Geometry and kinematics relevant information shall be stored using the data format COLLADA™<sup>1</sup>. COLLADA interfaces that need to be interconnected within the top level format shall be published as CAEX ExternalInterfaces.

NOTE 5 IEC 62714 does not syntactically modify the COLLADA data format. An overview example of how to reference COLLADA is provided in A.1.3. Details are intended to be specified in IEC 62714-3.

NOTE 6 By means of the COLLADA geometry information of different objects, a complete scene can be derived automatically. These files can be referenced from CAEX and can be interlinked using CAEX linking mechanisms.

**Logic information:** Logic information shall be stored using the data format PLCopen XML. If logic items, e.g. variables or signals, need to be interconnected within the top level format, they shall be published as CAEX ExternalInterfaces. All items of PLCopen XML that are published within the top level format shall have a unique ID within PLCopen XML.

---

<sup>1</sup> COLLADA is the trademark of a product supplied by the Khronos Group. This information is given for the convenience of users of this standard and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

NOTE 7 Logic information describes sequences of actions and the internal behaviour of objects including I/O connections and logical variables. IEC 62714 does not modify the PLCopen XML format. An informative overview of how to reference logic information is provided in A.1.4. Details are intended to be specified in IEC 62714-4.

**Referencing other data formats:** IEC 62714 may be extended in the future by additional parts specifying the integration of further XML data formats utilizing the AML reference mechanisms. Details may be defined in additional parts of IEC 62714.

The data format AML does not provide consistency checks of constraints, attribute values, relations, references or the semantic correctness of the contained data: this is the responsibility of the source or target tool or the corresponding import/export application. AML only allows a syntactical proof of the document against the corresponding schemas.

### 5.3 AML document versions

Each AML XML document shall store the underlying AML version which this standard follows.

NOTE 1 Normative provisions regarding the version information related to AML object instances are defined in 8.9. The storage of tool specific meta information is defined in 5.4.

Hence, the following provisions apply:

- The CAEX root element “CAEXFile” of each AML top level document shall have the CAEX child element “AdditionalInformation”.
- The element “AdditionalInformation” shall have an attribute “AutomationMLVersion”.
- The value of this attribute “AutomationMLVersion” shall be stored in the XML document. It shall be “2.0” to correspond to this standard.
- Every referenced CAEX document shall follow the same AML version of the root document. Mixing of documents with different AML versions is explicitly forbidden.
- Every referenced external document shall also follow the named schema versions specified in the above AML version specification. Mixing of external document versions outside of one AML version specification is explicitly forbidden.

Figure 2 illustrates the XML text for a CAEX document following the AML version 2.0.

```
<CAEXFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="
CAEX_ClassModel.xsd" FileName="AutomationMLStandardLibrary2010-01-14_v1.99.aml" SchemaVersion="2.15">
  <AdditionalInformation AutomationMLVersion="2.0"/>
</CAEXFile>
```

**Figure 2 – AML document version information**

- Every AML standard library and every user defined AML library shall define its version number utilizing the CAEX element “Version”. The syntax of the value of the version number is not defined in this part of IEC 62714.
  - If required, CAEX classes shall define their version number utilizing the CAEX element “Version”. The syntax and semantic of the version number of classes within an AML library is not defined in this part of IEC 62714.
  - Same libraries of different versions are forbidden to be stored in the same AML file.
- NOTE 2 This ensures the uniqueness of AML library names within an AML file.
- The creator of an AML document shall ensure that only version compatible classes and external documents are referenced.

IEC 62714 is based on the following document formats:

- CAEX version 2.15;
- PLCopenXML 2.0 and 2.0.1;
- COLLADA 1.5.0 as specified in ISO/PAS 17506 and COLLADA 1.4.1;

- AML standard libraries as specified in this part of IEC 62714 and further parts of IEC 62714.

#### 5.4 Meta information about the AML source tool

In case of the need of a transfer of user defined data from a source tool to a destination tool, it is necessary to store information about the source tool directly into the AML document. Hence, the following provisions apply:

- Each AML document shall provide information about the tool which has written the AML document.
- In a data exchange tool chain, all participating tools shall store this information in the CAEX 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 as part of the CAEX AdditionalInformation of the root object of the CAEX document.
- The AdditionalInformation block shall be named "WriterHeader".
- The meta information shall provide information about:
  - the name of the exporting software, the writer tool;
  - the ID of the writer tool (it shall remain unchanged);
  - the vendor of the writer tool;
  - the URL of the writer tool;
  - the product version of the writer tool;
  - the product release number of the writer tool;
  - the last writing time of the writer;
  - the project title of the source project;
  - the project ID of the source project.
- The content of the meta information shall be defined by the writer tool and shall be of type xs:string.
- The required information shall be stored by means of the attributes shown in Table 2.

**Table 2 – Meta information about the AML source tool**

| XML tag name        | Type        | Level     | Example                       |
|---------------------|-------------|-----------|-------------------------------|
| WriterName          | xs:string   | Mandatory | "ToolX AML Exporter"          |
| WriterID            | xs:string   | Mandatory | "ToolXToAML123"               |
| WriterVendor        | xs:string   | Mandatory | "ToolX Vendor"                |
| WriterVendorURL     | xs:string   | Mandatory | "http://www.ToolX-Vendor.org" |
| WriterVersion       | xs:string   | Mandatory | "0.2"                         |
| WriterRelease       | xs:string   | Mandatory | "123 prealpha"                |
| LastWritingDateTime | xs:DateTime | Mandatory | "2011-05-25T09:30:47"         |
| WriterProjectTitle  | xs:string   | Optional  | "eCarproduction"              |
| WriterProjectID     | xs:string   | Optional  | "eCarproduction_LinePLC.prj"  |

For the XML representation of the meta information, the following provisions apply:

- The element "WriterHeader" shall be a child XML element of the CAEX element AdditionalInformation of the CAEX root element.

- Each meta information named in Table 2 shall be described as a child XML element of the “WriterHeader”.
- Multiple meta information of the same name are forbidden in the same “WriterHeader” element.
- The order of the meta information shall be equivalent to Table 2.

Figure 3 illustrates the required XML text by means of an example.

```
<AdditionalInformation>
  <WriterHeader>
    <WriterName>ToolX AML Exporter</WriterName>
    <ToolWriterID>ToolXToAML123</ToolWriterID>
    <WriterVendor>ToolX Vendor</WriterVendor>
    <WriterVendorURL>http://www.ToolX-Vendor.org</WriterVendorURL>
    <WriterVersion>0.1</WriterVersion>
    <WriterRelease>123 prealpha</WriterRelease>
    <LastWritingDateTime>2013-12-31T12:00:00</LastWritingDateTime>
    <WriterProjectTitle>eCarproduction</WriterProjectTitle>
    <WriterProjectID>eCarproduction_LinePLC.prj</WriterProjectID>
  </WriterHeader>
</AdditionalInformation>
```

**Figure 3 – XML text of the AML source tool information**

## 5.5 Object identification

AML follows the object oriented paradigm. All engineering information is modelled as object or belongs to an object. But, in a heterogeneous tool landscape, different engineering tools use different concepts for the identification of objects, e.g. a unique name, a unique identifier or a unique path. Some tools allow changes of the identifiers over the life time, others do not. IEC 62714 enables the data exchange between different engineering tools with such individual object identification concepts. Owing to the described characteristics, this part of IEC 62714 neutralizes this variety and defines one mandatory object identification concept.

Regarding the object identification, the following provisions apply:

- According to IEC 62424:2008, A.2.2.1, AML classes (RoleClasses, InterfaceClasses and SystemUnitClasses) shall be identified by their CAEX tag “Name”.
- This name shall be unique within the hierarchy level of the corresponding AML library over the life time of the class.
- According to IEC 62424:2008, A.2.8, referencing of classes shall be done via full paths using the corresponding path separators.
- All AML object instances (CAEX InternalElements and CAEX ExternalInterfaces) shall be identified by their CAEX tag “ID”. This identifier shall be a universal unique identifier (UUID) according to ISO/IEC 9834-8.

NOTE 1 A possible implementation of the UUID is the global unique identifier (GUID).

NOTE 2 According to IEC 62424:2008, A.3.15, the tag “ID” is not mandatory in contrast to this part of IEC 62714.

NOTE 3 In this part of IEC 62714, UUIDs are presented in a short form such as “GUID1”, “GUID2” etc.

NOTE 4 The CAEX tag “Name” is a display name; it has informative character only and can change over the time or tool.

- Once created, this UUID shall never change over the life time of the corresponding object within all participating tools.
- Referencing instances shall use the “ID” value.
- Referencing CAEX interfaces shall be done using the corresponding UUID of the interface’s parent object followed by the separator string “.” and the name of the interface instance.

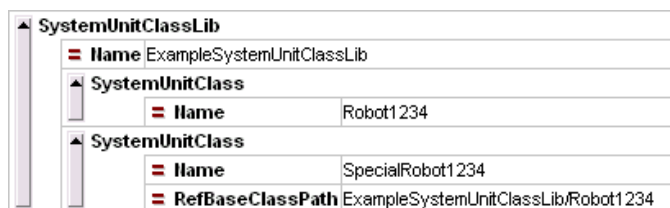
EXAMPLE 1 “GUID:out”.

- Referencing CAEX attributes shall be done using the corresponding UUID of the attribute's parent object followed by the separator string “.” and the name of the attribute. If the attribute is a nested attribute, the separator string is followed by the sub-path of the attribute.

EXAMPLE 2 “GUID.Colour”.

EXAMPLE 3 “GUID.Colour.red”.

Figure 4 gives an example of a SystemUnitClassLib with the SystemUnitClass “Robot1234” and another SystemUnitClass “SpecialRobot1234” derived from “Robot1234”.



```
<SystemUnitClassLib Name="ExampleSystemUnitClassLib">
  <SystemUnitClass Name="Robot1234"/>
  <SystemUnitClass Name="SpecialRobot1234" RefBaseClassPath="ExampleSystemUnitClassLib/Robot1234"/>
</SystemUnitClassLib>
```

Figure 4 – Object identification example of an AML class

Figure 5 gives an example of an InstanceHierarchy with one object “RB\_100” which has a unique ID represented by the string “GUID1”.



```
<InstanceHierarchy Name="ExampleProject">
  <InternalElement Name="RB_100" ID="GUID1"/>
</InstanceHierarchy>
```

Figure 5 – Object identification example of an AML object instance

## 5.6 AML relations specification

### 5.6.1 General

The focus on objects makes it necessary to define a mechanism to set objects in association to each other. This part of IEC 62714 distinguishes between two mechanisms to store this information: references and relations. Subclause 5.6 focuses on relations, whereas 5.7 focuses on references. An informative overview about relations and references is provided in A.1.5.

### 5.6.2 Parent-child-relations between AML objects

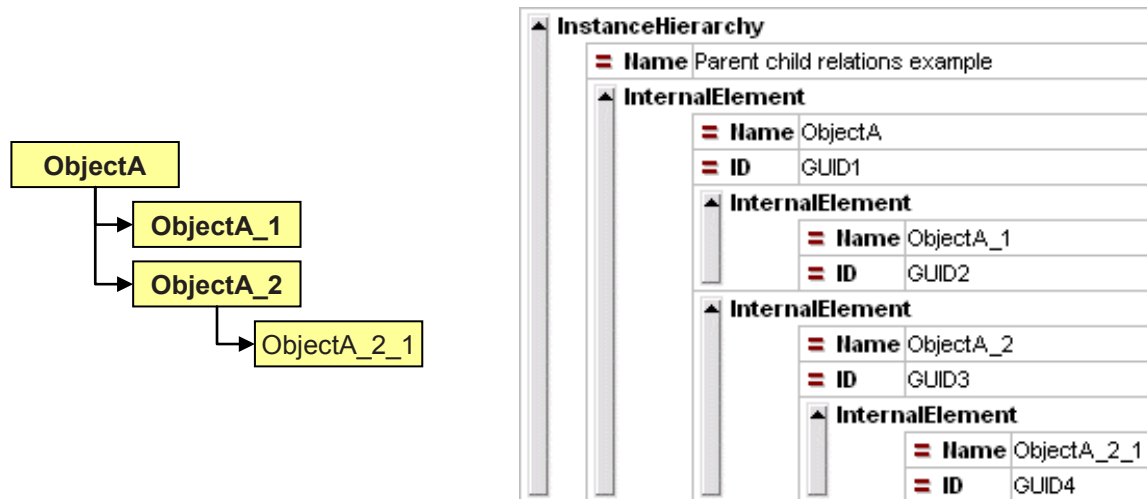
Parent-child-relations between object instances are used to represent hierarchical object structures and describe a “consist-of-relation”.

Regarding parent-child-relations between AML objects, the following provision applies:

- The storage of hierarchies shall be done according to IEC 62424:2008, Annex A, e.g. A.2.11.

NOTE In addition to simple hierarchies, also crossed hierarchies (object networks) can be stored according to IEC 62424:2008, A.2.14.

Figure 6 gives an example of an object hierarchy and its storage.



```
<InstanceHierarchy Name="Parent child relations example">
  <InternalElement Name="ObjectA" ID="GUID1">
    <InternalElement Name="ObjectA_1" ID="GUID2"/>
    <InternalElement Name="ObjectA_2" ID="GUID3">
      <InternalElement Name="ObjectA_2_1" ID="GUID4"/>
    </InternalElement>
  </InternalElement>
</InstanceHierarchy>
```

Figure 6 – Example of a parent-child-relation between AML objects

### 5.6.3 Parent-child-relations between AML classes

Regarding parent-child-relations between AML classes, the following provisions apply:

- A parent-child-relation between AML classes shall describe their hierarchical neighbour ship only. This allows definition of any user-defined hierarchical structure.
- This relation has no further semantics.

NOTE A parent-child-relation does not imply an inheritance relation. Inheritance relations are modelled explicitly as described in 5.6.4.

Figure 7 gives an example of a parent-child-relation between the classes “ParentClass” and “ChildClass”. The “ChildClass” does not have an inheritance relation to its parent.

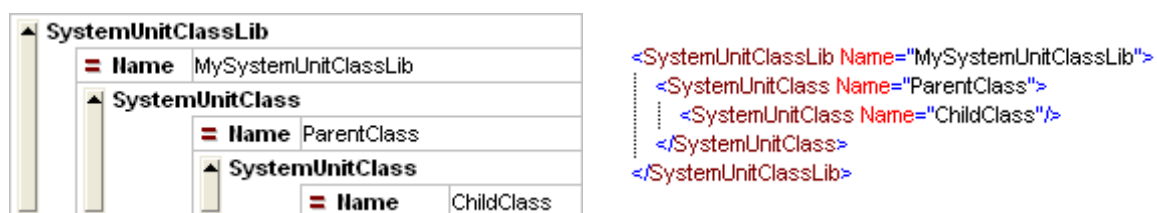


Figure 7 – Example of a parent-child-relation between classes

#### 5.6.4 Inheritance relations

Regarding inheritance relations, the following provisions apply:

- Inheritance between classes shall be defined according to IEC 62424:2008, A.2.7.
- If inheritance is required, the parent class shall be specified using the CAEX tag “RefBaseClassPath” comprising the full path of the class according to IEC 62424:2008, A.2.7.
- If the desired parent class is placed one hierarchy level above the child class, the parent class can be specified by storing the name of the parent class in the CAEX tag “RefBaseClassPath” without providing the full path.

Figure 8 gives an example of the class “Robot1234” and another class “SpecialRobot1234” which inherits from “Robot1234”.

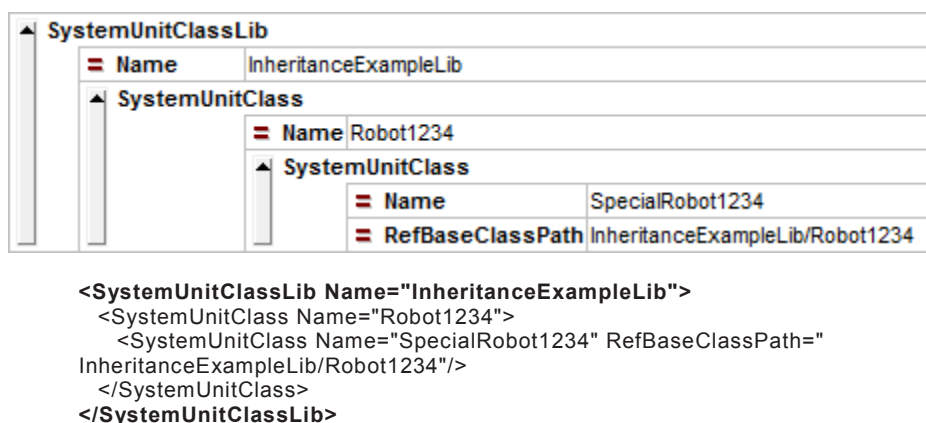


**Figure 8 – Example of an inheritance relation between two classes**

In addition to this example, the CAEX tag “RefBaseClassPath” can either be “InheritanceExampleLib/Robot1234” as well as “Robot1234” since the parent class is one hierarchy level above the class “SpecialRobot1234”.

#### 5.6.5 Class-instance-relations

Instances are characterized by a unique identifier and parameter set. The following provisions apply:

- An AML object shall be modelled as CAEX InternalElement as part of the CAEX InstanceHierarchy or of a SystemUnitClass.
- An AML object may be a singleton without a relation to any SystemUnitClass.

NOTE 1 However, an AML object has a relation to a standard AML role class.

NOTE 2 Instances without a relation to the AutomationMLBaseRole are possible but are user defined objects. They are not AML objects.

- If an AML object has a class-instance-relation to a SystemUnitClass, it shall be created as a copy of this SystemUnitClass including the internal architecture of the class and all inherited information.

NOTE 3 A SystemUnitClass serves as a template in this way. Changes in the SystemUnitClass are not automatically reflected in the corresponding Automation objects. Furthermore, the Automation object can be transported without the class information; it contains within itself the whole belonging information.

- The extension or reduction of instance data compared to the source class is allowed.
- The copied source class shall be indicated in the CAEX tag “RefBaseSystemUnitPath” of the instance for further usage. This tag shall comprise the full path and name of the source class.

NOTE 5 If the source-class of an instance changes, this does not entail a change of the instance. The update of instances is a possible tool functionality out of the scope of this part of IEC 62714.

- Changes of a source class should lead to a new version of the class with another name. Within the new class, the full path of the old version of the class should be stored in the CAEX tag “OldVersion”.

NOTE 6 This provision supports tracking of changes across different versions of a class.

- Inheritance between a SystemUnitClass and an object instance is not allowed.

NOTE 7 An instance can only be a copy of its class. This is a restriction against IEC 62424:2008, A.2.7. Inheritance between classes and instances can be part of future extension.

- The relation between an instance and a RoleClass shall be indicated according to IEC 62424:2008, A.2.7, by the attribute “RefBaseRoleClassPath” of the belonging RoleRequirement. In contrast to IEC 62424:2008, A.2.7, no inheritance is permitted; all RoleClass specifications shall be copied to the corresponding AML object.
- The relation between a CAEX ExternalInterface and an InterfaceClass shall be indicated according to IEC 62424:2008, A.2.7. In contrast to IEC 62424, no inheritance is allowed; all InterfaceClass specifications shall be copied to the corresponding AML object.

Figure 9 gives an example of a class-instance-relation between the object “ObjectA” and a user-defined SystemUnitClass “generic\_Valve”.

| InstanceHierarchy     |                                    |
|-----------------------|------------------------------------|
| Name                  | ClassInstanceRelation Example      |
| InternalElement       |                                    |
| Name                  | ObjectA                            |
| ID                    | GUID1                              |
| RefBaseSystemUnitPath | mySystemUnitClassLib/generic_Valve |

```
<InstanceHierarchy Name="ClassInstanceRelation Example">
... <InternalElement Name="ObjectA" ID="GUID1" RefBaseSystemUnitPath="mySystemUnitClassLib/generic_Valve"/>
</InstanceHierarchy>
```

**Figure 9 – Example of a class-instance-relation**

In addition to the standard class-instance-relation provisions, the following specific provision applies according to the CAEX mirror concept:

- The tag “RefBaseSystemUnitPath” may indicate another object instance instead of a SystemUnitClass according to the mirror concept of IEC 62424:2008, A.2.14.

### 5.6.6 Instance-instance-relations

Instance-instance-relations are relations between two interfaces of arbitrary AML objects.

Regarding instance-instance-relations, the following provisions apply:

- Instance-instance-relations shall be stored according to IEC 62424:2008, A.2.5.3 and A.2.14, by means of the CAEX InternalLink functionality.
- CAEX InternalLinks should be stored at the CAEX InternalElement which is the lowest common parent of the corresponding connected CAEX objects.
- Instance-instance-relations shall be defined only between CAEX ExternalInterfaces that belong to the corresponding AML objects. This is according to IEC 62424:2008, A.2.3.1.
- The ExternalInterfaces should be derived directly or indirectly from one of the AML standard interface classes.

NOTE 1 The AML standard interface class library is specified in 6.3. The interface classes define the semantic of the interface and thus the semantic of the link. A link between interfaces without a reference to an interface class has no semantics.

- COLLADA documents may be interlinked. The corresponding COLLADA interfaces may be any items that have a valid URI. If those nodes are required to be interlinked in CAEX, they shall be published in CAEX by adding a CAEX ExternalInterface to the corresponding object. This ExternalInterface shall be derived from the AML standard interface class “COLLADAInterface” or one of its derivatives.

NOTE 2 The standard interface class “COLLADAInterface” is specified in 6.3.7. Details are intended to be specified in IEC 62714-3.

- PLCopen XML documents may be interlinked by utilizing corresponding PLCopen XML interfaces. If PLCopen XML items are required to be interlinked in CAEX, they shall be published by adding a CAEX ExternalInterface to the corresponding object. This ExternalInterface shall be derived from the AML standard interface class “PLCopenXMLInterface” or one of its derivatives.

NOTE 3 The standard interface class “PLCopenXMLInterface” is specified in 6.3.8. Details are intended to be specified in IEC 62714-4.

Figure 10a) describes an example comprising a robot “Rob1” and a PLC “PLC1”, each with one signal interface that are connected. Figure 10b) depicts this example as an object hierarchy.

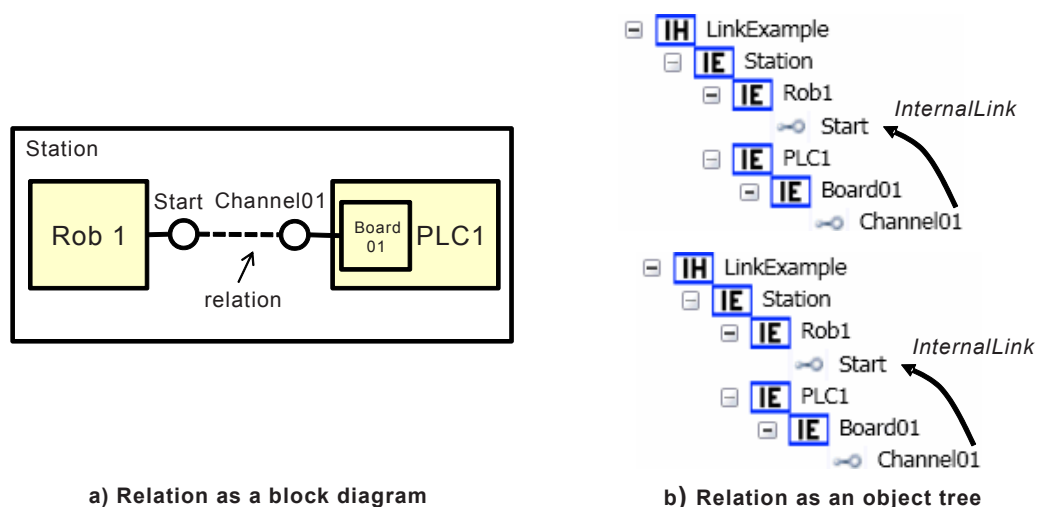
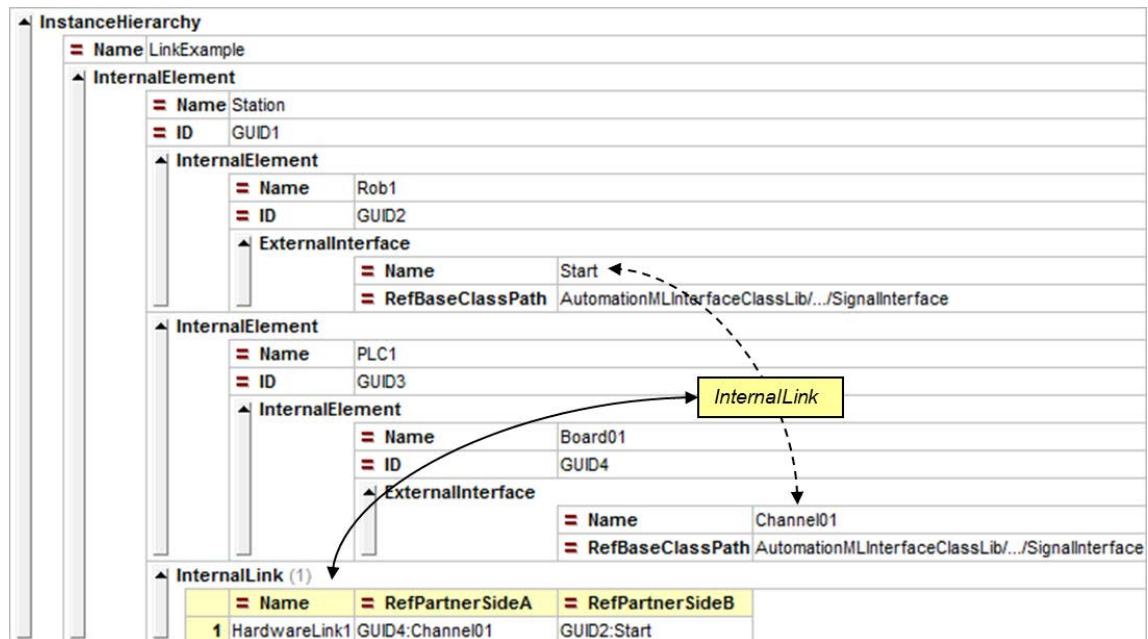


**Figure 10 – Example of a relation as block diagram and as object tree**

Figure 11 depicts the AML representation of the given example. The full XML text for the InstanceHierarchy for this example comprising all AML objects “Station”, “Rob1”, “PLC1” and “Board01” including their interfaces is shown below.

NOTE 4 The path strings given in this example are reduced with “/.../” in order to increase the readability.



```
<InstanceHierarchy Name="LinkExample">
  <InternalElement Name="Station" ID="GUID1">
    <InternalElement Name="Rob1" ID="GUID2">
      <ExternalInterface Name="Start" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface">
        <Attribute Name="Type">
          <Value>digital</Value>
        </Attribute>
        <Attribute Name="Direction">
          <Value>In</Value>
        </Attribute>
      </ExternalInterface>
    </InternalElement>
    <InternalElement Name="PLC1" ID="GUID3">
      <InternalElement Name="Board01" ID="GUID4">
        <ExternalInterface Name="Channel01" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface">
          <Attribute Name="Type">
            <Value>digital</Value>
          </Attribute>
          <Attribute Name="Direction">
            <Value>Out</Value>
          </Attribute>
        </ExternalInterface>
      </InternalElement>
    </InternalElement>
    <InternalLink Name="HardwareLink1" RefPartnerSideA="GUID4:Channel01" RefPartnerSideB="GUID2:Start"/>
  </InternalElement>
</InstanceHierarchy>
```

```
<InstanceHierarchy Name="LinkExample">
  <InternalElement Name="Station" ID="GUID1">
    <InternalElement Name="Rob1" ID="GUID2">
      <ExternalInterface Name="Start" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface"/>
    </InternalElement>
    <InternalElement Name="PLC1" ID="GUID3">
      <InternalElement Name="Board01" ID="GUID4">
        <ExternalInterface Name="Channel01" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface"/>
      </InternalElement>
    </InternalElement>
    <InternalLink Name="HardwareLink1" RefPartnerSideA="GUID4:Channel01" RefPartnerSideB="GUID2:Start"/>
  </InternalElement>
</InstanceHierarchy>
```

Figure 11 – Example relation between the objects “PLC1” and “Rob1”