Dezember 2016

DIN SPEC 16592

ICS 25.040.01; 35.240.50

Kombinieren von OPC Unified Architecture und Automation Markup Language

Combining OPC Unified Architecture and Automation Markup Language

Combinaison de l'architecture unifiée OPC et l'Automation Markup Language

Zur Erstellung einer DIN SPEC können verschiedene Verfahrensweisen herangezogen werden:
Das vorliegende Dokument wurde nach den Verfahrensregeln einer PAS erstellt.

Gesamtumfang 68 Seiten

Dieses Dokument wurde durch die im Vorwort genannten Verfasser erarbeitet und verabschiedet.

Preisgruppe 23
www.din.de
www.beuth.de

2596035

Contents

Page

Foreword

This DIN SPEC has been developed according to the PAS (Publicly Available Specification) procedure. DIN SPECs developed according to the PAS procedure are prepared in workshops that do not necessarily include all stakeholders.

This DIN SPEC (PAS) is the result of the INS (Innovation with Norms and Standards) project "Plug-and-Work – Universelle Schnittstellen für die Automatisierung" funded by the German Federal Ministry for Economic Affairs and Energy (BMWi).

The following initiators and authors developed and approved this document:

— Fraunhofer-Institut IOSB
Dr. Miriam Schleipen

— Fraunhofer-Institut IOSB
Robert Henßen

— Softing Industrial Automation GmbH
Hans Endl

— Enterprise Information Systems (EIS), Rheinische Friedrich-Wilhelms-Universität Bonn
Irlan Grangel-González

— Technische Universität München
Veit Hammerstingl

— Fraunhofer-Institut IPA
Günther Hörcher

— Fraunhofer-Institut IPA
Olha Meyer

— Fraunhofer-Institut IPK
Sebastian Neumeyer

— Otto-v.-Guericke Universität
Arndt Lüder

— Otto-v.-Guericke Universität
Nicole Schmidt

— Fraunhofer IOSB-INA Kompetenzzentrum Industrial
Florian Pethig

— ESR Pollmeier
Stefan Pollmeier

— CAN in Automation (CiA)
Thilo Schumann

— Lehrstuhl für Werkzeugmaschinen Werkzeugmaschinenlabor WZL der RWTH Aachen
Simon Storms

— ProSTEP iViP e.V.
Steven Vettermann

— DFKI GmbH
Stephan Weyer

— TU Dresden
Sachari Wassilew

There are currently no standards covering this subject. DIN SPECs are not part of the body of German standards. No draft of this DIN SPEC has been published.

Despite great effort to ensure that technical and non-technical descriptions are correct, reliable and precise, the workshop can make no express or implied warranty for the correctness of the document. This document is used on the understanding that the workshop shall not be held liable for damage or loss of any kind. Use of this DIN SPEC does not exempt users from being responsible for their own actions and they therefore use this DIN SPEC at their own risk.

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

1 Scope

This specification specifies the combination of AML engineering data with OPC Unified Architecture online information such as process data and diagnostic information.

The AML data format, developed by AutomationML e.V. has been standardized in the IEC 62714 series of standards. This format is an XML-based data exchange format which enables the transfer of engineering data for manufacturing systems within a heterogeneous engineering tool landscape.

The OPC Unified Architecture developed by the OPC Foundation is standardised in IEC 62541. It defines platform independent communication mechanisms for online data exchange and generic, extensible, and object-oriented modelling capabilities for the information a system wants to expose.

The combination of AML and OPC Unified Architecture is divided into two parts:

1. Providing an AML model via OPC Unified Architecture; this involves the AML integration into OPC Unified Architecture. The goal is to communicate, exchange, and operationalize AML by means of OPC Unified Architecture. The purpose is to simplify the creation of OPC Unified Architecture information models based on existing AML data. This can be used for re-engineering and maintenance use cases where the AML model evolves over time.

2. Accessing an OPC Unified Architecture server via AML; this involves the integration of OPC Unified Architecture information into AML. The goal is the lossless exchange of OPC Unified Architecture system configuration within AML models to simplify the setup of OPC Unified Architecture client connections to an OPC Unified Architecture server. This reduces the need for manual configuration for discovery and browsing mechanisms. This can be used for the configuration of communication networks based on the description of network configuration and structure (including communication components of sensors and actuators with respect to communication system parameters, network structure and wiring, quality of service, etc.).

Proposals for industrial usage and possible applications are listed in Annex A “Industrial application”. Annex B “Mapping example” contains an informative mapping example to explain the applied mapping rules. Annex C contains the ”AML Base Types” *NodeSet*[1)] including all *Nodes* and *References*, which are necessary for the application of AML and OPC Unified Architecture in combination.

This specification is based on the companion specification “OPC Unified Architecture for AutomationML” [13] which was created by a joint working group of the OPC Foundation and AutomationML e.V. It defines an OPC Unified Architecture information model to represent the AML models.

The tabular structure used within this specification is described in IEC 62541-5, 3.3, 4, and 5. The graphical notation used here is defined in IEC 62541-3, Annex D.

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.

ISO/IEC 25000:2014, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*

1) A *NodeSet* defines a set of *Nodes*, their *Attributes* and R*eferences* (see IEC 62541-6, Annex F).

IEC 62541-1:2015, *OPC Unified Architecture — Part 1: Overview and concepts*

IEC 62541-2:2015, *OPC Unified Architecture — Part 2: Security Model*

IEC 62541-3:2015, *OPC Unified Architecture — Part 3: Address Space Model*

IEC 62541-5:2015, *OPC Unified Architecture — Part 5: Information Model*

IEC 62541-6:2015, *OPC Unified Architecture — Part 6: Mappings*

IEC 62541-7:2015, *OPC Unified Architecture — Part 7: Profiles*

IEC 62714-1:2014, *Engineering data exchange format for use in industrial automation systems engineering — Automation markup language — Part 1: Architecture and general requirements*

IEC 62714-2:2015, *Engineering data exchange format for use in industrial automation systems engineering — Automation markup language — Part 2: Role class libraries*

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

3 Terms and definitions

For the purposes of this document, the terms and definitions given in the IEC 62714 series and the IEC 62541 parts 1, 2, 3, 5, 6 and 7 apply.

AML-related terms and names are preceded by the prefix AutomationML.

4 Symbols and abbreviated terms

AML	Automation Markup Language
CAEX	Computer Aided Engineering eXchange
CIP	Continuous Improvement Processes
COLLADA	Collaborative Design Activity
CPPS	Cyber-Physical Production System
HMI	Human-Machine Interface
JT	Jupiter Tesselation
MCO	Manufacturing Change Order
MCR	Manufacturing Change Request
MRO	Maintenance, Repair, Overhaul
OLP	Offline programming
PLC	Programmable Logic Controller

STEP	Standard for the Exchange of Product model data
UA	Unified Architecture
XML	Extensible Markup Language

5 AML model mapping to OPC Unified Architecture information model

5.1 AML modelling concept

AML models are divided into different hierarchical trees (see Figure 1): InstanceHierarchy, RoleClassLib, SystemUnitClassLib, and InterfaceClassLib.

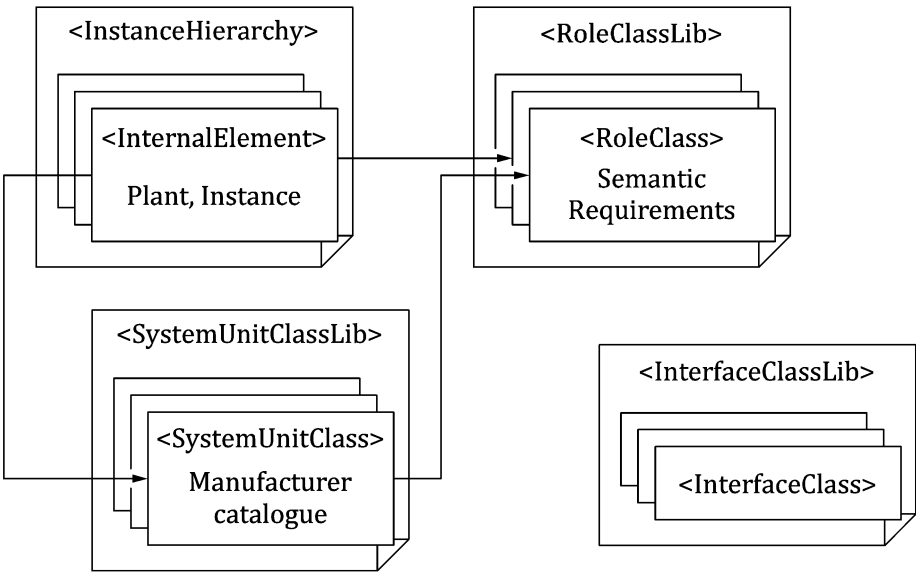


Figure 1 — AML hierarchical trees

The AML main elements and all possible relations between them are shown in Figure 2. The organization in a hierarchical tree is such that every element type can have child elements of the same type (see ChildElement relations). SystemUnitClasses, RoleClasses, and InterfaceClasses are defined in an inheritance structure; the RefBaseClassPath relations define such inheritance relations. RoleClasses can be assigned to SystemUnitClasses and InternalElements. In the latter case there are two different ways to make the RoleClass assignment using SupportedRoleClasses or the RoleRequirement. The remaining arrows in Figure 2 describe the possible relations between InternalElements and SystemUnitClasses. An InternalElement can be an instance of a SystemUnitClass. Within the scope of AML, SystemUnitClasses are only templates and can be changed after instantiation. The backward relation indicates that a SystemUnitClass can define sub-elements in terms of InternalElements. These sub-elements (InternalElements) should be included on instantiation. RoleClasses describe functions of a physical or logical plant object independent of a technical implementation. They offer a possibility of specifying an object in an abstract way and independent of the manufacturer. The assignment of a RoleClass to an element (SystemUnitClass or InternalElement) results in the allocation of fundamental functions or requirements.

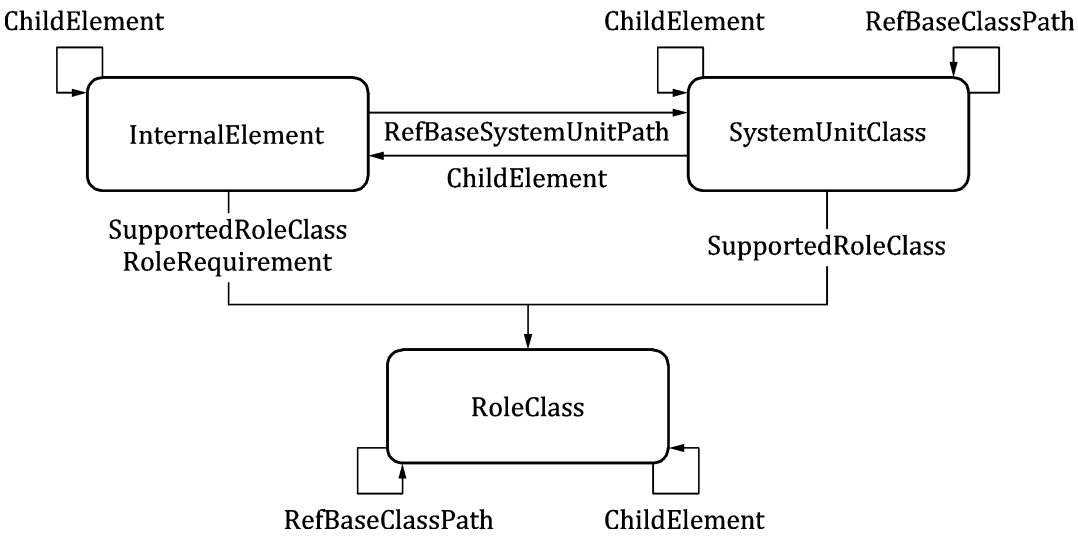


Figure 2 — AML main elements

Figure 3 depicts the basic structure of a RoleClass (RC), SystemUnitClass (SUC), or InternalElement (IE). Each can contain arbitrarily nested Attributes[2] and interfaces (see ChildElement relations). The interfaces are called ExternalInterfaces and are an instance of an InterfaceClass. The InterfaceClasses are stored hierarchically within InterfaceClassLibraries. They can be connected within an InstanceHierarchy to other ExternalInterfaces via InternalLinks. InterfaceClasses and certain ExternalInterfaces may also have arbitrarily nested Attributes.

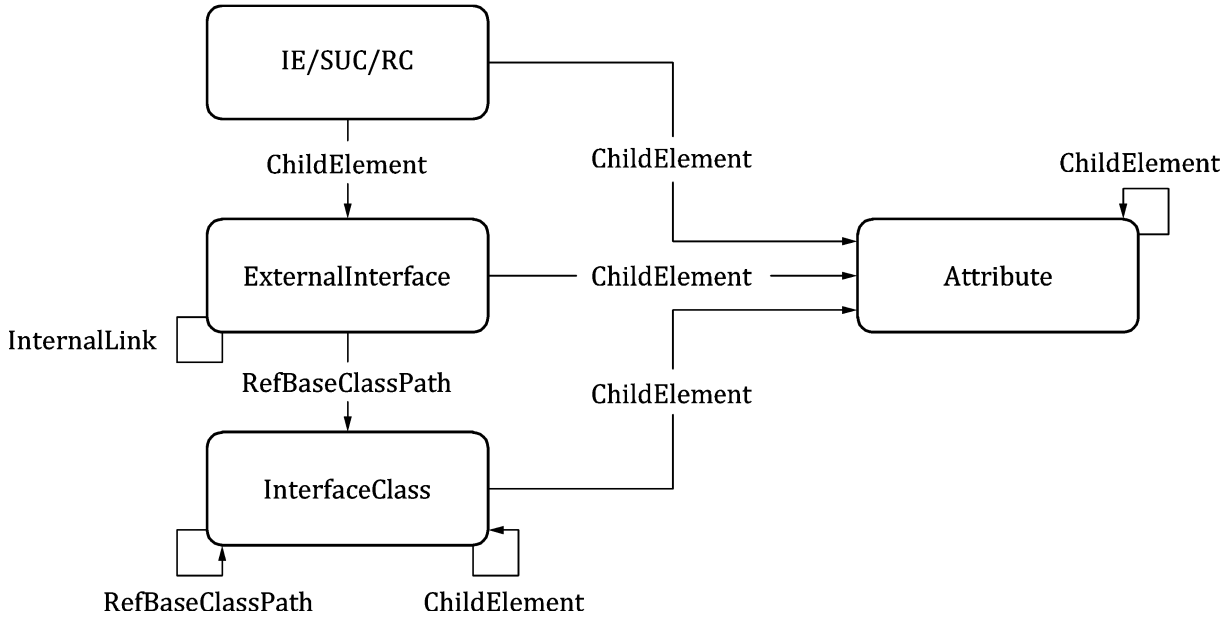


Figure 3 — AML main object details

[2] Attributes specify properties of an object, e.g. “length”, according to IEC 62424.

5.2 How to map OPC Unified Architecture Namespaces

Several OPC Unified Architecture *NodeSets* (see Figure 4) are defined to organize the AdressSpace. Figure 4 includes the *NodeSet* name, the corresponding *Namespace URI*, and the file name containing the *NodeSet*. All *NodeSet*s are based on the OPC Unified Architecture Base types. The *NodeSet "*AML base types" (see 5.5) defines all *Node*s and type definitions necessary for the modelling of AML models in OPC Unified Architecture. This includes organizational *Node*s for entry into the AML folder for RoleClassLibs or AML types (*ObjectType* "CAEXFileType"). One *NodeSet* is defined for the standardized, informative AML libraries as in IEC 62714-1, IEC 62714-2, IEC 62714-3, and IEC 62714-4. And finally, one or more *NodeSet*s are used for the concrete AML documents.

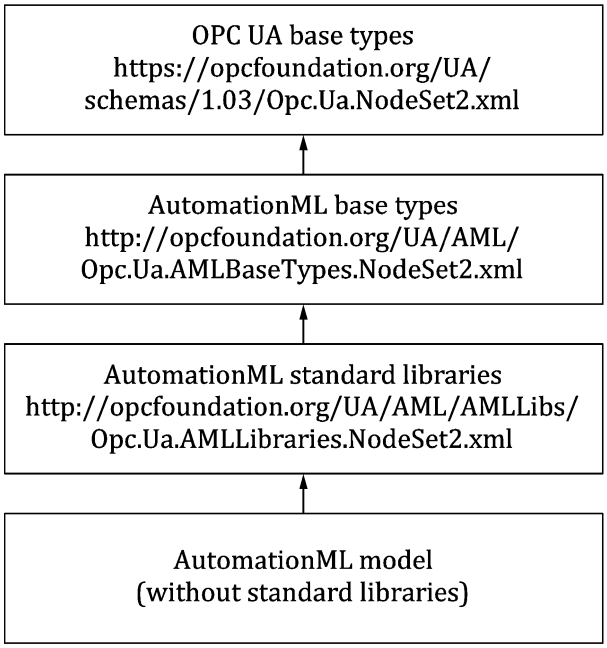


Figure 4 — Different *NodeSet*s for the usage of AML in OPC Unified Architecture information models

*Namespace*s are used in OPC Unified Architecture to create unique identifiers across different naming authorities. The *Attributes NodeId* and *BrowseName* are identifiers. A *Node* in the OPC Unified Architecture *AdressSpace* is unambiguously identified using a *NodeId*. Unlike *NodeIds*, the *BrowseName* cannot be used to unambiguously identify a *Node*. Different *Node*s may have the same *BrowseName*. They are used to build a browse path between two *Node*s or to define a standard *Property*.

Servers may often choose to use the same *Namespace* for the *NodeId* and the *BrowseName*. However, if they want to provide a standard *Property*, its *BrowseName* will have the *Namespace* of the standards body, while the *Namespace* of the *NodeId* reflects something else, for example the EngineeringUnits *Property*. All *NodeIds* of *Node*s not defined in this specification shall not use the standard *Namespace*s.

The *NodeIds* shall be generated during the mapping to OPC Unified Architecture following a specific schema which is arbitrary, e.g. numerical *NodeIds* from 1 to *n* (where *n* is the number of elements with *NodeId*).

NOTE For applications in embedded environments, numerical *NodeId*s or GUIDs should be used.

Table 1 provides a list of mandatory and optional *Namespace*s used in an AML OPC Unified Architecture *Server*.

Table 1 — Namespaces used in an OPC Unified Architecture server based on AML

Namespace	Description	Use
http://opcfoundation.org/UA/	*Namespace* for *NodeIds* and *BrowseNames* defined in the OPC Unified Architecture specification. This *Namespace* shall have *Namespace* index 0.	Mandatory
Local Server URI	*Namespace* for *Node*s defined in the local server. This may include types and instances used in a device represented by the server. This *Namespace* shall have *Namespace* index 1.	Mandatory
http://opcfoundation.org/UA/AML/	*Namespace* for *NodeIds* and *BrowseNames* defined in AML. The *Namespace* index shall be server-specific.	Mandatory
http://opcfoundation.org/UA/AML/AMLLibs/	*Namespace* for *NodeIds* and *BrowseNames* defined in AML libraries. The *Namespace* index is server-specific.	Optional

5.3 How to map AML model elements

The main structure of an AML file as in Figure 1 is mapped into a slightly different structure as that depicted in Figure 5. The InstanceHierarchies and libraries shall be stored in common folders ("AutomationMLInstanceHierarchies", "AutomationMLLibraries", "AutomationMLSystemUnitClassLibs", "AutomationMLRoleClassLibs", "AutomationMLInterfaceClassLibs").

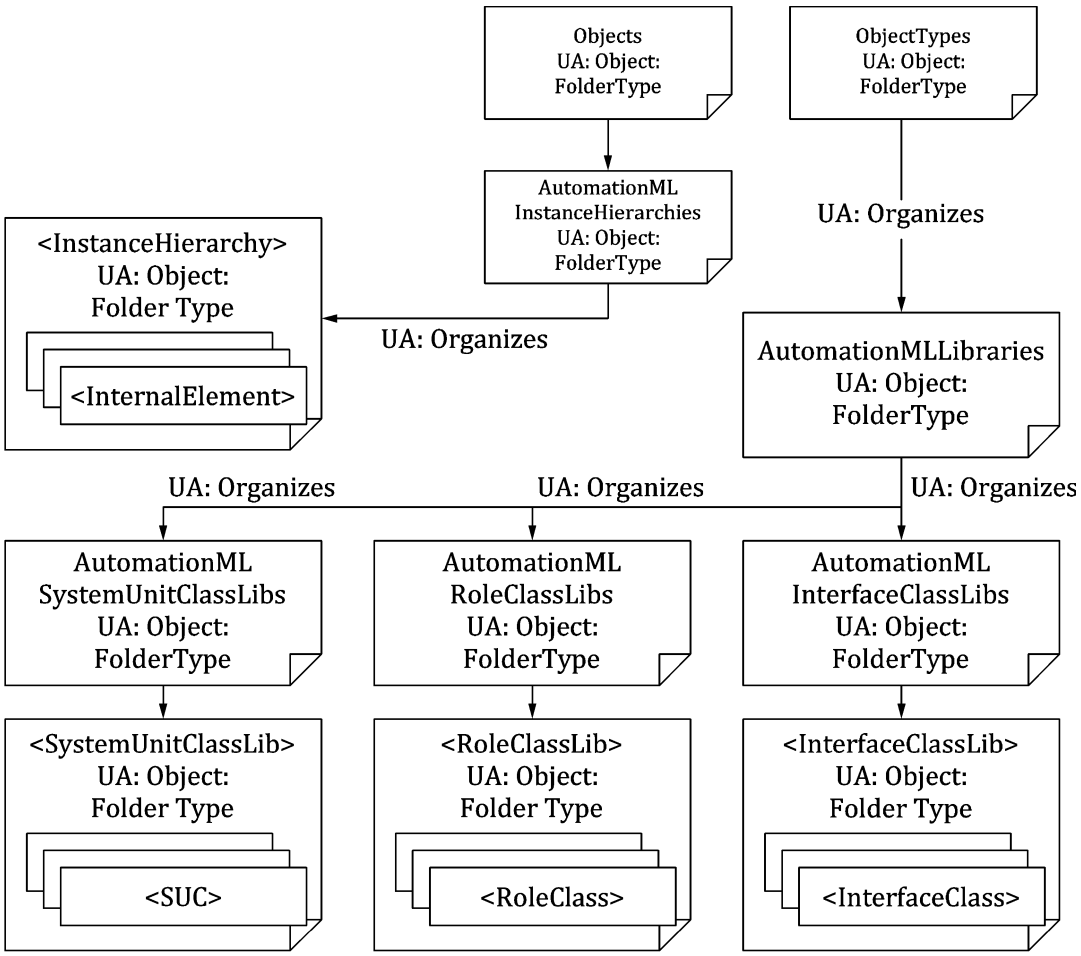


Figure 5 — CAEX file <-> OPC Unified Architecture Types [18]

The mapping of AML element type, and of their relations with OPC Unified Architecture *Nodes, NodeClasses* and *References* is depicted in Figure 6. The mapping of the internal structure of AML elements to OPC Unified Architecture *Nodes, NodeClasses* and *References* is depicted in Figure 7. The structure of the figures as compared to that in the figures in 5.1 has not been changed, so the mapping can be read directly from the graphic. Table 2, Table 3, and Table 4 summarize the mapping rules in lists. Basically, AML InternalElements are represented by OPC Unified Architecture *Objects*, while AML SystemUnitClasses and RoleClasses are represented by OPC Unified Architecture *ObjectTypes*. Detailed definitions of new *ObjectTypes*, *ReferenceTypes* and *VariableType*s are given in 5.5.

The following provision applies to the mapping of AML models to OPC Unified Architecture models:

- For every *Reference* created in OPC Unified Architecture there shall be a corresponding *SourceNode* and *TargetNode*. Otherwise, *Reference*s shall not be created.

ChildElement relations within class structures (SystemUnitClass, RoleClass, and InterfaceClass) are mapped to the OPC Unified Architecture *Organizes Reference*s. InternalElements, which follow the ChildElement relation, are referenced via *HasComponent Reference*s. The inheritance relations within SystemUnitClasses, RoleClasses, and InterfaceClasses are mapped to OPC Unified Architecture *HasSubtype Reference*s. The internal structures of SystemUnitClass are depicted by InternalElements, which are connected via OPC Unified Architecture *HasComponent Reference*s. The RefBaseClassPath relation in AML is an OPC Unified Architecture *HasSubtype Reference*. The RefBaseSystemUnitPath relation in AML is an OPC Unified

Architecture *HasTypeDefinition Reference*. The semantic direction of *HasSubtype Reference* is inverted compared to the corresponding RefBaseClassPath relation.

The referencing of RoleClasses from InternalElements or SystemUnitClass via SupportedRoleClass or RoleRequirement in AML is done via OPC Unified Architecture "HasAMLSupportedRoleClass" and "HasAMLRoleRequirement" *Reference*s.

Table 2 — Element mapping

AML	**OPC Unified Architecture**
InternalElement	Object
SystemUnitClass	ObjectType
RoleClass	ObjectType

Table 3 — Attribute mapping

AML	**OPC Unified Architecture**
Name	*BrowseName*
ID	*Property* "ID" with *DataType String*
Description	*Description*
Version	*Property* "Version"[a] with *DataType String*
Revision	—[a]
Copyright	*Property* "Copyright" with *DataType String*
AdditionalInformation	*Property* "AdditionalInformation" with *DataType String (ValueRank = 1)* Complete XML text, e.g. <AdditionalInformation AutomationMLVersion="2.0" />
ChangeMode	—[a]
CAEXFile.FileName	*Property* "FileName" with *DataType String*[b]
CAEXFile.SchemaVersion	*Property* "SchemaVersion" with *DataType String*
ExternalReference	*Property* Type "ExternalReferenceType" with *DataType String*
ExternalReference.Alias	*BrowseName* of the "ExternalReference" *Property*
ExternalReference.Path	*Value* of the "ExternalReference" *Property*

AML	OPC Unified Architecture
Attribute.Value	*Value*
Attribute.AttributeDataType	*DataType* (see Table 7)
Attribute.Unit	*Property* "Unit" with *DataType String*
Attribute.DefaultValue	*Property* "DefaultValue" with same *DataType* as the Attribute
Attribute.RefSemantic	*Property* "RefSemantic" with *DataType String (ValueRank = 1)*
Attribute.Constraint	*Property* "Name of Constraint" with *DataType BaseDataType* and *VariableType* "OrdinalScaledConstraintType", "NominalScaledConstraintType" or "UnknownConstraintType"
OrdinalScaledType.RequiredMaxValue	"RequiredMaxValue" of *Property* of type "OrdinalConstraintType" with *DataType String*
OrdinalScaledType.RequiredValue	"RequiredValue" of *Property* of type "OrdinalConstraintType" with *DataType String*
OrdinalScaledType.RequiredMinValue	"RequiredMinValue" of *Property* of type "OrdinalConstraintType" with *DataType String*
NominalScaledType.RequiredValue	One item of *Property* of type "NominalConstraintType" with *DataType String (ValueRank = 1)*
UnknownType.Requirements	*Property* of type "UnknownConstraintType" with *DataType String*
MappingObject	*Reference* of type "HasAMLSupportedRoleClass", "HasAMLRoleRequirement" between the OPC UA *Node*s of the mapped AML elements (Attribute or ExternalInterface).
CAEXBasicObject information (Version, Description, Copyright, and AdditionalInformation) of: RefSemantic, RoleRequirement, SupportedRoleClass, and MappingObject	—[a]

[a] Currently not mapped into the OPC UA model.

[b] For a better usability of the CAEXFile.FileName it should be used as part of the *BrowseName*.

Table 4 — Relation mapping

AML Source	AML Target	AML Relation	OPC Unified Architecture Reference	Description
InternalElement	RoleClass	SupportedRoleClass	HasAMLSupportedRoleClass	Semantic similar to HasTypeDefinition
InternalElement	RoleClass	RoleRequirement	HasAMLRoleRequirement	Semantic similar to HasTypeDefinition
InternalElement	SystemUnitClass	RefBaseSystemUnitPath	HasTypeDefinition	
InternalElement	InternalElement		HasComponent	ChildElement
SystemUnitClass	RoleClass	SupportedRoleClass	HasAMLSupportedRoleClass	Semantic similar to HasTypeDefinition
SystemUnitClass	InternalElement		HasComponent	ChildElement
SystemUnitClass	SystemUnitClass		Organizes	ChildElement
SystemUnitClass	SystemUnitClass	RefBaseClassPath	HasSubtype	
RoleClass	RoleClass		Organizes	ChildElement
RoleClass	RoleClass	RefBaseClassPath	HasSubtype	

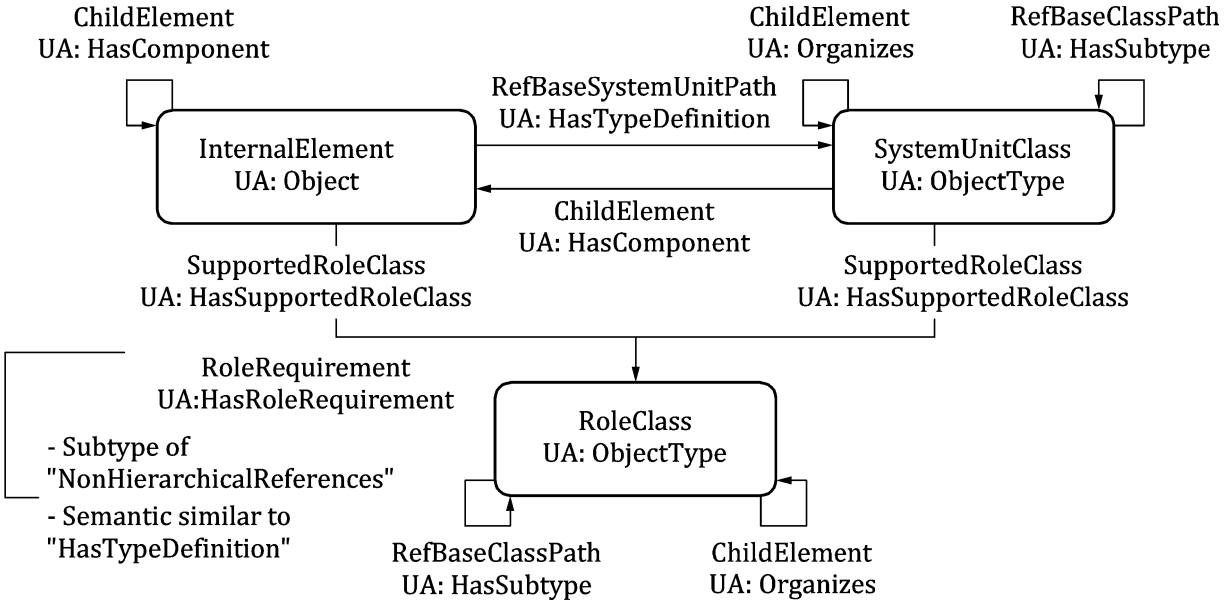


Figure 6 — AML element types and the OPC Unified Architecture *Node*s and *Reference*s [18]

The mappings of the AML object details (see Figure 3) are depicted in Figure 7. The analogy between the two figures is apparent. Table 5 and Table 6 summarize the mapping rules in lists. AML Attributes become OPC Unified Architecture *Variable*s. AML InterfaceClasses are transformed into OPC Unified Architecture *ObjectTypes* and the corresponding AML ExternalInterface elements are modelled by OPC Unified Architecture Objects.

The Attributes and ExternalInterfaces are referenced via *HasComponent Reference*s. The InterfaceClass is the type definition of the ExternalInterface; therefore, it is assigned via *HasTypeDefinition Reference*. Relations between ExternalInterfaces (AML InternalLinks) are assigned via “HasAMLInternalLink” *Reference*. Details are given in 5.4.

Table 5 — Element mapping

AML	**OPC Unified Architecture**
InternalElement	Object
SystemUnitClass	ObjectType
RoleClass	ObjectType
InterfaceClass	ObjectType
Attribute	Variable
ExternalInterface	Object

Table 6 — Relation mapping

AML Source	AML Target	AML Relation	OPC Unified Architecture Reference	Description
InternalElement	Attribute		HasComponent	ChildElement
SystemUnitClass	Attribute		HasComponent	ChildElement
RoleClass	Attribute		HasComponent	ChildElement
InternalElement	ExternalInterface		HasComponent	ChildElement
SystemUnitClass	ExternalInterface		HasComponent	ChildElement
RoleClass	ExternalInterface		HasComponent	ChildElement
ExternalInterface	ExternalInterface	InternalLink	HasAMLInternalLink	
ExternalInterface	Attribute		HasComponent	ChildElement
ExternalInterface	InterfaceClass	RefBaseClassPath	HasTypeDefinition	
InterfaceClass	Attribute		HasComponent	ChildElement
InterfaceClass	InterfaceClass	RefBaseClassPath	HasSubtype	
InterfaceClass	InterfaceClass		Organizes	ChildElement
Attribute	Attribute		HasComponent	ChildElement

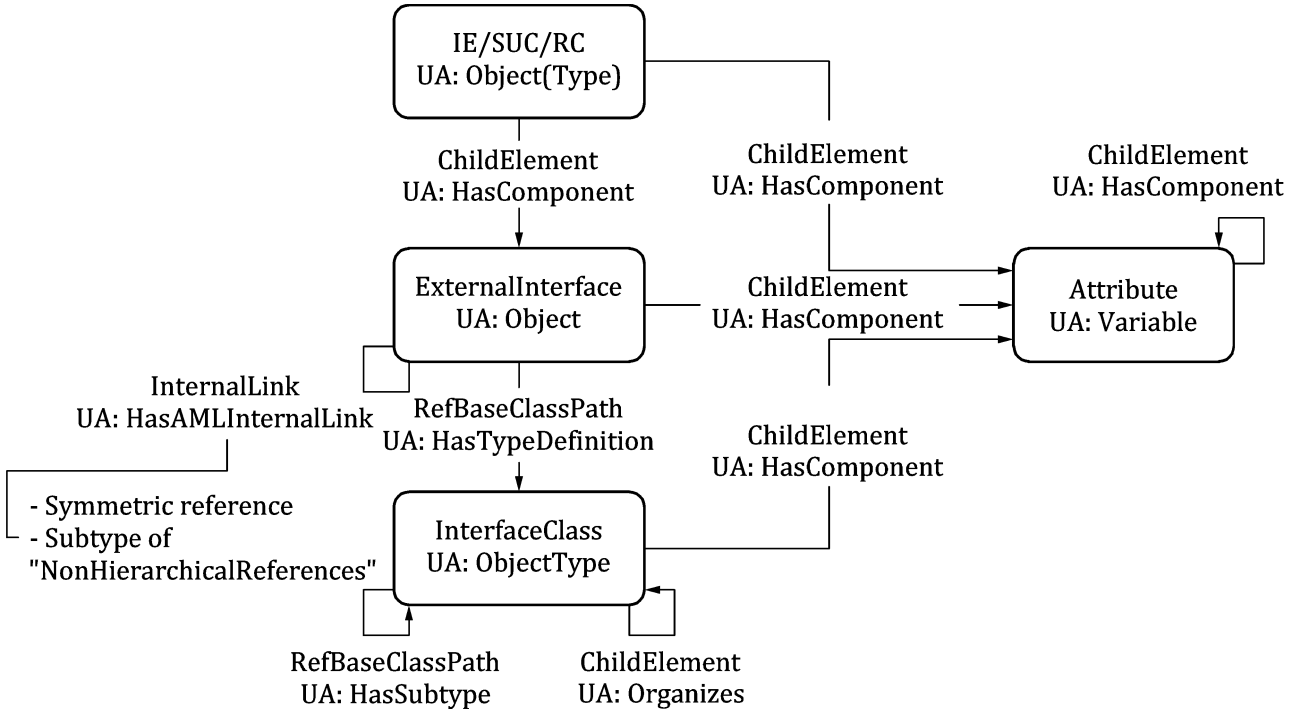


Figure 7 — Internal structure of AML elements and the OPC Unified Architecture *Node*s and *Reference*s [18]

5.4 Mapping of AML XML DataTypes to OPC Unified Architecture DataTypes

The mapping of AML XML DataTypes to OPC Unified Architecture *DataType*s is described in Table 7.

Table 7 — Mapping of XML DataTypes to OPC Unified Architecture DataTypes

XML SimpleType	OPC Unified Architecture DataType
string	String
boolean	Boolean
decimal	Double
float	Float
double	Double
duration	Duration
dateTime	DateTime
time	Duration
date	DateTime
gYearMonth	DateString

XML SimpleType	OPC Unified Architecture DataType
gYear	DateString
gMonthDay	String
gDay	String
gMonth	String
hexBinary	ByteString
base64Binary	ByteString
anyURI	String
QName	String
NOTATION	String (ValueRank = 1)
normalizedString	String
token	String
language	LocaleId
NMTOKEN	String
NMTOKENS	String (ValueRank = 1)
Name	String
NCName	String
ID	String
IDREF	String
IDREFS	String (ValueRank = 1)
ENTITY	String
ENTITIES	String (ValueRank = 1)
integer	Int64
nonPositiveInteger	Int64
negativeInteger	Int64
long	Int64

XML SimpleType	OPC Unified Architecture DataType
int	Int32
short	Int16
byte	SByte
nonNegativeInteger	UInt64
unsignedLong	UInt64
unsignedInt	UInt32
unsignedShort	UInt16
unsignedByte	Byte
positiveInteger	Int64

5.5 AML base types for OPC Unified Architecture information model

5.5.1 ObjectTypes

5.5.1.1 CAEXBasicObjectType

5.5.1.1.1 General

The "CAEXBasicObjectType" defines all general characteristics of a CAEX element. All other CAEX elements derive from the "CAEXBasicObjectType". The "CAEXBasicObjectType" inherits all *Properties* of the *BaseObjectType*.

5.5.1.1.2 ObjectType Definition

The “CAEXBasicObjectType” is formally defined in Table 8.

Table 8 — CAEXBasicObjectType Definition

Attribute	**Value**				
BrowseName	CAEXBasicObjectType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the BaseObjectType					
HasProperty	Variable	Version	String	PropertyType	Optional
HasProperty	Variable	Copyright	String	PropertyType	Optional
HasProperty	Variable	AdditionalInformation	String (ValueRank = 1)	PropertyType	Optional

5.5.1.1.3 ObjectType Description

5.5.1.1.3.1 Version

“Version” provides the optional organizational information about the state of the version for the “CAEXBasicObjectType”

5.5.1.1.3.2 Copyright

“Copyright” provides the optional organizational information about copyright for the “CAEXBasicObjectType”

5.5.1.1.3.3 AdditionalInformation

“AdditionalInformation” provides an optional auxiliary field that may contain any additional information about a CAEX object defined by the “CAEXBasicObjectType”. The *ValueRank* of this *Property* is 1.

5.5.1.2 CAEXFileType

5.5.1.2.1 General

The “CAEXFileType” defines all general characteristics of a CAEX file and includes all CAEX libraries and instance hierarchies. The “CAEXFileType” inherits all *Properties* of the “CAEXBasicObjectType”.

5.5.1.2.2 ObjectType Definition

The “CAEXFileType” is formally defined in Table 9.

Table 9 — CAEXFileType Definition

Attribute	Value				
BrowseName	CAEXFileType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the CAEXBasicObjectType					
HasComponent	Object	InstanceHierarchies		FolderType	Mandatory
HasComponent	Object	InterfaceClassLibs		FolderType	Mandatory
HasComponent	Object	RoleClassLibs		FolderType	Mandatory
HasComponent	Object	SystemUnitClassLibs		FolderType	Mandatory
HasProperty	Variable	SchemaVersion	String	PropertyType	Mandatory

5.5.1.2.3 ObjectType Description

5.5.1.2.3.1 InstanceHierarchies

The “InstanceHierarchies” folder includes all CAEX InstanceHierarchies of a CAEX file.

5.5.1.2.3.2 InterfaceClassLibs

The “InterfaceClassLibs” folder includes all CAEX InterfaceClassLibs of a CAEX file.

5.5.1.2.3.3 RoleClassLibs

The “RoleClassLibs” folder includes all CAEX RoleClassLibs of a CAEX file.

5.5.1.2.3.4 SystemUnitClassLibs

The “SystemUnitClassLibs” folder includes all CAEX SystemUnitClassLibs of a CAEX file.

5.5.1.2.3.5 SchemaVersion

“SchemaVersion” provides a mandatory field that describes the version of the CAEX schema.

5.5.1.3 CAEXObjectType

5.5.1.3.1 General

The “CAEXObjectType” defines all general characteristics of a CAEX object. All other CAEX objects derive from it. The “CAEXObjectType” inherits all *Properties* of the “CAEXBasicObjectType”.

5.5.1.3.2 ObjectType Definition

The “CAEXObjectType” is formally defined in Table 10.

Table 10 — CAEXObjectType Definition

Attribute	Value				
BrowseName	CAEXObjectType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the CAEXBasicObjectType					
HasProperty	Variable	ID	String	PropertyType	Optional

5.5.1.3.3 ObjectType Description

5.5.1.3.3.1 ID

"ID" provides a unique ID of the "CAEXObjectType".

5.5.2 ReferenceTypes

5.5.2.1 HasAMLSupportedRoleClass

5.5.2.1.1 General

The "HasAMLSupportedRoleClass" is a concrete *ReferenceType* that can be used directly. It is a subtype of *NonHierarchicalReference*.

5.5.2.1.2 ObjectType Definition

The "HasAMLSupportedRoleClass" is formally defined in Table 11.

Table 11 — HasAMLSupportedRoleClass

Attribute	**Value**		
BrowseName	HasAMLSupportedRoleClass		
InverseName	IsSupportedRole		
Symmetric	False		
IsAbstract	False		
References	**NodeClass**	**BrowseName**	**Comment**
Subtype of NonHierarchicalReference ReferenceType defined in OPC Unified Architecture Part 5			

This *ReferenceType* is used to describe a SupportedRoleClass relation in AML.

The *SourceNode* of this *ReferenceType* is an *Object* or *ObjectType* (InternalElement or SystemUnitClass in AML).

The *TargetNode* of this *ReferenceType* is an *ObjectType* (RoleClass in AML).

5.5.2.2 HasAMLRoleRequirement

5.5.2.2.1 General

The "HasAMLRoleRequirement" is a concrete *ReferenceType* that can be used directly. It is a subtype of *NonHierarchicalReference*.

5.5.2.2.2 ObjectType Definition

The "HasAMLRoleRequirement" is formally defined in Table 12.

Table 12 — HasAMLRoleRequirement

Attribute	**Value**		
BrowseName	HasAMLRoleRequirement		
InverseName	IsRequiredRole		
Symmetric	False		
IsAbstract	False		
References	**NodeClass**	**BrowseName**	**Comment**
Subtype of NonHierarchicalReference ReferenceType defined in OPC Unified Architecture Part 5			

This *ReferenceType* is used to describe a RoleRequirement relation in AML.

The *SourceNode* of this *ReferenceType* is an *Object* (InternalElement in AML).

The *TargetNode* of this *ReferenceType* is an *ObjectType* (RoleClass in AML).

5.5.2.3 HasAMLInternalLink

5.5.2.3.1 General

The "HasAMLInternalLink" is a concrete *ReferenceType* that can be used directly. It is a subtype of *NonHierarchicalReference*.

5.5.2.3.2 ReferenceType Definition

The "HasAMLInternalLink" is formally defined in Table 13.

Table 13 — HasAMLInternalLink

Attribute	**Value**		
BrowseName	HasAMLInternalLink		
InverseName	HasAMLInternalLink		
Symmetric	True		
IsAbstract	False		
References	**NodeClass**	**BrowseName**	**Comment**
Subtype of NonHierarchicalReference ReferenceType defined in OPC Unified Architecture Part 5			

This *ReferenceType* is used to describe an InternalLink relation in AML.

The *SourceNode* of this *ReferenceType* is an *Object* (ExternalInterface in AML).

The *TargetNode* of this *ReferenceType* is an *Objec*. (ExternalInterface in AML).

5.5.2.4 HasAMLUAReference

5.5.2.4.1 General

The "HasAMLUAReference" is a concrete *ReferenceType* that can be used directly. It is a subtype of *NonHierarchicalReference*.

5.5.2.4.2 ObjectType Definition

The "HasAMLUAReference" is formally defined in Table 14.

Table 14 — HasAMLUAReference

Attribute	**Value**		
BrowseName	HasAMLUAReference		
InverseName	IsAMLReferenceOf		
Symmetric	False		
IsAbstract	False		
References	**NodeClass**	**BrowseName**	**Comment**
Subtype of NonHierarchicalReference ReferenceType defined in OPC Unified Architecture Part 5			

This *ReferenceType* is used to describe a relation to an OPC Unified Architecture *Node* from AML which forms part of another *Namespace*.

The *SourceNode* of this *ReferenceType* is an *Object*, *ObjectType*, or *Variable* which is part of a transformed AML model.

The *TargetNode* of this *ReferenceType* is an OPC Unified Architecture *Node* of any type.

5.5.3 VariableTypes

5.5.3.1 AMLBasicVariableType

5.5.3.1.1 General

The "AMLBasicVariableType" defines all general characteristics of a CAEX element which becomes an OPC Unified Architecture *Variable*. The "AMLBasicVariableType" inherits all *Properties* of the *BaseVariableType*.

5.5.3.1.2 VariableType Definition

The "AMLBasicVariableType" is formally defined in Table 15.

Table 15 — AMLBasicVariableType Definition

Attribute	**Value**				
BrowseName	AMLBasicVariableType				
IsAbstract	False				
ValueRank	-2				
DataType	BaseDataType				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the BaseVariableType					
HasProperty	Variable	Version	String	PropertyType	Optional
HasProperty	Variable	Copyright	String	PropertyType	Optional
HasProperty	Variable	AdditionalInformation	String	PropertyType	Optional

5.5.3.1.3 VariableType Description

5.5.3.1.3.1 Version

"Version" provides the optional organizational information about the state of the version for the "AMLBasicVariableType"

5.5.3.1.3.2 Copyright

"Copyright" provides the optional organizational information about copyright for the "AMLBasicVariableType"

5.5.3.1.3.3 AdditionalInformation

"AdditionalInformation" provides an optional auxiliary field that may contain any additional information about a CAEX object defined by the "AMLBasicVariableType". The *ValueRank* of this *Property* is 1.

5.5.3.2 ExternalReferenceType

5.5.3.2.1 General

The "ExternalReferenceType" defines a container element for alias definitions of external CAEX files.

5.5.3.2.2 VariableType Definition

The “ExternalReferenceType” is formally defined in Table 16.

Table 16 — ExternalReferenceType Definition

Attribute	**Value**				
BrowseName	ExternalReferenceType				
IsAbstract	False				
ValueRank	-1				
DataType	String				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the AMLBasicVariableType					

5.5.3.3 ConstraintType

5.5.3.3.1 General

The “ConstraintType” defines an element to restrict the range of validity of a defined AML Attribute. All other constraint types derive from it.

5.5.3.3.2 VariableType Definition

The “ConstraintType” is formally defined in Table 17.

Table 17 — ConstraintType Definition

Attribute	**Value**				
BrowseName	ConstraintType				
IsAbstract	True				
ValueRank	-2				
DataType	BaseDataType				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the AMLBasicVariableType					

5.5.3.4 NominalConstraintType

5.5.3.4.1 General

The “NominalConstraintType” defines constraints of nominal scaled AML Attribute values. Elements of this type can define multiple required values in order to define a discrete value range of the AML Attribute.

5.5.3.4.2 VariableType Definition

The “NominalConstraintType” is formally defined in Table 18.

Table 18 — NominalConstraintType Definition

Attribute	**Value**				
BrowseName	NominalConstraintType				
IsAbstract	False				
ValueRank	1				
DataType	String				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the ConstraintType					

5.5.3.5 OrdinalConstraintType

5.5.3.5.1 General

The “OrdinalConstraintType” defines constraints of ordinal scaled AML Attribute values.

5.5.3.5.2 VariableType Definition

The “OrdinalConstraintType” is formally defined in Table 19.

Table 19 — OrdinalConstraintType Definition

Attribute	Value				
BrowseName	OrdinalConstraintType				
IsAbstract	False				
ValueRank	-1				
DataType	OrdinalConstraintItem				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the ConstraintType					

5.5.3.6 UnknownConstraintType

5.5.3.6.1 General

The “UnknownConstraintType” defines constraints for attribute values of an unknown scale type. Elements of this type define informative requirements as a constraint for an AML Attribute value.

5.5.3.6.2 VariableType Definition

The “UnknownConstraintType” is formally defined in Table 20.

Table 20 — UnknownConstraintType Definition

Attribute	Value				
BrowseName	UnknownConstraintType				
IsAbstract	False				
ValueRank	-1				
DataType	String				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the ConstraintType					

5.5.3.7 AMLVariableType

5.5.3.7.1 General

The “AMLVariableType” defines all general characteristics of AML Attributes. All other AML Attributes derive from it.

5.5.3.7.2 VariableType Definition

The “AMLVariableType” is formally defined in Table 21.

Table 21 — AMLVariableType Definition

Attribute	Value				
BrowseName	AMLVariableType				
IsAbstract	False				
ValueRank	-1				
DataType	BaseDataType				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Inherit the components of the AMLBasicVariableType					
HasProperty	Variable	ID	String	PropertyType	Optional
HasProperty	Variable	Unit	String	PropertyType	Optional
HasProperty	Variable	DefaultValue	String	PropertyType	Optional
HasProperty	Variable	RefSemantic	String	PropertyType	Optional

5.5.3.7.3 VariableType Description

5.5.3.7.3.1 ID

"ID" provides a unique ID of the "AMLVariableType".

5.5.3.7.3.2 Unit

"Unit" describes the unit of the "AMLVariableType".

5.5.3.7.3.3 DefaultValue

"DefaultValue" predefines a default value of the "AMLVariableType".

5.5.3.7.3.4 RefSemantic

"RefSemantic" describes a reference to a definition to an attribute in a standardized library of the "AMLVariableType". This allows the semantic definition of AML Attributes. The *ValueRank* of this *Property* is 1.

5.5.4 Structure DataTypes

5.5.4.1 OrdinalConstraintItem

This *DataType* is a structure that defines ordinal constraint of an AML Attribute. Its composition is formally defined in Table 22.

Table 22 — OrdinalConstraintItem Structure

Name	Type	Description
OrdinalConstraintItem	structure	Constraint of ordinal scaled AML Attribute values
RequiredMaxValue	String	Maximum value of a AML Attribute
RequiredValue	String	Required value of a AML Attribute
RequiredMinValue	String	Minimum value of a AML Attribute

Its representation in the *AddressSpace* is defined in Table 23.

Table 23 — OrdinalConstraintItem Definition

Attributes	Value
BrowseName	OrdinalConstraintItem

5.5.5 OPC Unified Architecture Objects used to organize the AdressSpace structure

There are two possibilities of accessing the model: one that is AML-specific and one that is OPC Unified Architecture-specific. The organizational *Node*s facilitate the navigation (browsing) to elements. Figure 8 depicts an example of an AdressSpace. Different colours represent different *NodeSet*s as explained in 5.2.

For users familiar with OPC Unified Architecture, *Objects* and *ObjectTypes* can be accessed via the usual Unified Architecture means (*Objects* folder and *ObjectTypes* folder). The global instance view can be reached via the *Objects Node* and a specific *Node* called AutomationMLInstanceHierarchies of the type *FolderType*. The global type view can be reached via the *ObjectTypes Node* and the corresponding AML-specific *ObjectType*s.

The file view is intended for AML users who are looking for a direct AML view/representation. This is realized by the *Node* of the type FolderType named AMLFiles below the Objects *Node*. This *Node* organizes all File_xyz *Node*s where File_xyz stands for the AML file name.

The File_xyz *Node* consists of *Node*s of type *FolderType* called

- “InstanceHierarchies” (for AML InstanceHierarchy elements),
- “RoleClassLibs” (for AML RoleClassLibrary elements),
- “SystemUnitClassLibs” (for AML SystemUnitClassLibrary elements), and
- “InterfaceClassLibs” (for AML InterfaceClassLibrary elements).

Below File_xyz all *Reference*s need to be browsed bidirectionally (isForward = False).

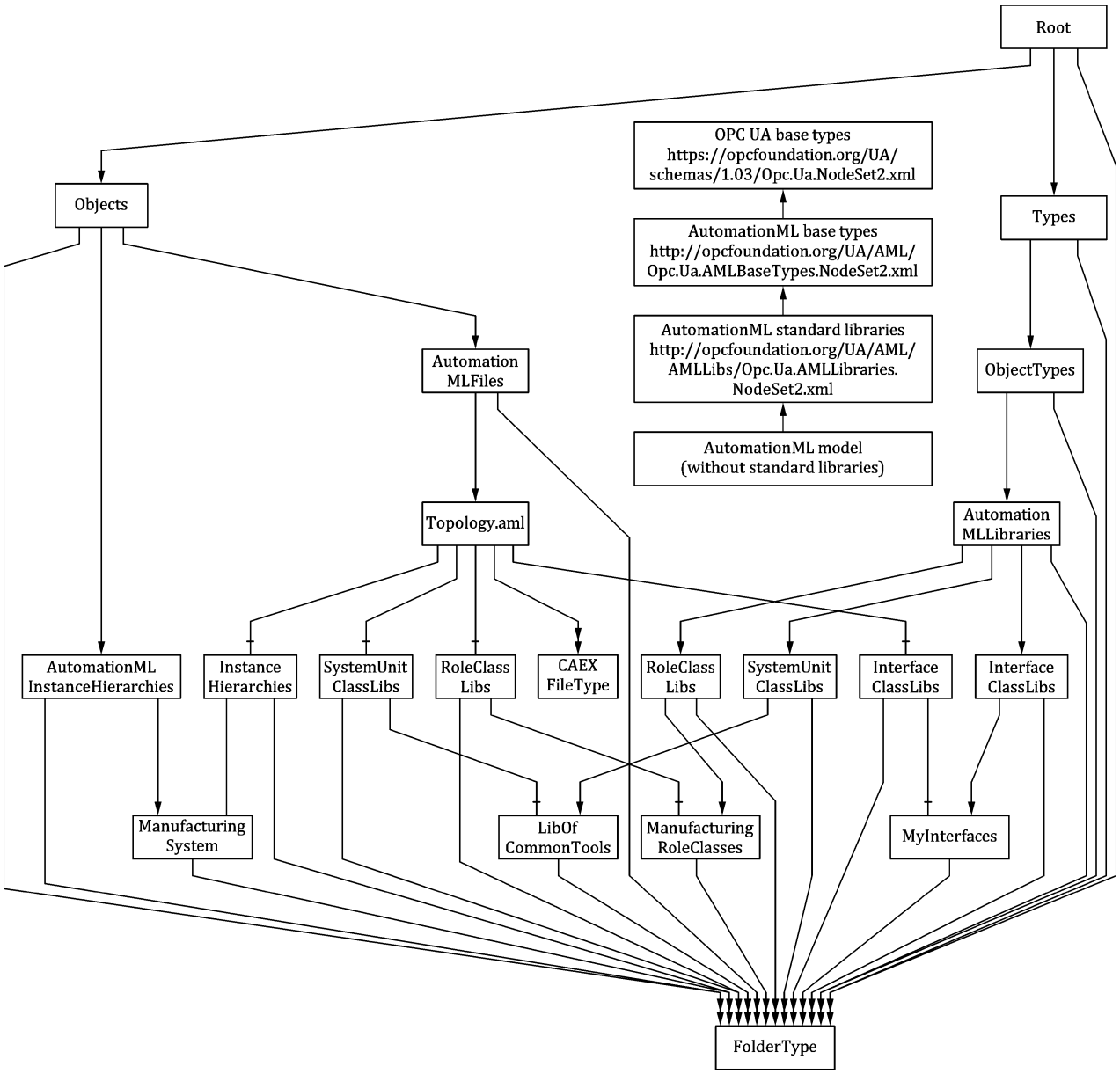


Figure 8 — AML information model structure in OPC Unified Architecture

The names of the AML-specific *Node*s are as follows: “AutomationMLFiles”, “AutomationMLLibraries”, “AutomationMLInstanceHierarchies”, “CAEXBasicObjectType” are entry points into the hierarchies. They include AML or CAEX in their *BrowseName*s. Everything below these *Node*s does not need an AML or CAEX prefix anymore, e.g. “RoleClassLibs”.

NOTE AML base libraries should not be included in more than one AML file in order to avoid numerous representations of types.

5.5.5.1 Instances and entry points

5.5.5.1.1 AutomationMLFiles

This Instance is a child of *Objects*. It is defined in Table 24.

The "AutomationMLFiles" folder represents a browse entry point when looking for AML files in the OPC Unified Architecture *information model*.

Table 24 — AutomationMLFiles Instance Definition

Name	TypeDefinition
AutomationMLFiles	FolderType

5.5.5.1.2 AutomationMLInstanceHierarchies

This Instance is a child of *Objects*. It is defined in Table 25.

The "AutomationMLInstanceHierarchies" folder represents a browse entry point when looking for AML InstanceHierarchies in the OPC Unified Architecture *information model*.

Table 25 — AutomationMLInstanceHierarchies Instance Definition

Name	TypeDefinition
AutomationMLInstanceHierarchies	FolderType

5.5.5.1.3 AutomationMLLibraries

This Instance is a child of *ObjectType*s. It is defined in Table 26.

The "AutomationMLLibraries" folder represents a browse entry point when looking for AML libraries in the OPC Unified Architecture *information model*.

Table 26 — AutomationMLLibraries Instance Definition

Name	TypeDefinition
AutomationMLLibraries	FolderType

5.5.5.1.4 InterfaceClassLibs

This Instance is a child of "AutomationMLLibraries". It is defined in Table 27.

The "InterfaceClassLibs" folder represents a browse entry point when looking for AML InterfaceClassLibs in the OPC Unified Architecture *information model*.

Table 27 — InterfaceClassLibs Instance Definition

Name	TypeDefinition
InterfaceClassLibs	FolderType

5.5.5.1.5 RoleClassLibs

This Instance is a child of "AutomationMLLibraries". It is defined in Table 28.

The "RoleClassLibs" folder represents a browse entry point when looking for AML RoleClassLibs in the OPC Unified Architecture *information model*.

Table 28 — RoleClassLibs Instance Definition

Name	TypeDefinition
RoleClassLibs	FolderType

5.5.5.1.6 SystemUnitClassLibs

This Instance is a child of "AutomationMLLibraries". It is defined in Table 29.

The "SystemUnitClassLibs" folder represents a browse entry point when looking for AML SystemUnitClassLibs in the OPC Unified Architecture *information model*.

Table 29 — SystemUnitClassLibs Instance Definition

Name	TypeDefinition
SystemUnitClassLibs	FolderType

5.6 Mapping of AML libraries

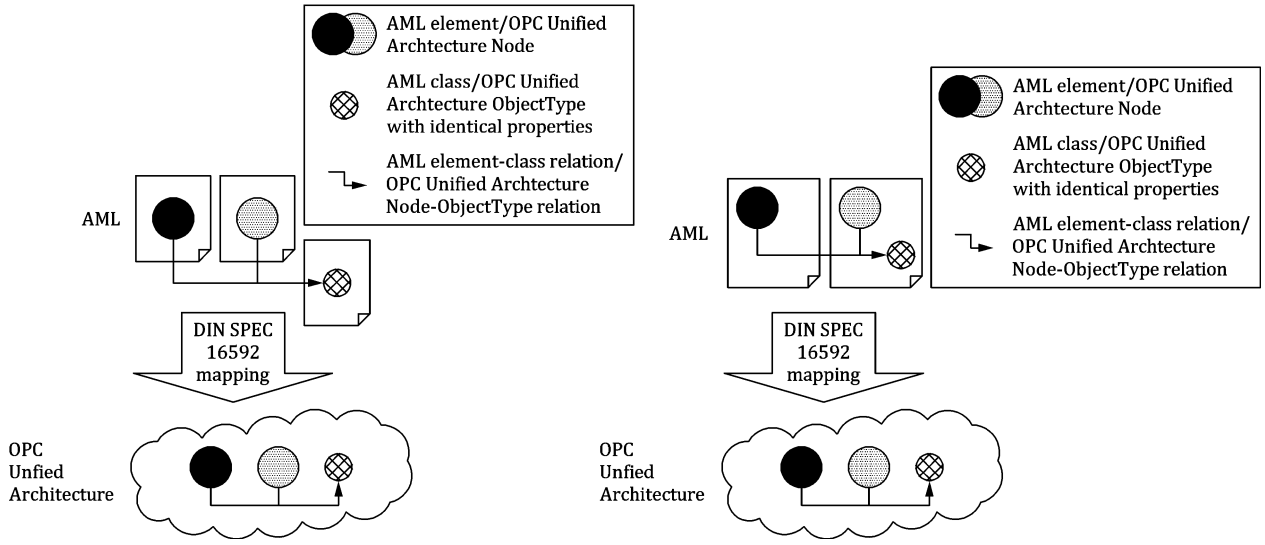
The normative and informative libraries of AML defined in IEC 62714-1, IEC 62714-2, IEC 62714-3, and IEC 62714-4 shall be translated to OPC Unified Architecture according to the mapping rules laid down in this clause.

5.7 Mapping of multiple AML documents

The following provisions shall apply to the mapping of multiple AML documents in one OPC Unified Architecture information model:

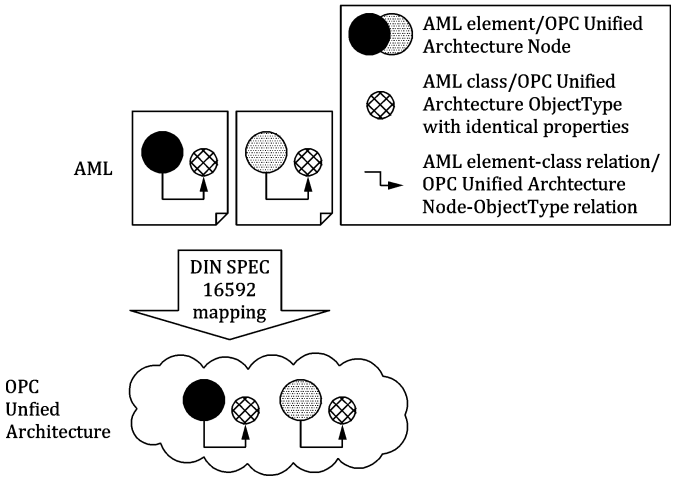
- If multiple AML elements reference **one single** AML class (i.e. SystemUnitClass/InterfaceClass/RoleClass), regardless of whether the class is located in the same or in an external AML document, **for this, one single** OPC Unified Architecture *ObjectType* shall be created (see Figure 9).

- If multiple AML elements reference **multiple** AML classes (i.e. SystemUnitClass/InterfaceClass/RoleClass), regardless of whether the classes are located in the same or in an external AML document, **for those, multiple** OPC Unified Architecture *ObjectTypes* shall be created (see Figure 10)



NOTE Possible cases are instance-class assignments and class-inheritance.

Figure 9 — Multiple AML elements reference one single AML class



NOTE Possible cases are instance-class assignments and class-inheritance.

Figure 10 — Multiple AML elements reference multiple AML classes

6 Integration of OPC Unified Architecture configuration information into AML models

To integrate OPC Unified Architecture configuration information into AML models, the OPC Unified Architecture server and its variables shall be modelled within AML models (see Figure 11).

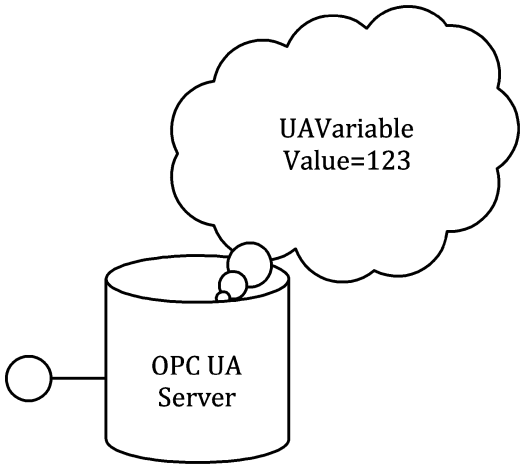


Figure 11 — OPC Unified Architecture configuration information

The following provisions apply to the description of OPC Unified Architecture configuration information:

- The OPC Unified Architecture *server* shall be modelled as an element.
- The AML model element representing the OPC Unified Architecture *server* shall include descriptive attributes containing the following information:
  - *DiscoveryURL*
  - *EndpointURL*
  - *TransportProfileURI*
  - *SecurityPolicy*
  - *MessageSecurityMode*
  - *UserToken*
  - *NamespaceTable*
- The *variables* of the OPC Unified Architecture *server* shall be modelled in relation to the properties of the manufacturing system elements they belong to.
- The AML model element representing the OPC Unified Architecture *variable* shall reference the ID of the AML model element of the OPC Unified Architecture *server*.
- The AML model element representing the OPC Unified Architecture *variable* shall include descriptive attributes containing the following information:
  - *NodeId*

The concept for modelling this information in AML will be explained in the IEC 62714 series of standards.

7 Relation to other standards and specifications

There are different ways to create relations to other standards and specifications, namely:

1. Relation to elements which exist in the OPC Unified Architecture information model:
   - Relation is realized in AML via AML ExternalInterface of InterfaceClass AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector or derivations (see IEC 62714-1and -4).
   - If a related element exists in OPC Unified Architecture model (e.g. IEC 61131, Figure 12), a *Reference* of type "HasAMLUAReference" should be created between the two *Node*s, see 5.7.

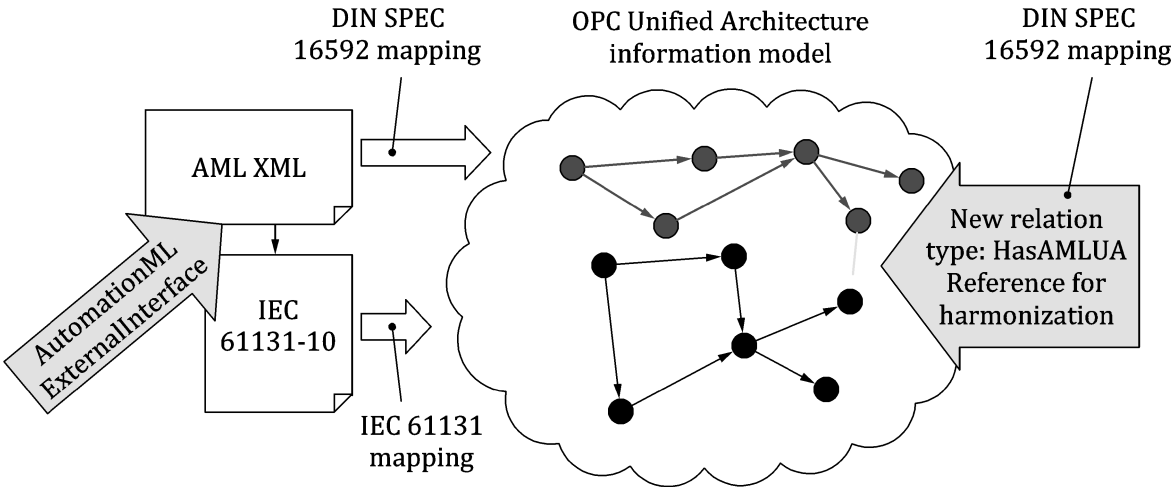


Figure 12 — Relation between *Node*s of transformed *NodeSet*s in OPC Unified Architecture information model

2. Relation to elements which do not exist in the OPC Unified Architecture information model:
   - Relation is realized in AML via AML ExternalInterface of InterfaceClass AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector or derivations (e.g. IEC 62714-1and -3, BPR External Data Reference [19], see Figure 13).
   - The related element exists in the OPC Unified Architecture model.

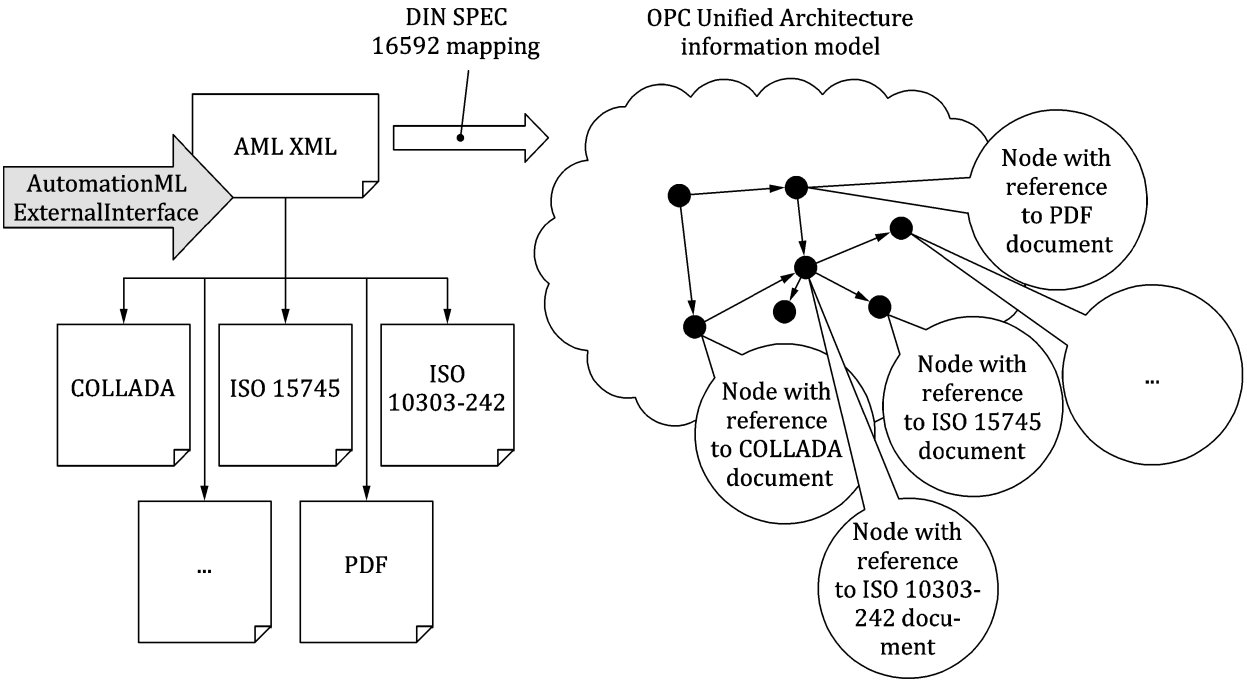


Figure 13 — Relation to elements which do not exist in OPC Unified Architecture information model

3. Relation to semantics which do not exist in the OPC Unified Architecture information model:
   - Relation is realized in AML via referencing special RoleClasses for InternalElements and use of RefSemantic element of a specific Attribute (e.g. Whitepaper AutomationML and eCl@ss integration [20], via RefSemantic, see Figure 14).
   - The related element exists in the OPC Unified Architecture model.

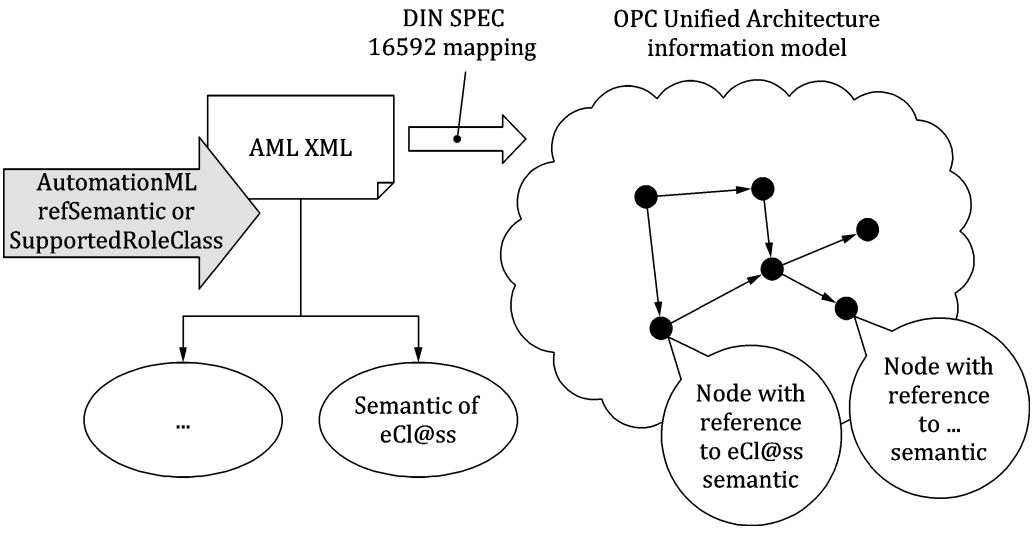


Figure 14 — Relation to semantics which do not exist in OPC Unified Architecture information model

Annex A
(informative)

Industrial application

A.1 General

The specification describes the combination of AML information models with OPC Unified Architecture online information communication, like process data and diagnostic information. It extends the application domain of OPC Unified Architecture (see Figure A.1) and AML. Therefore, different use cases which will be possible by the combination of both standards were identified. Use cases are described hereinafter in a general form listing actors, a brief description with the corresponding process flow, the benefit of the use case, and optionally a usage scenario.

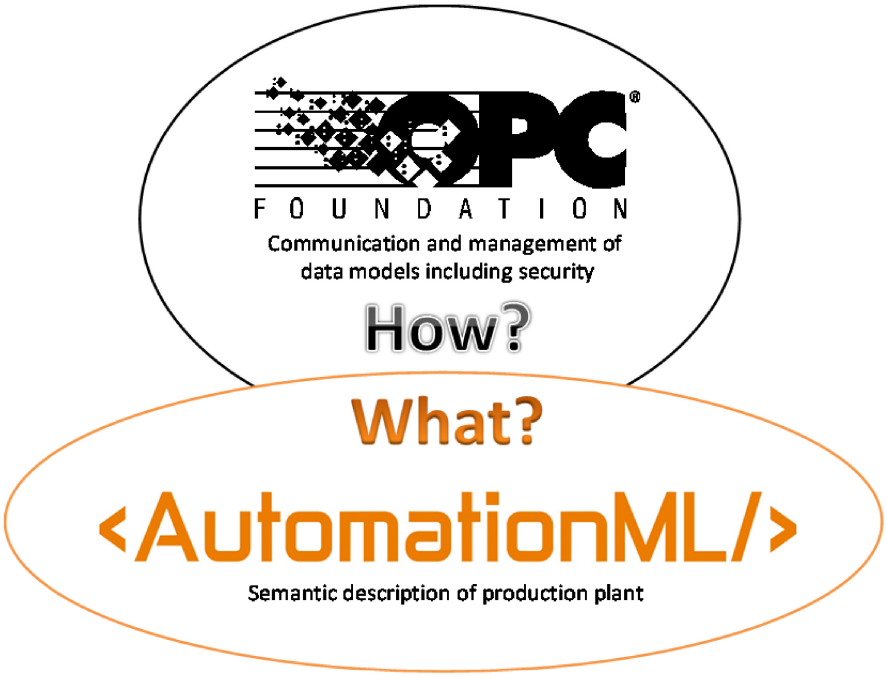


Figure A.1 — Goals of AML and OPC Unified Architecture

One possibility of combining AML and OPC Unified Architecture is to communicate and operationalize AML by means of OPC Unified Architecture. It is possible to simplify the creation of OPC Unified Architecture information models based on existing AML data. This is realized by an OPC Unified Architecture companion specification due to analogies between AML and the OPC Unified Architecture information model. The companion specification for AML consists of an object model including many specific semantics which are exchanged via OPC Unified Architecture.

Data management, online communication functionality, multi-user support, access methods, and security mechanisms[3] as specified by OPC Unified Architecture are used when combining AML with OPC Unified Architecture.

Important use cases to be considered are listed in sections A.2, A.3, A.4, A.6, A.8 and A.10.

Another possibility is to exchange the OPC Unified Architecture system configuration within AML models. The manual exchange of OPC Unified Architecture server configuration data is replaced by a specified description in AML. Parameters for setting up OPC Unified Architecture communication between tools are exchanged using AML. This realizes consistent data, produces less errors, and results in an easier and faster configuration of OPC Unified Architecture clients.

OPC Unified Architecture benefits from the description of the communication network configuration and structure including communication system parameters, network system topology, quality of service, passive and active infrastructure elements.

Major use cases to be considered are listed in section A.5, A.7 and A.9.

There is another overall use case which combines both types of integration as described above: Plug-and-work. Plug-and-work is defined as the "setting up, modification or termination of interoperation between two or more involved parties with minimal effort" [14]. Requirements and steps towards plug-and-work by means of OPC Unified Architecture and Automation Markup Language are described in [15]. Objects that can be changed within a manufacturing system can be "products, technological or logistical processes, parts of the manufacturing facilities or a company's organization" [15]. IT systems are also objects to change – they have to be adapted to reflect changes to products and facilities on the shop floor. Today, the adaption of IT systems is carried out manually. Automating the modification of production IT systems over the lifecycle of the manufacturing system is a focus of the present specification. In the context of plug-and-work, AML describes the content, i.e. that which is exchanged between the systems involved. It serves to model plants and plant components, including their skills, topology, interfaces and relation to each other, geometry, kinematics, and even logic and behaviour. It provides a description of components that can be used as the basis for an automatic comparison and selection of components based on the description of their provided skills/functions. See [15] for details. OPC Unified Architecture as standardized communication middleware for automation systems serves as a bridge between offline-based engineering tasks and the runtime communication of the systems involved. It is used to exchange information necessary for plug-and-work between the systems, and deals with data management and communication management including reliability, security, and an information model to include object-oriented descriptions. It provides a standardized access to the components and the possibility to interact with and control the component.

The use cases listed in sections A.2 to A.10 are related to each other. This relation is depicted in Figure A.2; the correlation is classified by means of the lifecycle of manufacturing systems.

[3] Security mechanisms are defined in IEC 62541-2.

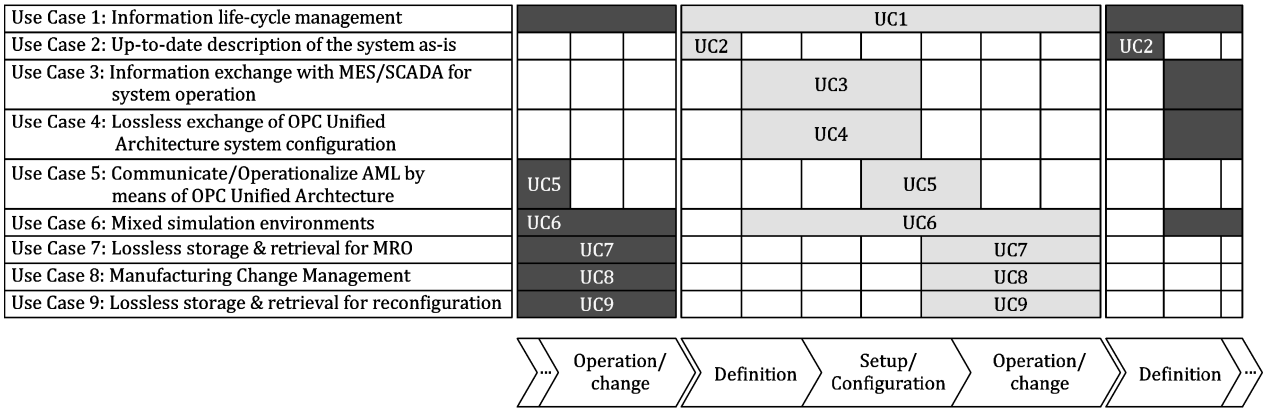


Figure A.2 — Relations between use cases over the lifecycle

The overview of all actors and their relation to the use cases is depicted in Table A.1.

Table A.1 — Actors and related use cases

	Use Case 1: Information life-cycle management	Use Case 2: Up-to-date description of the system as-is	Use Case 3: Information exchange (e. g. asset information, quality information, diagnostic data, etc.) with MES or SCADA system for system operation	Use Case 4: Lossless exchange of OPC Unified Architecture system configuration	Use Case 5: Communicate/Operationalize AML by means of OPC Unified Architecture	Use Case 6: Mixed simulation environments	Use Case 7: Lossless storage and retrieval of system engineering information for system maintenance, repair, overhaul (MRO)	Use Case 8: Manufacturing Change Management	Use Case 9: Lossless storage and retrieval of system engineering information for manufacturing reconfiguration
Mechanical engineer	x	x		x	x		x	x	x
Electrical engineer	x	x		x	x		x	x	x
Software developer (PLC , HMI, Robot, distributed control system (DCS), network, etc.)	x	x		x	x	x	x	x	x
Plant operator	x		x	x		x	x	x	
SCADA system/MES provider, IT integrator			x	x				x	

Maintenance personnel					x		x	x	
Commissioner	x	x			x			x	
Plant/Factory Planner	x							x	

In general, requirements for the operational infrastructure are defined individually depending on the application, and taking into account the non-functional requirements specified in ISO/IEC 25000, e.g. performance, usablility, IT security, etc. For each application, these non-functional requirements may be defined for actor, system, server, or client.

A.2 Use case 1: Information lifecycle management

Information about the current engineering and build status of a technical system can change over the lifetime of the system of interest. Starting with a high level description of the manufacturing system at the beginning of the manufacturing system engineering phase, the description of the manufacturing system is increasingly given greater detail until the manufacturing system is completely planned by the end of the engineering. During the installation and ramp-up of the manufacturing system additional information about the as-built status can be integrated into the engineering documents, resulting in a complete manufacturing system documentation that could be regarded as Digital Twin. This documentation can change over the use phase of the manufacturing system due to maintenance, technical changes, reengineering of the manufacturing system within the configuration management process. One main aim of the joint use of AML and OPC Unified Architecture is the provision of the current status of the information on a manufacturing system as-is or as-built over the complete lifecycle of the manufacturing system.

1. Actors:
    a. mechanical engineer;
    b. electrical engineer;
    c. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.);
    d. plant operator;
    e. commissioner;
    f. plant/factory planner;
2. Brief description with process flow (see Figure A.3);
    a. initially, the set of information on the manufacturing system is empty.
    b. in the first step the general engineering project is set up in a repository residing on an OPC Unified Architecture server (or set of OPC Unified Architecture servers with a centralized entry, possibly including other storage media) providing a kind of empty project information set.
    c. in the second step information relevant to the manufacturing system is generated. Therefore, all required information for the information generation process is read out of the project information set by accessing the OPC Unified Architecture server.

        NOTE This information is to be accessible by monitoring systems, which will provide the information to the use case actors.

    d. after information is generated it is integrated into the repository residing on an OPC Unified Architecture server, ensuring information consistency.

        NOTE Steps c) and d) can be repeated as often as necessary.

        NOTE The activities for information creation can be any engineering activity, any installation activity, any use activity, any maintenance activity, etc. related to the manufacturing system.

        NOTE The ability of actors to access the information can change over time.
3. Benefits:
    a. Centralized access to all information on a manufacturing system on an as-is or as-built basis. This will enable all engineers, regardless of their discipline or intention, to work with up-to-

date and consistent information, and provide higher quality results in their activities while requiring less effort to retrieve information about the manufacturing system.

b. The system's failures can be traced.

4. Scenarios (optional):
   a. within steps a) and c) the repository used to store the information can change over the lifecycle of a manufacturing system. During the engineering and installation phases this repository may reside on a central server of the engineering organization. At the end of the installation phase (at the latest, during the ramp-up of the manufacturing system) the repository will be migrated to an OPC Unified Architecture server.
   b. if possible, the version history should be saved.

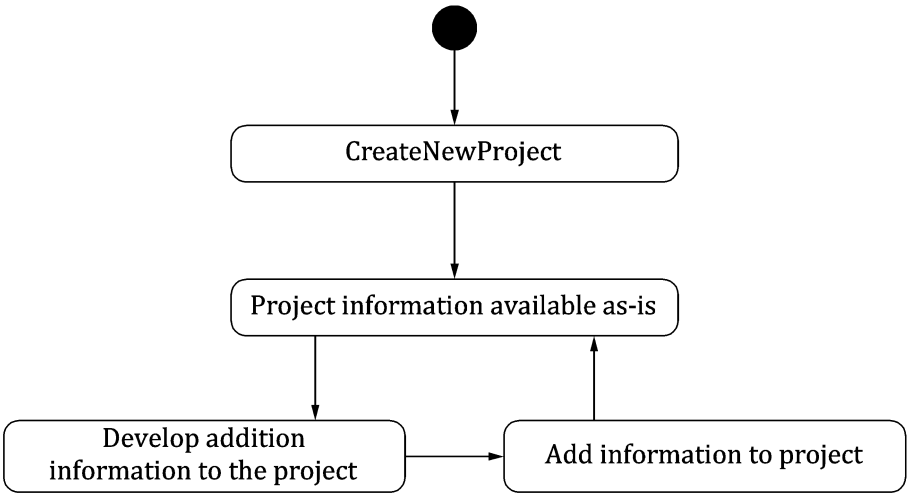


Figure A.3 — Process flow of use case 1

A.3 Use case 2: Up-to-date description of the system as-is

The manufacturing system is described in (several) proprietary models of different manufacturing engineering applications, which are combined into a unified model in AML. This model is enriched with current process data by means of OPC Unified Architecture to update the AML models.

1. Actors:
   a. mechanical engineer;
   b. electrical engineer;
   c. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.);
   d. commissioner;
2. Brief description with process flow:
   a. gGeneral process flow:
      i. engineers from different disciplines (see actors) model the engineering information for the available system components and store them locally in the manufacturing engineering applications.
      ii. the data is combined into a unified model in AML and stored in one or more XML files.
      iii. the AML model is transferred into an information model of an OPC Unified Architecture server which is accessible via a network connection.
      iv. this OPC Unified Architecture model is updated with the current field configuration information.

v. this engineering information is made available to the actors in their specific manufacturing engineering applications.

b. process flow for device programming (sub-process of the generic process flow above):
   i. The Commissioner defines configurations of controller and peripheral equipment, e.g. creating a proprietary device description.
   ii. the resulting device description is referenced in the AML model.
   iii. the device description[4] and the reference to this description are transferred into an information model of an OPC Unified Architecture server which is accessible via a network connection.
   iv. this OPC Unified Architecture model is updated with the current field configuration information including device and controller parameters.
   v. this engineering information is made available to the actors (see list), e.g. PLC programmer.

3. Benefits:
   a. up-to-date description of the system as-is;
   b. provision of information from operation to planning, e.g. used ports of PLC for electrical- and software engineering data;
   c. human creates model in engineering;
   d. system/machine enriches model from current system;
   e. reasonability of configuration, fusion, and correction;
   f. adjustment of plant information and engineering data, e.g. in case of device exchange;
   g. updated data available (push guarantees deployment of new configuration data).

A.4 Use Case 3: Information exchange (e.g. asset information, quality information, diagnostic data, etc.) with MES or SCADA systems for system operation

Data about plant topology, current process values, control parameters, and communication interfaces is provided to plant operators for the operation of MES or SCADA systems. This information is used for quality management and diagnosis, for example.

1. Actors:
   a. plant operator;
   b. SCADA system/MES provider, IT integrator;
2. Brief description with process flow:
   a. Provide access to engineering information included in the AML model during operation phase. It can contain additional information about the following aspects which relate to the quality of the process and products or the assets themselves:
      i. hierarchical plant topology including relations between components;
      ii. communication interfaces;
      iii. current process values, operational states, and calculated KPI;
      iv. control parameters and reference values.
3. Benefits:
   a. The engineering information can be used for diagnosis and analysis as well as process optimization to be related to the process signals/variables.
   b. Simplified integration and update of (reconfigured) manufacturing systems in MES/SCADA systems.

[4] Device descriptions may be described with other OPC Unfied Architecture companion specifications, e.g. FDI, or other device description standards

A.5 Use Case 4: Lossless exchange of OPC Unified Architecture system configuration

Lossless exchange of description of parameters for seting up OPC Unified Architecture communication using AML. Explicit description of parameters for the configuration and setup of OPC Unified Architecture clients is used to simplify this process. This description is included in AML models.

1. Actors:
   a. mechanical engineer;
   b. electrical engineer;
   c. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.);
   d. plant operator;
   e. SCADA system/MES provider, IT integrator.
2. Brief description with process flow:
   a. creation of OPC Unified Architecture system configuration using an appropriate tool (for example Electrical Design, PLC Programming, HMI configuration, Robot Programming, Vision system setup, etc.);
   b. export of OPC Unified Architecture system configuration (OPC Unified Architecture communication partners (server endpoint, server profile, transport protocol, …), variable *NodeId*s) via file format in AML model;
   c. exchange of configuration in AML model;
   d. import of entire configuration (e.g. via configuration file) in AML model via file format into OPC Unified Architecture client configuration tool to set up communication.
3. Benefits:
   a. manual exchange of configuration data will replaced by standardized/specified description in AML;
   b. automated OPC Unified Architecture client configuration and automated communication setup;
   c. ewer errors and easier and faster configuration of OPC Unified Architecture servers and clients.

A.6 Use Case 5: Communicate/Operationalize AML by means of OPC Unified Architecture

Within the maintenance of manufacturing system maintenance engineers usually require three sets of information: information about the current status of the production equipment, particularly including sensor and actuator states, information about the manufacturing system setup, and information about the engineering and installation of the manufacturing system. These information sets need to be interlinked so that service personnel can use them to identify possible problems within the manufacturing systems.

1. Actors:
   a. mechanical engineer;
   b. electrical engineer;
   c. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.);
   d. maintenance personnel;
   e. commissioner.
2. Brief description with a process flow (see Figure A.4):
   a. initially it is assumed that the current status of the automation devices of the manufacturing system is available within an OPC Unified Architecture server. In addition, it is assumed that the manufacturing system engineering and installation information is available via an OPC Unified Architecture server under security as defined in OPC Unified Architecture (see IEC 62541-2).

Finally, it is assumed, that data relating to a sensor or actuator within the engineering and installation information is linked to the corresponding status of the automation devices and vice versa.

NOTE Automation devices also include sub-devices and their information

b. in the first step the service personnel exports the necessary status of the automation devices of interest or exports the relevant engineering data out of the engineering data set. Data is converted into AML and transferred to OPC Unified Architecture. Export AML model or parts of it when needed in tools.

c. in the second step the service personnel use the link from the current status data to the related engineering data to read out the relevant engineering data and aggregate it, or use the link from the engineering data to the necessary status of the automation devices of interest to read out the status information.

NOTE This action should be simplified.

d. service personnel make decisions based on this aggregated engineering and installation and runtime information.

3. Benefits:

a. this use case enables service personnel to make informed decisions regarding necessary maintenance activities without needing to acquire additional data. This can include information about the current calibration state of sensors, dependencies between sensors and actuators, handbook access, and detailed replacement information;

b. wider acceptance and usability of both standards (AML and OPC Unified Architecture);

c. object model including many specific semantics, which can be used online with multiple involved parties/disciplines/tools by OPC Unified Architecture;

d. additional OPC Unified Architecture communication principles are available.

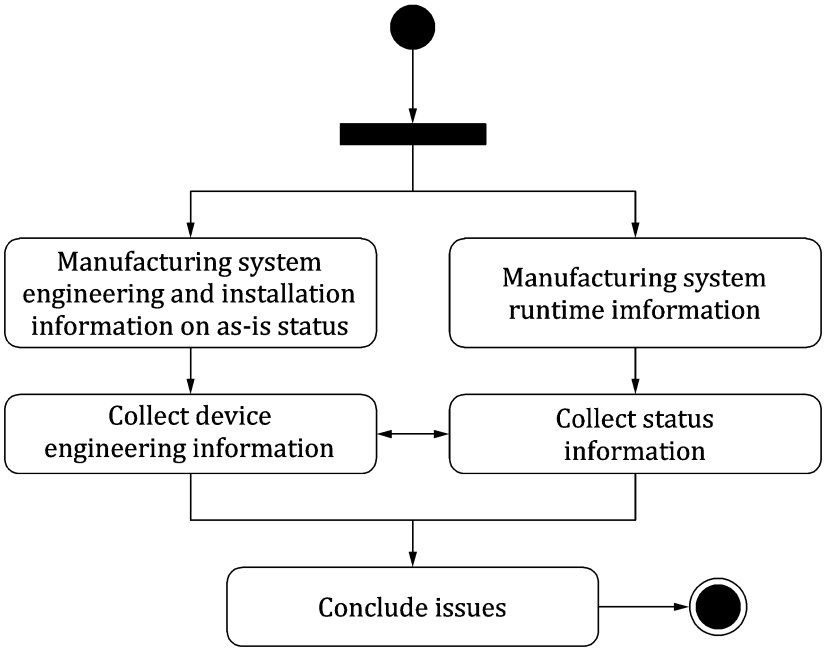


Figure A.4 — Process flow of use case 5

A.7 Use Case 6: Mixed simulation environments

Current production environments involve a tremendous amount of virtual and real data for products (e.g. geometrical data), processes (e.g. cycle time), and resources (e.g. energy consumption). The aim is to realize a digital twin of the real world and to enable a seamless flow of information between the planning and executing domains in order to make virtual try-out or simulations of mixed virtual and real data possible. In this manner concurrent mixed simulation environments can be realized – for mixed hardware-in-the-loop and software-in-the-loop simulation, validation, and virtual commissioning. This can be used e.g. in the field of cooperative and collaborative robots (CoBots) or worker-specific 360° ergonomics.

1. Actors:
    a. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.);
    b. plant operator.
2. Brief description with process flow:
    a. engineering provides a virtual representation of a product (including requirements and validation and test results) via exchange format, e.g. STEP AP 242 XML and JT, to production planning;
    b. the interconnection to the machines via OPC Unified Architecture is used for hardware-in-the-loop and/or software-in-the-loop tests – in production planning as well as in engineering;
    c. based on this, production planning realizes and simulates the virtual assembly process, checks feasibilities, optimizes sequences and times;
    d. results are communicated backward (to engineering) and forward (to production) via an exchange format, e.g. AML combined with other standards (see chapter 7).
3. Benefits:
    a. acceleration of the ramp-up phase and increase of production stability through robust and optimized processes (e.g. by reduced media breaks and increased data quality based on planning and field data);
    b. acceleration of production planning (e.g. virtual validation and commissioning), production ramp-up and downtime;
    c. acceleration of change phases based on simulation without interrupting operational processes;
    d. a configurable production planning can be realized, so that one product can be manufactured in several facilities with diverging machine parks;
    e. through the established information flows, field and process data can be fed back to production planning and engineering online.

A.8 Use Case 7: Lossless storage and retrieval of system engineering information for system maintenance, repair, overhaul (MRO)

Engineering information about the reconfigured manufacturing system is used for maintenance, repair, and overhaul (MRO) (see DIN 31051:2003, DIN EN 13460:2009)

1. Actors:
    a. mechanical engineer;
    b. electrical engineer;
    c. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.);
    d. plant operator;
    e. maintenance personnel;

NOTE KPIs for maintenance, repair, overhaul should be based on PLC implementation and possibly provided via OPC Unified Architecture to maintenance, monitoring and planning systems.

f. the as-is description (optional) could be enriched with information for maintenance, repair, and overhaul (wiring diagrams, assembly and disassembly manuals or figures, maintenance plans, maintenance manuals, manuals, trouble shooting guides, bill of materials, spare part lists, tubing diagrams, test cases and procedures, etc.) and be made accessible (e.g. directly integrated in the OPC Unified Architecture server *Namespace* or referenced);

g. the description of the system as-is includes all sub-devices (in hierarchical form) and their information for maintenance, repair, and overhaul.

2. Brief description with process flow:
   a. maintenance of manufacturing systems:
      i. in case of maintenance of manufacturing systems, a detailed description of the system as-is, and topology information of components and AutomationML sub-configurations are provided to plant operators and service personnel.
      ii. the AutomationML model is updated (e.g. new components, component properties, new communication interfaces).
   b. repair of manufacturing systems:
      i. in case of repairs, the failed or badly worn components are replaced or serviced;
      ii. the AutomationML model is updated (e.g. new serial numbers of changed components, date of repair, changes in mechanic, electric, PLC configuration and OLP).
   c. overhaul of manufacturing system
      i. in case of overhauls, the changes in electric, mechanical and controller code / robot programs are updated by service personnel;
      ii. ihe AutomationML model is updated (e.g. changes in mechanic, electric, PLC configuration).
3. Benefits:
   a. handover of description of the system as-is, including information for maintenance, repair,, and overhaul in digital form synchronizing system and system description lifecycles, reducing possible retyping errors.

A.9 Use Case 8: Manufacturing change management

Production planning processes begin at an early stage of product development. Continually changing conditions, particularly on the shop floor, pose a challenge to the production planning process. Manufacturing Change Management (MCM) (see [17]) addresses the relevant steps for managing these changes and thereby ensures transparency and efficiency. Corresponding information is stored in AML documents and is exchanged via OPC Unified Architecture servers. Version management is dicussed in A.2

Use case 1: Information lifecycle management. This helps to capture undocumented changes and can be applied as support for Continuous Improvement Processes (CIP) or a Company Suggestion System.

1. Actors:
   a. mechanical engineer;
   b. electrical engineer;
   c. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.);
   d. plant operator;
   e. SCADA system/MES provider, IT integrator;
   f. maintenance personnel;
   g. commissioner;
   h. plant/factory planner.

2. Brief description with process flow (see Figure A.5):
    a. manufacturing Change Request (MCR):
        i. capturing:
        An actor (initiator) creates a change enquiry. Actors are in-house divisions such as product development and manufacturing, or assembly planning, or external parties such as suppliers or customers. The change request is then created, the planning objects and responsible parties affected are identified, and the change request is incorporated into the superordinate and central Manufacturing Change List. The corresponding information is stored in an AML model.
        ii. evaluation:
        The objective of this evaluation is to make an informed decision about whether a change order is issued based on the information stored in the AML model. The AML model is exchanged via an OPC Unified Architecture server with the different actors involved (relevant decision-makers as well as the people involved)
    b. manufacturing Change Order (MCO):
        i. documentation, execution:

        This step comprises the re-planning or updating of the product plan and associated planning documentation. This applies to the detailed planning documents, digital models, and analyses as well as the documentation required for the shop floor (work plan etc.). This is followed by the actual implementation of the change and the final confirmation by the responsible parties. Within the Manufacturing Change List the change is then marked as “completed”. The corresponding information is stored in an AML model.
3. Benefits:
    a. basis for synchronized documentation of production processes;
    b. specified and transparent processes for implementation of changes in the production into the planning documentation;
    c. transparent and efficient implementation of new production processes at the shop floor;
    d. reduction of planning iterations based on skills and knowledge.

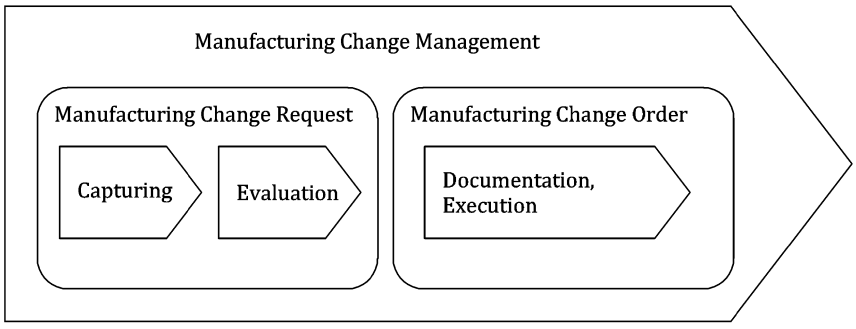


Figure A.5 — Process flow for Use case 8

A.10 Use case 9: Lossless storage and retrieval of system engineering information for manufacturing system reconfiguration

During the reconfiguration of manufacturing systems, existing system components are reused. Therefore, a detailed description of the manufacturing system and its components as-is is required (and e.g. retrieved from the system itself or another storage location).

1. Actors:
    a. mechanical engineer;
    b. electrical engineer;
    c. software developer (PLC, HMI, Robot, distributed control system (DCS), network, etc.).
2. Brief description with process flow:
    a. the AML model of the manufacturing system is updated (in terms of change in geometric design, electric design, controller (PLC), HMI, or robot programms);
    b. the updated AML data is used for validation and verification of manufacturing system engineering information;
    c. the updated AML data is transferred to the OPC Unfied Architecture server;
    d. usage of the information: see Use case 1.
3. Benefits:
    a. reduction of effort for modelling information used in simulation and verification
    b. enabling frontloading in redesigns and/or changes;
    c. enabling decision support, based on real facts.
4. Scenarios (optional):
    a. cerification and simulation of commissioned manufacturing system under consideration of current configuration in shop floor (e.g. due to process improvements or changes under control of manufacturing change management);
    b. transfer from design to set-up / commissioning (update of AutomationML model by design/ planning) for following PLC programming and commissioning;
        i. consistency of mechanical, electrical, PLC code and OLP data,
    c. alignment/verification of “As-planned”/“as-set-up” (alignment/verification of AutomationML model with current setup configuration of manufacturing system)
    d. consistency of mechanical, electrical, PLC code and OLP data. Transfer from setup to planning (manufacturing system updates AutomationML-model planning/design retrieves updated information)
        i. consistency of mechanical, electrical, PLC code and OLP data,
    e. transfer from setup to MES-/SCADA (manufacturing system updates AutomationML model → MES-/SCADA systems use updated AutomationML model for manufacturing execution / superiviserior control and data acquisition)
    f. simulation and optimization in case of reconfigured manufacturing system for
        i. processes (manufacturing, assembly),
        ii. process/cycle times,
        iii. material flow,
        iv. PLC code (virtual commissioning),
        v. robot movements, trajectories, kinematics.

Annex B
(informative)

Mapping example

Figure B.2 depicts the tree structure of an AML example. This example shows a simple AML document (version 2.0) which contains

- one InstanceHierarchy with InternalElements,
- one SystemUnitClassLib with one SystemUnitClass,
- one RoleClassLib with one RoleClass, and
- one InterfaceClassLib with one InterfaceClass.

The elements in this example are designed to model some aspects of a manufacturing system. The RoleClassLib contains a RoleClass to characterize a system component as a tool. The InterfaceClassLib contains an InterfaceClass which models the energy supply of any tool. The SystemUnitClassLib contains a SystemUnitClass representing an electric screwdriver. The InstanceHierarchy describes a manufacturing system containing the two screwdrivers in the system.

The AML libraries used (AutomationMLBaseRoleClassLibrary and AutomationMLInterfaceClassLibrary) are stored in separated CAEX files and are included via ExternalReferences.

Figure B.1 contains the XML text of the example in AML.

```
<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="Topology.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="./Source/CAEX_ClassModel_V2.15.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <AdditionalInformation AutomationMLVersion="2.0"/>
    <AdditionalInformation>
        <WriterHeader>
            <WriterName>AutomationML e.V.</WriterName>
            <WriterID>AutomationML e.V.</WriterID>
            <WriterVendor>AutomationML e.V.</WriterVendor>
            <WriterVendorURL>www.AutomationML.org</WriterVendorURL>
            <WriterVersion>1.0</WriterVersion>
            <WriterRelease>1.0</WriterRelease>
            <LastWritingDateTime>2012-02-20
            </LastWritingDateTime>
            <WriterProjectTitle>AutomationML Tutorial Examples</WriterProjectTitle>
            <WriterProjectID>AutomationML Tutorial Examples
            </WriterProjectID>
        </WriterHeader>
    </AdditionalInformation>
    <ExternalReference Path="Libs/RoleClass Libraries/AutomationMLBaseRoleClassLib.aml"
Alias="BaseRoleClassLib"/>
    <ExternalReference Path="Libs/InterfaceClass Libraries/AutomationMLInterfaceClassLib.aml"
Alias="BaseInterfaceClassLib"/>
    <InstanceHierarchy Name="ManufacturingSystem">
        <InternalElement Name="firstScrewdriver" ID="{788eb291-f103-4fdc-aba0-4893b599f556}"
RefBaseSystemUnitPath="LibOfCommonTools/ElectricScrewdriver">
            <Attribute Name="New Attribute"/>
            <ExternalInterface Name="EnergySupply" ID="{5f535d4c-dd46-4c1c-898c-4e58419048b6}"
RefBaseClassPath="MyInterfaces/Energy"/>
            <SupportedRoleClass RefRoleClassPath="ManufacturingRoleClasses/Tool"/>
            <RoleRequirements RefBaseRoleClassPath="ManufacturingRoleClasses/Tool"/>
        </InternalElement>
        <InternalElement Name="secondScrewdriver" ID="{19dcf818-4716-4fc1-a85f-28e1938c4c3a}"
RefBaseSystemUnitPath="LibOfCommonTools/ElectricScrewdriver">
            <ExternalInterface Name="EnergySupply" ID="50e10905-ac18-413c-afab-ad8ed1569fff"
RefBaseClassPath="MyInterfaces/Energy"/>
            <ExternalInterface Name="PLCopenXMLInterface_instance_1"
RefBaseClassPath="BaseInterfaceClassLib@AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalData
```

```
Connector/PLCopenXMLInterface" ID="0e2014a2-fb61-44c8-8025-7e3161dfece4">
                  <Attribute Name="refURI" AttributeDataType="xs:anyURI">
                      <Value>file:///OPCopen.xml#ID1
</Value>
                  </Attribute>
              </ExternalInterface>
              <SupportedRoleClass RefRoleClassPath="ManufacturingRoleClasses/Tool"/>
              <RoleRequirements RefBaseRoleClassPath="ManufacturingRoleClasses/Tool"/>
         </InternalElement>
     </InstanceHierarchy>
     <InterfaceClassLib Name="MyInterfaces">
         <Version>1.0</Version>
         <InterfaceClass Name="Energy"
RefBaseClassPath="BaseInterfaceClassLib@AutomationMLInterfaceClassLib/AutomationMLBaseInterface"/>
     </InterfaceClassLib>
     <RoleClassLib Name="ManufacturingRoleClasses">
         <Version>1.0</Version>
         <RoleClass Name="Tool"
RefBaseClassPath="BaseRoleClassLib@AutomationMLBaseRoleClassLib/AutomationMLBaseRole"/>
     </RoleClassLib>
     <SystemUnitClassLib Name="LibOfCommonTools">
         <Version>1.0</Version>
         <SystemUnitClass Name="ElectricScrewdriver">
             <ExternalInterface Name="EnergySupply" ID="dd0e0dfe-10f8-4068-845b-9c29699ac79b"
RefBaseClassPath="MyInterfaces/Energy"/>
             <SupportedRoleClass RefRoleClassPath="ManufacturingRoleClasses/Tool"/>
         </SystemUnitClass>
     </SystemUnitClassLib>
</CAEXFile>
```

Figure B.1 — AML XML text

The example shown in Figure B.2 will now be redefined within OPC Unified Architecture.

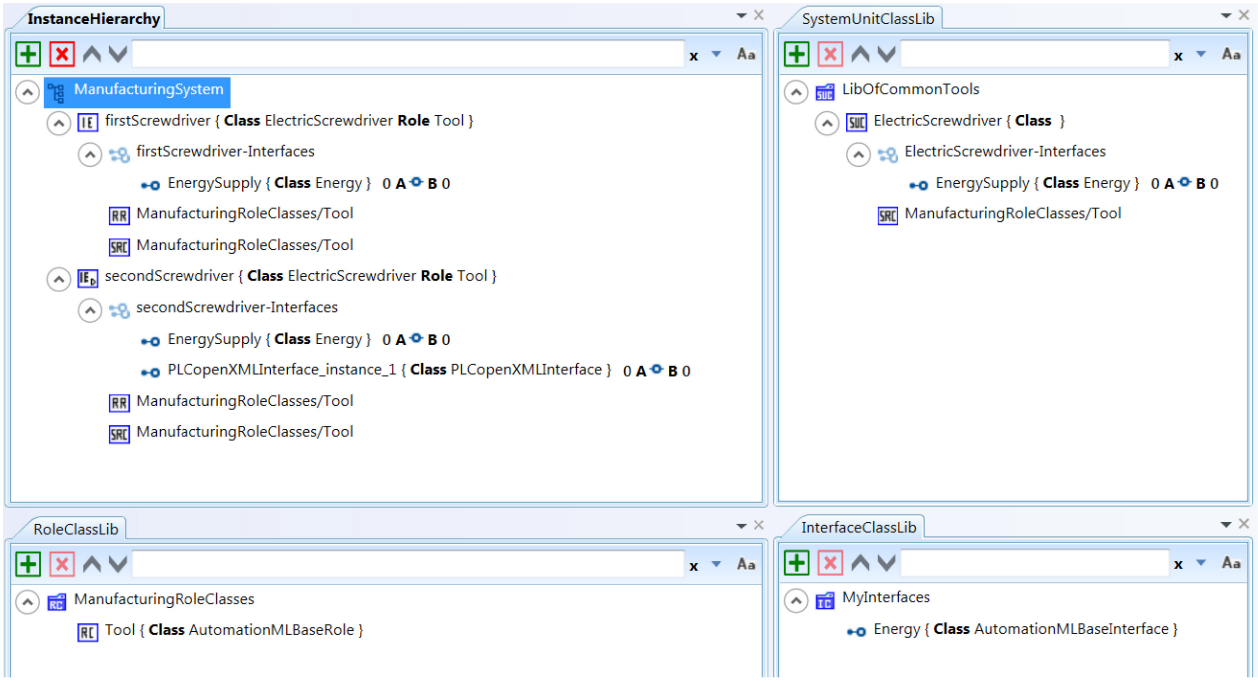


Figure B.2 — AML example

Figure B.3 contains the XML text of the example in OPC Unified Architecture.

```
<?xml version="1.0" encoding="utf-8"?>
<UANodeSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://opcfoundation.org/UA/2011/03/UANodeSet.xsd">
  <NamespaceUris>
    <Uri>http://opcfoundation.org/UA/AML/</Uri>
    <Uri>http://www.iosb.fraunhofer.de/Topology.aml</Uri>
  </NamespaceUris>
  <Aliases>
    <Alias Alias="Boolean">i=1</Alias>
    <Alias Alias="SByte">i=2</Alias>
    <Alias Alias="Byte">i=3</Alias>
    <Alias Alias="Int16">i=4</Alias>
    <Alias Alias="UInt16">i=5</Alias>
    <Alias Alias="Int32">i=6</Alias>
    <Alias Alias="UInt32">i=7</Alias>
    <Alias Alias="Int64">i=8</Alias>
    <Alias Alias="UInt64">i=9</Alias>
    <Alias Alias="Float">i=10</Alias>
    <Alias Alias="Double">i=11</Alias>
    <Alias Alias="DateTime">i=13</Alias>
    <Alias Alias="String">i=12</Alias>
    <Alias Alias="ByteString">i=15</Alias>
    <Alias Alias="Guid">i=14</Alias>
    <Alias Alias="XmlElement">i=16</Alias>
    <Alias Alias="NodeId">i=17</Alias>
    <Alias Alias="ExpandedNodeId">i=18</Alias>
    <Alias Alias="StatusCode">i=19</Alias>
    <Alias Alias="QualifiedName">i=20</Alias>
    <Alias Alias="LocalizedText">i=21</Alias>
    <Alias Alias="Structure">i=22</Alias>
    <Alias Alias="Number">i=26</Alias>
    <Alias Alias="Integer">i=27</Alias>
    <Alias Alias="UInteger">i=28</Alias>
    <Alias Alias="Organizes">i=35</Alias>
    <Alias Alias="HasEventSource">i=36</Alias>
```

```
    <Alias Alias="HasModellingRule">i=37</Alias>
    <Alias Alias="HasEncoding">i=38</Alias>
    <Alias Alias="HasDescription">i=39</Alias>
    <Alias Alias="HasTypeDefinition">i=40</Alias>
    <Alias Alias="HasSubtype">i=45</Alias>
    <Alias Alias="HasProperty">i=46</Alias>
    <Alias Alias="HasComponent">i=47</Alias>
    <Alias Alias="HasNotifier">i=48</Alias>
    <Alias Alias="FolderType">i=61</Alias>
    <Alias Alias="PropertyType">i=68</Alias>
    <Alias Alias="Mandatory">i=78</Alias>
    <Alias Alias="Optional">i=80</Alias>
    <Alias Alias="Objects">i=85</Alias>
    <Alias Alias="Duration">i=290</Alias>
  </Aliases>
  <UAObject NodeId="ns=2;i=1" BrowseName="ManufacturingSystem">
    <DisplayName>ManufacturingSystem</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=2</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5002</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=45</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=12</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=15</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=3" BrowseName="MyInterfaces">
    <DisplayName>MyInterfaces</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=4</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=5</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5004</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=48</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=31</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=6" BrowseName="ManufacturingRoleClasses">
    <DisplayName>ManufacturingRoleClasses</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=7</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=8</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5005</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=47</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=33</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=9" BrowseName="LibOfCommonTools">
    <DisplayName>LibOfCommonTools</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=10</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=11</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5006</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=46</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=35</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=12" BrowseName="firstScrewdriver">
    <DisplayName>firstScrewdriver</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=13</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=14</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=1</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=2;i=35</Reference>
      <Reference ReferenceType="ns=1;i=4001">ns=2;i=33</Reference>
      <Reference ReferenceType="ns=1;i=4002">ns=2;i=33</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=18</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=20</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=15" BrowseName="secondScrewdriver">
    <DisplayName>secondScrewdriver</DisplayName>
    <Description></Description>
```

```
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=16</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=17</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=1</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=2;i=35</Reference>
      <Reference ReferenceType="ns=1;i=4001">ns=2;i=33</Reference>
      <Reference ReferenceType="ns=1;i=4002">ns=2;i=33</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=23</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=26</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=20" BrowseName="EnergySupply" ParentNodeId="ns=2;i=12">
    <DisplayName>EnergySupply</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=21</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=22</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=12</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=2;i=31</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=23" BrowseName="EnergySupply" ParentNodeId="ns=2;i=15">
    <DisplayName>EnergySupply</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=24</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=25</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=15</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=2;i=31</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=26" BrowseName="PLCopenXMLInterface_instance_1" ParentNodeId="ns=2;i=15">
    <DisplayName>PLCopenXMLInterface_instance_1</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=27</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=28</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=15</Reference>
      <Reference ReferenceType="HasTypeDefinition">i=</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=29</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=37" BrowseName="EnergySupply" ParentNodeId="ns=2;i=35">
    <DisplayName>EnergySupply</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=38</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=39</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=35</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=2;i=31</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=40" BrowseName="Topology.aml">
    <DisplayName>Topology.aml</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=41</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=1;i=1002</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=42</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=43</Reference>
      <Reference ReferenceType="HasProperty">ns=2;i=44</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=45</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=46</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=47</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=48</Reference>
      <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5001</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=45" BrowseName="InstanceHierarchies" ParentNodeId="ns=2;i=40">
    <DisplayName>InstanceHierarchies</DisplayName>
    <References>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=40</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=1</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=46" BrowseName="SystemUnitClassLibs" ParentNodeId="ns=2;i=40">
    <DisplayName>SystemUnitClassLibs</DisplayName>
```

```
    <References>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=40</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=9</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=47" BrowseName="RoleClassLibs" ParentNodeId="ns=2;i=40">
    <DisplayName>RoleClassLibs</DisplayName>
    <References>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=40</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=6</Reference>
    </References>
  </UAObject>
  <UAObject NodeId="ns=2;i=48" BrowseName="InterfaceClassLibs" ParentNodeId="ns=2;i=40">
    <DisplayName>InterfaceClassLibs</DisplayName>
    <References>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=40</Reference>
      <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=3</Reference>
    </References>
  </UAObject>
  <UAVariable NodeId="ns=2;i=2" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=1"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=4" BrowseName="Version" ParentNodeId="ns=2;i=3" DataType="String">
    <DisplayName>Version</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">1.0</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=5" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=3"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=7" BrowseName="Version" ParentNodeId="ns=2;i=6" DataType="String">
    <DisplayName>Version</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">1.0</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=8" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=6"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=10" BrowseName="Version" ParentNodeId="ns=2;i=9" DataType="String">
    <DisplayName>Version</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">1.0</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=11" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=9"
```

```
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=13" BrowseName="ID" ParentNodeId="ns=2;i=12" DataType="String">
    <DisplayName>ID</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">{788eb291-f103-4fdc-aba0-
4893b599f556}</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=14" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=12"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=16" BrowseName="ID" ParentNodeId="ns=2;i=15" DataType="String">
    <DisplayName>ID</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">{19dcf818-4716-4fc1-a85f-
28e1938c4c3a}</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=17" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=15"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=18" BrowseName="New Attribute" ParentNodeId="ns=2;i=12" DataType="String"
AccessLevel="15" UserAccessLevel="15">
    <DisplayName>New Attribute</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=19</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=1;i=3007</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=12</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=19" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=18"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=21" BrowseName="ID" ParentNodeId="ns=2;i=20" DataType="String">
    <DisplayName>ID</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
```

```
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">{5f535d4c-dd46-4c1c-898c-
4e58419048b6}</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=22" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=20"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=24" BrowseName="ID" ParentNodeId="ns=2;i=23" DataType="String">
    <DisplayName>ID</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">50e10905-ac18-413c-afab-
ad8ed1569fff</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=25" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=23"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=27" BrowseName="ID" ParentNodeId="ns=2;i=26" DataType="String">
    <DisplayName>ID</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">0e2014a2-fb61-44c8-8025-
7e3161dfece4</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=28" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=26"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=29" BrowseName="refURI" ParentNodeId="ns=2;i=26" DataType="String"
AccessLevel="15" UserAccessLevel="15">
    <DisplayName>refURI</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=30</Reference>
      <Reference ReferenceType="HasTypeDefinition">ns=1;i=3007</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=26</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">file:///OPCopen.xml#ID1
</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=30" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=29"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=32" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=31"
DataType="String">
```

```
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=34" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=33"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=36" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=35"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=38" BrowseName="ID" ParentNodeId="ns=2;i=37" DataType="String">
    <DisplayName>ID</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">dd0e0dfe-10f8-4068-845b-
9c29699ac79b</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=39" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=37"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd" />
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=41" BrowseName="AdditionalInformation" ParentNodeId="ns=2;i=40"
DataType="String">
    <DisplayName>AdditionalInformation</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <ListOfString xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">
        <String>&lt;AdditionalInformation AutomationMLVersion="2.0" /&gt;</String>
        <String>&lt;AdditionalInformation&gt;&lt;WriterHeader&gt;&lt;WriterName&gt;AutomationML
e.V.&lt;/WriterName&gt;&lt;WriterID&gt;AutomationML e.V.&lt;/WriterID&gt;&lt;WriterVendor&gt;AutomationML
e.V.&lt;/WriterVendor&gt;&lt;WriterVendorURL&gt;www.AutomationML.org&lt;/WriterVendorURL&gt;&lt;WriterVersio
n&gt;1.0&lt;/WriterVersion&gt;&lt;WriterRelease&gt;1.0&lt;/WriterRelease&gt;&lt;LastWritingDateTime&gt;2012-
02-20
            &lt;/LastWritingDateTime&gt;&lt;WriterProjectTitle&gt;AutomationML Tutorial
Examples&lt;/WriterProjectTitle&gt;&lt;WriterProjectID&gt;AutomationML Tutorial Examples
            &lt;/WriterProjectID&gt;&lt;/WriterHeader&gt;&lt;/AdditionalInformation&gt;</String>
      </ListOfString>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=42" BrowseName="SchemaVersion" ParentNodeId="ns=2;i=40" DataType="String">
    <DisplayName>SchemaVersion</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">2.15</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=43" BrowseName="BaseRoleClassLib" ParentNodeId="ns=2;i=40" DataType="String">
```

```
    <DisplayName>BaseRoleClassLib</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">Libs/RoleClass
Libraries/AutomationMLBaseRoleClassLib.aml</String>
    </Value>
  </UAVariable>
  <UAVariable NodeId="ns=2;i=44" BrowseName="BaseInterfaceClassLib" ParentNodeId="ns=2;i=40"
DataType="String">
    <DisplayName>BaseInterfaceClassLib</DisplayName>
    <References>
      <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
    </References>
    <Value>
      <String xmlns="http://opcfoundation.org/UA/2008/02/Types.xsd">Libs/InterfaceClass
Libraries/AutomationMLInterfaceClassLib.aml</String>
    </Value>
  </UAVariable>
  <UAObjectType NodeId="ns=2;i=31" BrowseName="Energy">
    <DisplayName>Energy</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=32</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=3</Reference>
      <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=1003</Reference>
    </References>
  </UAObjectType>
  <UAObjectType NodeId="ns=2;i=33" BrowseName="Tool">
    <DisplayName>Tool</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=34</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=6</Reference>
      <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=1003</Reference>
    </References>
  </UAObjectType>
  <UAObjectType NodeId="ns=2;i=35" BrowseName="ElectricScrewdriver">
    <DisplayName>ElectricScrewdriver</DisplayName>
    <Description></Description>
    <References>
      <Reference ReferenceType="HasProperty">ns=2;i=36</Reference>
      <Reference ReferenceType="HasComponent" IsForward="false">ns=2;i=9</Reference>
      <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=1003</Reference>
      <Reference ReferenceType="ns=1;i=4001">ns=2;i=33</Reference>
      <Reference ReferenceType="HasComponent">ns=2;i=37</Reference>
    </References>
  </UAObjectType>
</UANodeSet>
```

Figure B.3 — OPC Unified Architecture XML text

Figure B.4 depicts the main structure of the example including the folder *Objects* for the organization of the *AdressSpace*. The graphical notation used has been defined by the OPC Foundation. It includes the OPC Unified Architecture *Objects* used to organize the *AdressSpace* structure, which are explained in 5.5.5.

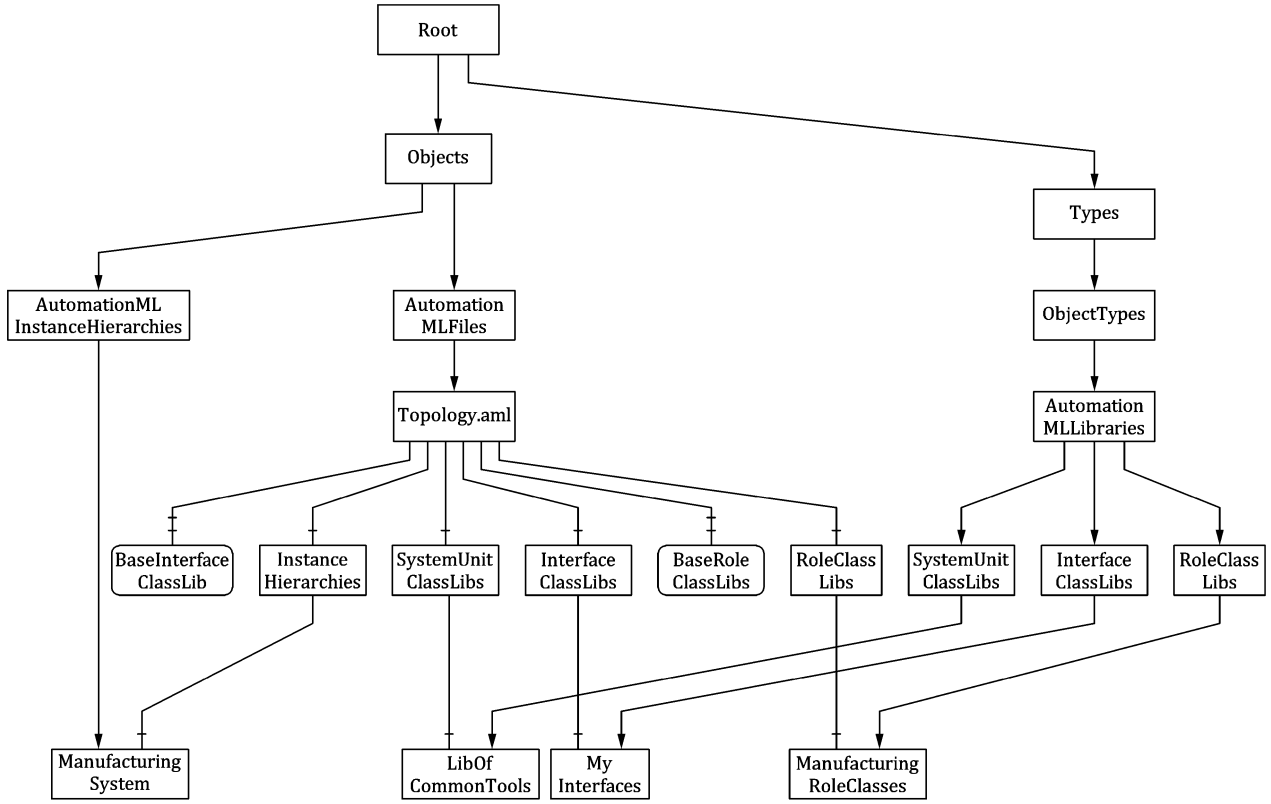


Figure B.4 —Example: Main structure in OPC Unified Architecture

Figure B.5 depicts the role classes of the example. It includes the inheritance structure of the roles. In the RoleClassLib "ManufacturingRoleClasses" the RoleClass "Tool" is included.

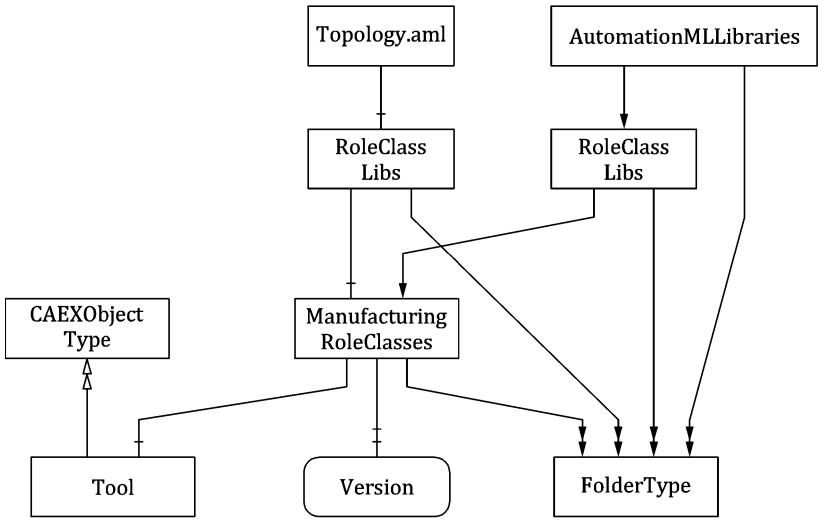


Figure B.5 — Example of role classes in OPC Unified Architecture

Figure B.6 shows the SystemUnitClasses. The SystemUnitClassLib "LibOfCommonTools" includes a SystemUnitClass "ElectricScrewdriver" with reference to the role "Tool". This SystemUnitClass includes an ExternalInterface "EnergySupply".

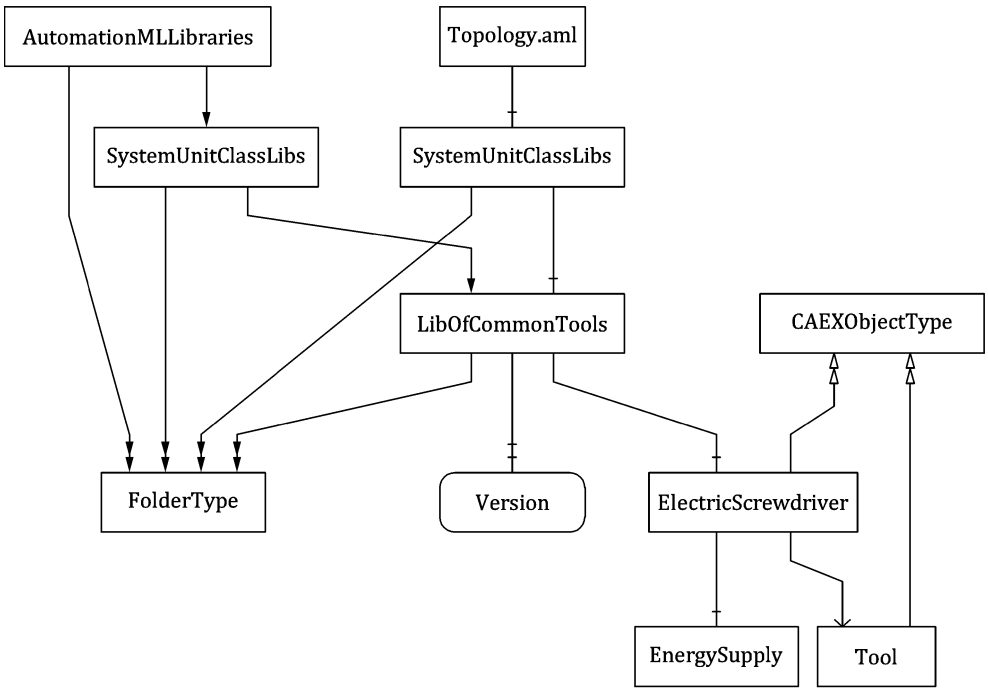


Figure B.6 — Example of system unit classes in OPC Unified Architecture

Figure B.7 shows the InterfaceClassLib "MyInterfaces", including the InterfaceClass "Energy".

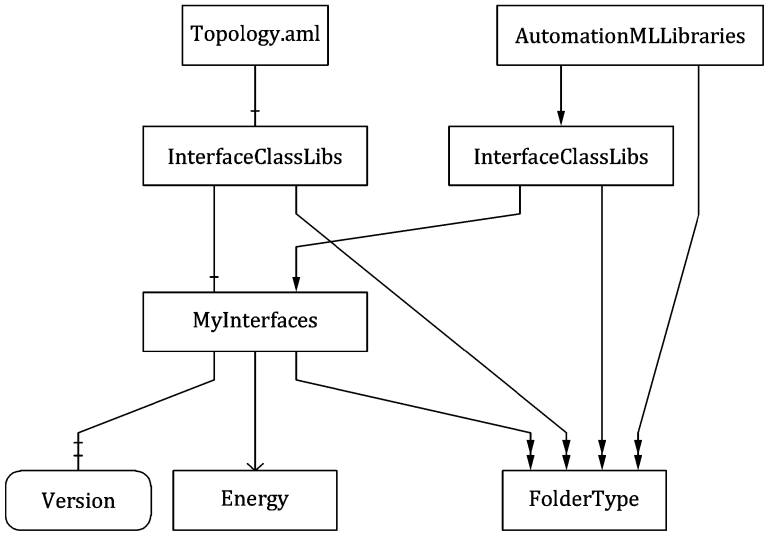


Figure B.7 — Example of interface classes in OPC Unified Architecture

The mapping of the InstanceHierarchy is shown in Figure B.8. The file "Topology.aml" includes the InstanceHierarchy "ManufacturingSystem" which includes two InternalElements "firstScrewdriver" and "secondScrewdriver". They are derived from the SystemUnitClass "ElectricScrewdriver" and inherit its structure and elements, such as a Variable "ID", an ExternalInterface "EnergySupply", and the assigned

RoleClass “Tool”. The InternalElement “firstScrewdriver” additionally consists of an Attribute “NewAttribute”. The InternalElement “secondScrewdriver” additionally consists of an ExternalInterface “PLCopenXMLInterface_instance_1” which references an external PLCopenXML file.

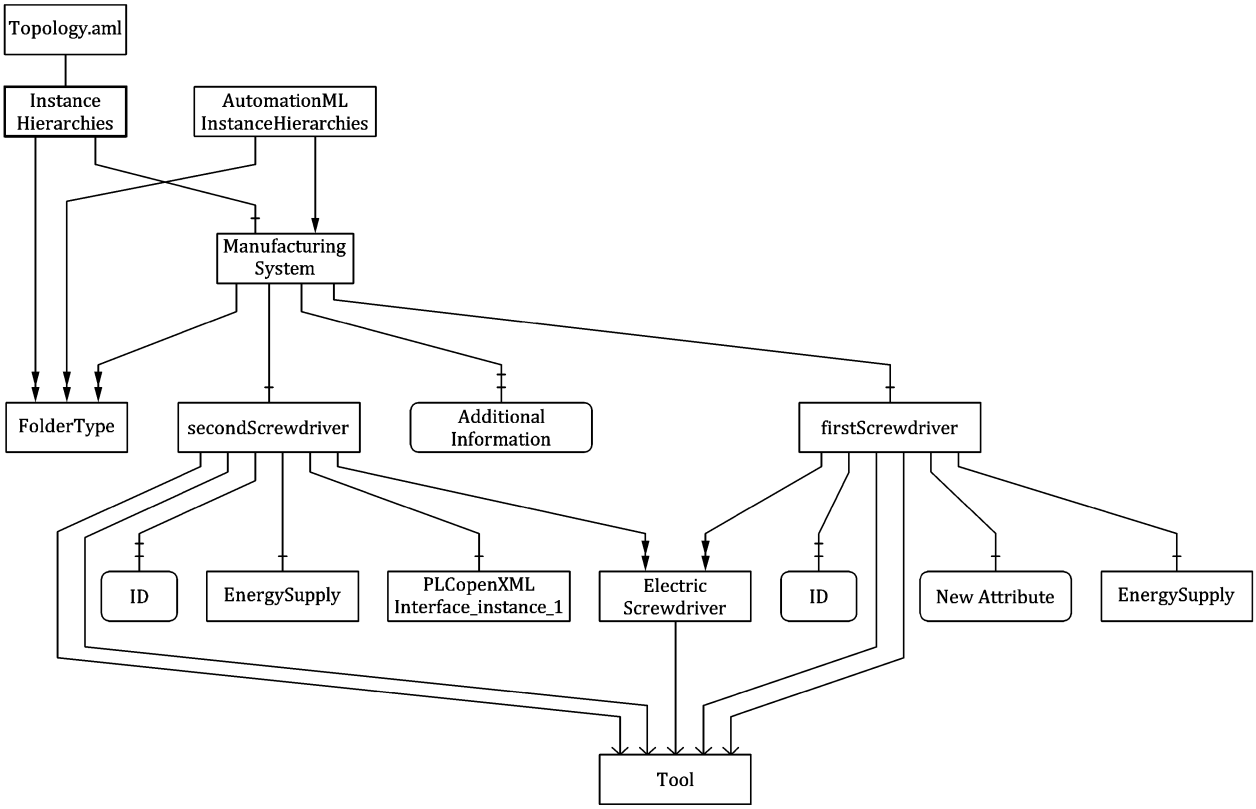


Figure B.8 — Example of InstanceHierarchy in OPC Unified Architecture

Annex C
(informative)

AML base types NodeSet


```
<?xml version="1.0" encoding="utf-8"?>
<UANodeSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://opcfoundation.org/UA/2011/03/UANodeSet.xsd"
Version="0.1.6">
    <NamespaceUris>
        <Uri>http://opcfoundation.org/UA/AML/</Uri>
    </NamespaceUris>
    <Aliases>
        <Alias Alias="Boolean">i=1</Alias>
        <Alias Alias="SByte">i=2</Alias>
        <Alias Alias="Byte">i=3</Alias>
        <Alias Alias="Int16">i=4</Alias>
        <Alias Alias="UInt16">i=5</Alias>
        <Alias Alias="Int32">i=6</Alias>
        <Alias Alias="UInt32">i=7</Alias>
        <Alias Alias="Int64">i=8</Alias>
        <Alias Alias="UInt64">i=9</Alias>
        <Alias Alias="Float">i=10</Alias>
        <Alias Alias="Double">i=11</Alias>
        <Alias Alias="DateTime">i=13</Alias>
        <Alias Alias="String">i=12</Alias>
        <Alias Alias="ByteString">i=15</Alias>
        <Alias Alias="Guid">i=14</Alias>
        <Alias Alias="XmlElement">i=16</Alias>
        <Alias Alias="NodeId">i=17</Alias>
        <Alias Alias="ExpandedNodeId">i=18</Alias>
        <Alias Alias="StatusCode">i=19</Alias>
        <Alias Alias="QualifiedName">i=20</Alias>
        <Alias Alias="LocalizedText">i=21</Alias>
        <Alias Alias="Structure">i=22</Alias>
        <Alias Alias="Number">i=26</Alias>
        <Alias Alias="Integer">i=27</Alias>
        <Alias Alias="UInteger">i=28</Alias>
        <Alias Alias="Organizes">i=35</Alias>
        <Alias Alias="HasEventSource">i=36</Alias>
        <Alias Alias="HasModellingRule">i=37</Alias>
        <Alias Alias="HasEncoding">i=38</Alias>
        <Alias Alias="HasDescription">i=39</Alias>
        <Alias Alias="HasTypeDefinition">i=40</Alias>
        <Alias Alias="HasSubtype">i=45</Alias>
        <Alias Alias="HasProperty">i=46</Alias>
        <Alias Alias="HasComponent">i=47</Alias>
        <Alias Alias="HasNotifier">i=48</Alias>
        <Alias Alias="FolderType">i=61</Alias>
        <Alias Alias="PropertyType">i=68</Alias>
        <Alias Alias="Mandatory">i=78</Alias>
        <Alias Alias="Optional">i=80</Alias>
        <Alias Alias="Objects">i=85</Alias>
        <Alias Alias="Duration">i=290</Alias>
    </Aliases>
    <UAObject NodeId="ns=1;i=5001" BrowseName="1:AutomationMLFiles" ParentNodeId="">
        <DisplayName>AutomationMLFiles</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="Organizes" IsForward="false">Objects</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5002" BrowseName="1:AutomationMLInstanceHierarchies" ParentNodeId="">
        <DisplayName>AutomationMLInstanceHierarchies</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="Organizes" IsForward="false">Objects</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5003" BrowseName="1:AutomationMLLibraries" ParentNodeId="">
        <DisplayName>AutomationMLLibraries</DisplayName>
        <Description>The browse entry point when looking for AutomationML libraries in the server address
space.</Description>
```

```
        <References>
            <Reference ReferenceType="Organizes" IsForward="false">i=88</Reference>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="Organizes">ns=1;i=5004</Reference>
            <Reference ReferenceType="Organizes">ns=1;i=5005</Reference>
            <Reference ReferenceType="Organizes">ns=1;i=5006</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5004" BrowseName="1:InterfaceClassLibs" ParentNodeId="">
        <DisplayName>InterfaceClassLibs</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5003</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5005" BrowseName="1:RoleClassLibs" ParentNodeId="">
        <DisplayName>RoleClassLibs</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5003</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5006" BrowseName="1:SystemUnitClassLibs" ParentNodeId="">
        <DisplayName>SystemUnitClassLibs</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="Organizes" IsForward="false">ns=1;i=5003</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5007" BrowseName="1:InstanceHierarchies" ParentNodeId="ns=1;i=1002">
        <DisplayName>InstanceHierarchies</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1002</Reference>
            <Reference ReferenceType="HasModellingRule">Mandatory</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5008" BrowseName="1:InterfaceClassLibs" ParentNodeId="ns=1;i=1002">
        <DisplayName>InterfaceClassLibs</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1002</Reference>
            <Reference ReferenceType="HasModellingRule">Mandatory</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5009" BrowseName="1:RoleClassLibs" ParentNodeId="ns=1;i=1002">
        <DisplayName>RoleClassLibs</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="HasModellingRule">Mandatory</Reference>
            <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1002</Reference>
        </References>
    </UAObject>
    <UAObject NodeId="ns=1;i=5010" BrowseName="1:SystemUnitClassLibs" ParentNodeId="ns=1;i=1002">
        <DisplayName>SystemUnitClassLibs</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">FolderType</Reference>
            <Reference ReferenceType="HasModellingRule">Mandatory</Reference>
            <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1002</Reference>
        </References>
    </UAObject>
    <UAVariable NodeId="ns=1;i=6001" BrowseName="1:Version" ParentNodeId="ns=1;i=1001" DataType="String">
        <DisplayName>Version</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=1001</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6002" BrowseName="1:Copyright" ParentNodeId="ns=1;i=1001" DataType="String">
        <DisplayName>Copyright</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=1001</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6003" BrowseName="1:AdditionalInformation" ParentNodeId="ns=1;i=1001"
```

```
DataType="String" ValueRank="1">
        <DisplayName>AdditionalInformation</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=1001</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6004" BrowseName="1:SchemaVersion" ParentNodeId="ns=1;i=1002"
DataType="String">
        <DisplayName>SchemaVersion</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=1002</Reference>
            <Reference ReferenceType="HasModellingRule">Mandatory</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6005" BrowseName="1:ID" ParentNodeId="ns=1;i=1003" DataType="String">
        <DisplayName>ID</DisplayName>
        <References>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=1003</Reference>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6006" BrowseName="1:Version" ParentNodeId="ns=1;i=3001" DataType="String">
        <DisplayName>Version</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=3001</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6007" BrowseName="1:Copyright" ParentNodeId="ns=1;i=3001" DataType="String">
        <DisplayName>Copyright</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=3001</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6008" BrowseName="1:AdditionalInformation" ParentNodeId="ns=1;i=3001"
DataType="String" ValueRank="1">
        <DisplayName>AdditionalInformation</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=3001</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6009" BrowseName="1:ID" ParentNodeId="ns=1;i=3007" DataType="String">
        <DisplayName>ID</DisplayName>
        <References>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=3007</Reference>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6010" BrowseName="1:Unit" ParentNodeId="ns=1;i=3007" DataType="String">
        <DisplayName>Unit</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=3007</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6011" BrowseName="1:DefaultValue" ParentNodeId="ns=1;i=3007"
DataType="String">
        <DisplayName>DefaultValue</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=3007</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAVariable NodeId="ns=1;i=6012" BrowseName="1:RefSemantic" ParentNodeId="ns=1;i=3007"
DataType="String" ValueRank="1">
        <DisplayName>RefSemantic</DisplayName>
        <References>
            <Reference ReferenceType="HasTypeDefinition">PropertyType</Reference>
```

```
            <Reference ReferenceType="HasProperty" IsForward="false">ns=1;i=3007</Reference>
            <Reference ReferenceType="HasModellingRule">Optional</Reference>
        </References>
    </UAVariable>
    <UAObjectType NodeId="ns=1;i=1001" BrowseName="1:CAEXBasicObjectType" IsAbstract="true">
        <DisplayName>CAEXBasicObjectType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">i=58</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6001</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6002</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6003</Reference>
        </References>
    </UAObjectType>
    <UAObjectType NodeId="ns=1;i=1002" BrowseName="1:CAEXFileType">
        <DisplayName>CAEXFileType</DisplayName>
        <References>
            <Reference ReferenceType="HasComponent">ns=1;i=5007</Reference>
            <Reference ReferenceType="HasComponent">ns=1;i=5008</Reference>
            <Reference ReferenceType="HasComponent">ns=1;i=5009</Reference>
            <Reference ReferenceType="HasComponent">ns=1;i=5010</Reference>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=1001</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6004</Reference>
        </References>
    </UAObjectType>
    <UAObjectType NodeId="ns=1;i=1003" BrowseName="1:CAEXObjectType">
        <DisplayName>CAEXObjectType</DisplayName>
        <References>
            <Reference ReferenceType="HasProperty">ns=1;i=6005</Reference>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=1001</Reference>
        </References>
    </UAObjectType>
    <UAVariableType NodeId="ns=1;i=3001" BrowseName="1:AMLBasicVariableType" ValueRank="-2"
IsAbstract="true">
        <DisplayName>AMLBasicVariableType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">i=62</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6006</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6007</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6008</Reference>
        </References>
    </UAVariableType>
    <UAVariableType NodeId="ns=1;i=3002" BrowseName="1:ExternalReferenceType" ValueRank="-1"
DataType="String">
        <DisplayName>ExternalReferenceType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=3001</Reference>
        </References>
    </UAVariableType>
    <UAVariableType NodeId="ns=1;i=3003" BrowseName="1:ConstraintType" IsAbstract="true" ValueRank="-2">
        <DisplayName>ConstraintType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=3001</Reference>
        </References>
    </UAVariableType>
    <UAVariableType NodeId="ns=1;i=3004" BrowseName="1:NominalConstraintType" DataType="String"
ValueRank="1">
        <DisplayName>NominalConstraintType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=3003</Reference>
        </References>
    </UAVariableType>
    <UAVariableType NodeId="ns=1;i=3005" BrowseName="1:OrdinalConstraintType" DataType="ns=1;i=2001">
        <DisplayName>OrdinalConstraintType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=3003</Reference>
        </References>
    </UAVariableType>
    <UAVariableType NodeId="ns=1;i=3006" BrowseName="1:UnknownConstraintType" DataType="String">
        <DisplayName>UnknownConstraintType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=3003</Reference>
        </References>
    </UAVariableType>
    <UAVariableType NodeId="ns=1;i=3007" BrowseName="1:AMLVariableType">
        <DisplayName>AMLVariableType</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">ns=1;i=3001</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6009</Reference>
```

```
            <Reference ReferenceType="HasProperty">ns=1;i=6010</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6011</Reference>
            <Reference ReferenceType="HasProperty">ns=1;i=6012</Reference>
        </References>
    </UAVariableType>
    <UADataType NodeId="ns=1;i=2001" BrowseName="1:OrdinalConstraintItem">
        <DisplayName>OrdinalConstraintItem</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">Structure</Reference>
        </References>
        <Definition>
            <Field Name="RequiredMaxValue" DataType="i=24">
                <Description/>
            </Field>
            <Field Name="RequiredValue" DataType="i=24" Value="0">
                <Description/>
            </Field>
            <Field Name="RequiredMinValue" DataType="i=24" Value="1">
                <Description/>
            </Field>
        </Definition>
    </UADataType>
    <UAReferenceType NodeId="ns=1;i=4001" BrowseName="1:HasAMLSupportedRoleClass">
        <DisplayName>HasAMLSupportedRoleClass</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">i=32</Reference>
        </References>
        <InverseName>IsSupportedRole</InverseName>
    </UAReferenceType>
    <UAReferenceType NodeId="ns=1;i=4002" BrowseName="1:HasAMLRoleRequirement">
        <DisplayName>HasAMLRoleRequirement</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">i=32</Reference>
        </References>
        <InverseName>IsRequiredRole</InverseName>
    </UAReferenceType>
    <UAReferenceType NodeId="ns=1;i=4003" BrowseName="1:HasAMLInternalLink" Symmetric="true">
        <DisplayName>HasAMLInternalLink</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">i=32</Reference>
        </References>
        <InverseName>HasAMLInternalLink</InverseName>
    </UAReferenceType>
    <UAReferenceType NodeId="ns=1;i=4004" BrowseName="1:HasAMLUAReference">
        <DisplayName>HasAMLUAReference</DisplayName>
        <References>
            <Reference ReferenceType="HasSubtype" IsForward="false">i=32</Reference>
        </References>
        <InverseName>IsAMLReferenceOf</InverseName>
    </UAReferenceType>
</UANodeSet>
```

Figure C.1 — AML base types NodeSet

Bibliography

[1] IEC 62714-3, *Engineering data exchange format for use in industrial automation systems engineering — Automation markup language — Part 3: Geometry and kinematics* (in preparation by SC 65E)

[2] IEC 62714-4, *Engineering data exchange format for use in industrial automation systems engineering — Automation markup language — Part 4: Logic and behaviour* (in preparation by SC 65E)

[3] IEC 61131-10, *PLCopen XML Exchange Format according to IEC 61131-3* (in preparation by SC 65E)

[4] IEC 62541-100:2015, *OPC Unified Architecture — Part 100: Device interface*

[5] IEC 62264-1:2013, *Enterprise-control system integration — Part 1: Models and terminology*

[6] IEC 61512-1:1997, *Batch control — Part 1: Models and terminology*

[7] ISO 10303-242:2014, *Industrial automation systems and integration — Product data representation and exchange — Part 242: Application protocol: Managed model-based 3D engineering*

[8] ISO 14306:2012, *Industrial automation systems and integration — JT file format specification for 3D visualization*

[9] ISO 15745 (all parts), *Industrial automation systems and integration — Open systems application integration framework*

[10] DIN 31051:2003, *Fundamentals of maintenance*

[11] DIN EN 13306:2008, *Maintenance — Maintenance terminology*

[12] DIN EN 13460:2009, *Maintenance — Documentation for maintenance*

[13] AMLUA, Companion specification "OPC Unified Architecture for AutomationML", https://opcfoundation.org/developer-tools/specifications-unified-architecture/opc-unified-architecture-for-automationml/, February 22, 2016.

[14] VDI, Thomas Bangemann, Christian Bauer, Heinz Bedenbender, Markus Diesner, Ulrich Epple, Filiz Elmas, Jens Friedrich, Thomas Goldschmidt, Florian Göbe, Sten Grüner, Martin Hankel, Roland Heidel, Klaus Hesselmann, Guido Hüttemann, Heinrich Kehl, Ulrich Löwen, Julius Pfrommer, Miriam Schleipen, Bastian Schlich, Thomas Usländer, Clemens Westerkamp, Albrecht Winter, Martin Wollschläger: Industrie 4.0 – Technical Assets. Basic terminology concepts, life cycles and administration models.
https://www.vdi.de/fileadmin/vdi_de/redakteur_dateien/gma_dateien/6092_PUB_E_TW_GMA_Status_Report_ZVEI_-_Industrie_4_0_-_Technical_Assets_Internet.pdf, March 2016

[15] Miriam Schleipen, Arndt Lüder, Olaf Sauer, Holger Flatt, Jürgen Jasperneite: Requirements and concept for Plug-and-Work - Flexibility in the context of Industry 4.0 (Anforderungen und Konzept for Plug-and-Work – Flexibilität im Kontext von Industrie 4.0). at - Automatisierungstechnik. Band 63, Heft 10, Seiten 801–820, ISSN (Online) 2196-677X, ISSN (Print) 0178-2312, DOI: 10.1515/auto-2015-0015, October 2015

[16] ProSTEP iViP Recommendation PSI 14-1: JT Industrial Application Package, V2.0, ISBN 978-3-9812689-9-7, June 2016

[17] ProSTEP iViP Recommendation PSI 12: Manufacturing Change Management, V1.1, ISBN 978-3-9812689-7-3, May 2015

[18] Robert Henssen, Miriam Schleipen: Interoperability between OPC-UA and AutomationML.Disruptive Innovation in Manufacturing Engineering towards the 4th Industrial Revolution, Proceedings of the 8th International CIRP Conference on Digital Enterprise Technology - DET 2014 mit CD-ROM, Hrsg.: Wilhelm Bauer, Carmen Constantinescu, Olaf Sauer, Paul Maropoulos, Jody Muelaner; Fraunhofer IAO, Stuttgart; 2014, Sprache: Englisch, Fraunhofer Verlag, ISBN 978-3-8396-0697-1, 2014. Full-text: Procedia CIRP, Volume 25, 2014, Pages 297–304, http://www.sciencedirect.com/science/article/pii/S2212827114010737, 2014

[19] Best Practice Recommendations: ExternalDataReference, https://www.automationml.org/o.red/uploads/dateien/1470211528-BPR_005E_ExternalDataReference_Jul2016.zip, State: July 2016

[20] White paper AutomationML and eCl@ss integration, https://www.automationml.org/o.red/uploads/dateien/1448438009-20151030_WP_AutomationML_and_eClass_integration_v1.0_neu.pdf, State: November 2015