

DIN EN IEC 62714-4

ICS 25.040.40; 35.060; 35.240.50

Einsprüche bis 2019-06-05

Entwurf

**Datenaustauschformat für Planungsdaten industrieller
Automatisierungssysteme –
Automation Markup Language (AML) –
Teil 4: Logik
(IEC 65E/619/CD:2018);
Text Deutsch und Englisch**

Engineering data exchange format for use in industrial automation systems engineering –
Automation Markup Language –
Part 4: Logic
(IEC 65E/619/CD:2018);
Text in German and English

Anwendungswarnvermerk

Dieser Norm-Entwurf mit Erscheinungsdatum 2019-04-05 wird der Öffentlichkeit zur Prüfung und Stellungnahme vorgelegt.

Weil die beabsichtigte Norm von der vorliegenden Fassung abweichen kann, ist die Anwendung dieses Entwurfs besonders zu vereinbaren.

Stellungnahmen werden erbeten

- vorzugsweise online im Norm-Entwurfs-Portal von DIN unter www.din.de/go/entwuerfe bzw. für Norm-Entwürfe der DKE auch im Norm-Entwurfs-Portal der DKE unter www.entwuerfe.normenbibliothek.de, sofern dort wiedergegeben;
- oder als Datei per E-Mail an dke@vde.com möglichst in Form einer Tabelle. Die Vorlage dieser Tabelle kann im Internet unter www.din.de/go/stellungnahmen-norm-entwuerfe oder für Stellungnahmen zu Norm-Entwürfen der DKE unter www.dke.de/stellungnahme abgerufen werden;
- oder in Papierform an die DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE, Stresemannallee 15, 60596 Frankfurt am Main.

Die Empfänger dieses Norm-Entwurfs werden gebeten, mit ihren Kommentaren jegliche relevanten Patentrechte, die sie kennen, mitzuteilen und unterstützende Dokumentationen zur Verfügung zu stellen.

Gesamtumfang 233 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE

Nationales Vorwort

Das internationale Dokument IEC 65E/619/CD:2018 „Engineering data exchange format for use in industrial automation systems engineering – Automation Markup Language – Part 4: Logic“ (CD, en: Committee Draft) ist unverändert in diesen Norm-Entwurf übernommen worden. Dieser Norm-Entwurf enthält eine noch nicht autorisierte deutsche Übersetzung.

Um Zweifelsfälle in der Übersetzung auszuschließen, ist die englische Originalfassung des CD entsprechend der diesbezüglich durch die IEC erteilten Erlaubnis beigefügt. Die Nutzungsbedingungen für den deutschen Text des Norm-Entwurfes gelten gleichermaßen auch für den englischen IEC-Text.

Das internationale Dokument wurde vom SC 65E „Devices and integration in enterprise systems“ der Internationalen Elektrotechnischen Kommission (IEC) erarbeitet und den nationalen Komitees zur Stellungnahme vorgelegt.

Die Internationale Elektrotechnische Kommission (IEC) und das Europäische Komitee für Elektrotechnische Normung (CENELEC) haben vereinbart, dass ein auf IEC-Ebene erarbeiteter Entwurf für eine Internationale Norm zeitgleich (parallel) bei IEC und CENELEC zur Umfrage (CDV-Stadium) und Abstimmung als FDIS (en: Final Draft International Standard) bzw. Schluss-Entwurf für eine Europäische Norm gestellt wird, um eine Beschleunigung und Straffung der Normungsarbeit zu erreichen. Dokumente, die bei CENELEC als Europäische Norm angenommen und ratifiziert werden, sind unverändert als Deutsche Normen zu übernehmen.

Da der Abstimmungszeitraum für einen FDIS bzw. Schluss-Entwurf FprEN nur 6 Wochen beträgt, und dann keine sachlichen Stellungnahmen mehr abgegeben werden können, sondern nur noch eine „JA/NEIN“-Entscheidung möglich ist, wobei eine „NEIN“-Entscheidung fundiert begründet werden muss, wird bereits der CD als DIN-Norm-Entwurf veröffentlicht, um die Stellungnahmen aus der Öffentlichkeit frühzeitig berücksichtigen zu können.

Für dieses Dokument ist das nationale Arbeitsgremium K 941 „Engineering“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE (www.dke.de) zuständig.

Für den Fall einer undatierten Verweisung im normativen Text (Verweisung auf ein Dokument ohne Angabe des Ausgabedatums und ohne Hinweis auf eine Abschnittsnummer, eine Tabelle, ein Bild usw.) bezieht sich die Verweisung auf die jeweils aktuellste Ausgabe des in Bezug genommenen Dokuments.

Für den Fall einer datierten Verweisung im normativen Text bezieht sich die Verweisung immer auf die in Bezug genommene Ausgabe des Dokuments.

Der Zusammenhang der zitierten Dokumente mit den entsprechenden deutschen Dokumenten ergibt sich, soweit ein Zusammenhang besteht, grundsätzlich über die Nummer der entsprechenden IEC-Publikation. Beispiel: IEC 60068 ist als EN 60068 als Europäische Norm durch CENELEC übernommen und als DIN EN 60068 ins Deutsche Normenwerk aufgenommen.

Es wird auf die Möglichkeit hingewiesen, dass einige Elemente dieses Dokuments Patentrechte berühren können. IEC ist nicht dafür verantwortlich, einige oder alle diesbezüglichen Patentrechte zu identifizieren.

Das Original-Dokument enthält Bilder und Tabellen in Farbe, die in der Papierversion in einer Graustufen-Darstellung wiedergegeben werden. Elektronische Versionen dieses Dokuments enthalten die Bilder und die Tabellen in der originalen Farbdarstellung.

Die Deutsche Fassung wurde in den Abschnitten 2 und 3 an die aktuellen Standardtexte angepasst.

Inhalt

Einleitung	10
1 Anwendungsbereich	12
2 Normative Verweisungen	14
3 Begriffe und Abkürzungen	14
3.1 Begriffe	14
3.2 Abkürzungen	16
4 Konformität	16
5 Teil 4 bezogene Erweiterungen der AML-Bibliotheken	16
5.1 Allgemeine Angaben	16
5.2 AutomationMLBaseRoleClassLib	16
5.2.1 Allgemeine Angaben	16
5.2.2 Rollenklasse InterlockingTargetGroup	17
5.2.3 Rollenklasse InterlockingSourceGroup	18
5.2.4 Rollenklasse LogicModelObject	19
5.3 AutomationMLInterfaceClassLib	20
5.3.1 Allgemeine Angaben	20
5.3.2 InterfaceClass PLCopenXMLInterface	20
5.3.3 Schnittstellenklasse VariableInterface	21
5.3.4 Schnittstellenklasse AMLLogicXMLInterface	21
5.3.5 Schnittstellenklasse LogicModelInterface	21
5.3.6 Schnittstellenklasse SequencingLogicModelInterface	22
5.3.7 Schnittstellenklasse BehaviourLogicModelInterface	22
5.3.8 Schnittstellenklasse InterlockingLogicModelInterface	23
5.3.9 Schnittstellenklasse LogicModelElementInterface	23
5.3.10 Schnittstellenklasse VariableInterface	23
5.3.11 Schnittstellenklasse InterlockingConnector	24
5.3.12 Schnittstellenklasse InterlockingVariableInterface	24
6 Metainformationen zu IML-Exportwerkzeugen	25
7 Referenzierung von AML logic XML Dokumenten	26
7.1 Allgemeine Angaben	26
7.2 Referenzierung von Logikinformationen	26
8 Beschreibung des AML logic XML Schemas	27
8.1 Überblick über das Schema	27
8.1.1 Verwendung des Schemas aus IEC 61131-10	27
8.1.2 Schemaversionierung	29
8.2 Root-Element „AMLLogic“	29
8.2.1 Allgemeine Angaben	29
8.3 Komplexer Typ „FunctionBlock“	30

8.3.1	Allgemeine Angaben	30
8.3.2	Komplexer Typ „ParameterSet“	31
8.3.3	Komplexer Typ „VariableDecl“	31
8.4	Komplexer Typ „IML“	32
8.4.1	Allgemeine Angaben	32
8.4.2	Komplexer Typ „IMLStep“	34
8.4.3	Komplexer Typ „IMLTransition“	35
8.4.4	Komplexer Typ „IMLSimultaneousDivergence“	35
8.4.5	Komplexer Typ „IMLSimultaneousConvergence“	35
8.5	Komplexer Typ „MathematicalExpression“	36
8.5.1	Allgemeine Angaben	36
8.6	Einfacher Typ „LogicModelTypeEnum“	36
8.7	Einfacher Typ „TimeUnion“	36
8.8	Einfacher Typ „TimeFormatEnum“	37
8.9	Einfacher Typ „UuidString“	37
9	Verknüpfung von AML-Objekten mit Verriegelungsinformationen	37
9.1	Allgemeine Angaben	37
9.2	Referenzierung von Verriegelungsinformationen	37
Anhang A (informativ) Logikinformationen in AML		39
A.1	Logikinformationen in der Planung von Produktionssystemen	39
A.2	Logikmodelle in der Planung von Produktionssystemen	40
A.3	Speicherung von Logikmodellen in der AML logic XML	42
A.4	Referenzierung von Logikinformationen in AML	45
Anhang B (informativ) Logikmodelle in AML		46
B.1	Allgemeine Angaben	46
B.2	Gantt-Diagramme	46
B.2.1	Allgemeine Angaben	46
B.2.2	Grafische Elemente	46
B.2.3	Diagrammstruktur	46
B.2.4	Logikinformationen	47
B.2.5	Logikinformationen in Gantt-Diagrammen	47
B.3	Vorgangsknotennetze	48
B.3.1	Allgemeine Angaben	48
B.3.2	Grafische Elemente	48
B.3.3	Knotenstruktur	49
B.3.4	Netzstruktur	49
B.3.5	Logikinformationen	49
B.3.6	Logikinformationen in Vorgangsknotennetzen	50
B.4	Zeitverlaufsdiagramme	51

B.4.1	Allgemeine Angaben	51
B.4.2	Grafische Elemente	51
B.4.3	Diagrammstruktur	51
B.4.4	Logikinformationen	52
B.4.5	Logikinformationen in Zeitverlaufsdigrammen	52
B.5	Ablaufsprache	53
B.6	Funktionsblockdiagramme	53
B.7	Mathematischer Ausdruck	53
Anhang C (normativ) Speicherung von Logikmodellen		54
C.1	Allgemeine Angaben	54
C.2	Speicherung von Gantt-Diagrammen in der AML logic XML	54
C.2.1	Allgemeine Regeln	54
C.2.2	Speicherung des Startpunkts eines Gantt-Diagramms	54
C.2.3	Speicherung von Balken	55
C.2.4	Speicherung von Pfeilen	56
C.2.5	Speicherung von Balken für nachfolgende Aktivitäten	56
C.2.6	Speicherung von Balken für vorhergehende Aktivitäten	58
C.3	Speicherung von Vorgangsknotennetzen in AML logic XML	59
C.3.1	Allgemeine Regeln	59
C.3.2	Speicherung des Startpunkts eines Vorgangsknotennetzes	60
C.3.3	Speicherung von Knoten	60
C.3.4	Speicherung von Pfeilen	61
C.3.5	Speicherung von Knoten für nachfolgende Aktivitäten	62
C.3.6	Speicherung von Knoten für vorhergehende Aktivitäten	63
C.4	Speicherung von Zeitverlaufsdigrammen in AML logic XML	64
C.4.1	Allgemeine Regeln	64
C.4.2	Speicherung der Zeitachse eines Zeitverlaufsdigramms	65
C.4.3	Speicherung von Ressourcen und Ressourcenzuständen	65
C.4.4	Speicherung von Lebenslinien	66
C.4.5	Speicherung von Zeit- und Ressourcensignalen	68
C.5	Speicherung der Ablaufsprache in AML logic XML	69
C.5.1	Allgemeine Regeln	69
C.5.2	Speicherung von Variablen	69
C.6	Speicherung von Funktionsblockdiagrammen in AML logic XML	70
C.6.1	Allgemeine Regeln	70
C.6.2	Speicherung von Variablen	70
C.7	Speicherung von mathematischen Ausdrücken in AML logic XML	71
C.7.1	Allgemeine Regeln	71
C.7.2	Speicherung von Variablen	72

C.7.3	Speicherung von Variablenzuordnungen	73
C.7.4	Speicherung von mathematischen Ausdrücken	73
Anhang D (informativ)	Beispiele für die Speicherung von Logikmodellen in der AML logic XML	74
D.1	Beispiele für die Speicherung von Gantt-Diagrammen	74
D.1.1	Speicherung von Aktivitäten ohne Vorgänger- und Nachfolger-Beziehung	74
D.1.2	Speicherung eines Ablaufs von Aktivitäten	75
D.1.3	Speicherung eines Ablaufs von Aktivitäten mit Divergenz	76
D.1.4	Speicherung eines Ablaufs von Aktivitäten mit Konvergenz	77
D.2	Beispiel für die Speicherung von Vorgangsknotennetzen	79
D.2.1	Speicherung von Aktivitäten ohne Vorgänger- und Nachfolger-Beziehung	79
D.2.2	Speicherung eines Ablaufs von Aktivitäten	80
D.2.3	Speicherung eines Ablaufs von Aktivitäten mit Divergenz	81
D.2.4	Speicherung eines Ablaufs von Aktivitäten mit Konvergenz	82
D.3	Beispiele für die Speicherung von Zeitverlaufsdigrammen	84
D.3.1	Beispiel für die Speicherung eines internen Signals	84
D.3.2	Beispiel für die Speicherung eines externen Signals	85
D.3.3	Beispiel für die Speicherung eines Signals zwischen zwei Zustandsflüssen einer Ressource	87
D.4	Beispiel für die Speicherung der Ablaufsprache	89
D.5	Beispiel für die Speicherung von Funktionsblockdiagrammen	91
D.6	Beispiel für die Speicherung von mathematischen Ausdrücken	93
Anhang E (informativ)	Referenzierungsmethoden bei Logikinformationen	97
E.1	Allgemeine Angaben	97
E.2	Referenzierung von als Logikmodelle ausgedrückten Logikinformationen	97
E.2.1	Referenzierung von Logikinformationen in einem Funktionsblock	97
E.2.2	Referenzierung von aus mehreren Funktionsblöcken bestehenden Logikinformationen	98
E.2.3	Referenzierung von aus mehreren AML logic XML Dokumenten bestehenden Logikinformationen	99
E.3	Referenzierung von als Teile von Logikmodellen ausgedrückten Logikinformationen	101
E.3.1	Referenzierung einer Variablen	101
E.3.2	Referenzierung eines Logikelements	103
E.4	Referenzierung von als Teile von Logikmodellen bereits referenzierten Logikinformationen	104
Anhang F (informativ)	Referenzierung von Verriegelungsinformationen	107
F.1	Allgemeine Angaben	107
F.2	Verriegelungsinformationen	108
F.3	Referenzierung von Verriegelungsinformationen ohne Verriegelungsbedingung	109
F.4	Referenzierung von Verriegelungsinformationen mit Verriegelungsbedingung	112
Anhang G (informativ)	XML-Darstellung von AML-Standardbibliotheken	117
G.1	AutomationMLBaseRoleClassLib	117
G.2	AutomationMLInterfaceClassLib	118

Anhang H (normativ) XML-Darstellung des AML logic XML Schemas	120
Literaturhinweise	124
Bilder	
Bild 1 – Überblick über das Planungsdatenaustauschformat AML	11
Bild 2 – AutomationMLBaseRoleClassLib	17
Bild 3 – AutomationMLInterfaceClassLib	20
Bild 4 – Überblick über das AML-Schema	28
Bild 5 – Root-Element „AMLLogic“	29
Bild 6 – Komplexer Typ „FunctionBlock“	30
Bild 7 – Komplexer Typ „ParameterSet“	31
Bild 8 – Komplexer Typ „VariableDecl“	32
Bild 9 – Komplexer Typ „IML“	33
Bild 10 – Komplexer Typ „IMLStep“	34
Bild 11 – Komplexer Typ „IMLTransition“	35
Bild 12 – Komplexer Typ „IMLSimultaneousDivergence“	35
Bild 13 – Komplexer Typ „IMLSimultaneousConvergence“	35
Bild 14 – Komplexer Typ „MathematicalExpression“	36
Bild A.1 – Beispiel für eine Systemdarstellung mit verschiedenen Informationsarten in AML	40
Bild A.2 – Logikmodelle in AML	41
Bild A.3 – Speicherung von Logikmodellen in der AML logic XML	43
Bild A.4 – Modellierungselemente in der AML logic XML	44
Bild B.1 – Modellelemente von Gantt-Diagrammen	46
Bild B.2 – Daten in Gantt-Diagrammen	48
Bild B.3 – Modellelemente von Vorgangsknotennetzen	49
Bild B.4 – Daten in Vorgangsknotennetzen	50
Bild B.5 – Modellelemente von Zeitverlaufdiagrammen	51
Bild B.6 – Informationen in Zeitverlaufdiagrammen	53
Bild D.1 – Durchflussmengen von Ventilen	93
Bild D.2 – Beispiel für die Speicherung eines mathematischen Ausdrucks	96
Bild E.1 – Referenzierung von (als SFC ausgedrückten) in einem Funktionsblock gespeicherten Logikinformationen	97
Bild E.2 – XML-Text der CAEX-Datei zur Referenzierung von in einem Funktionsblock gespeicherten Logikinformationen	98
Bild E.3 – Referenzierung von aus mehreren Funktionsblöcken bestehenden Logikinformationen	99
Bild E.4 – Referenzierung von aus mehreren AML logic XML Dokumenten bestehenden Logikinformationen	100
Bild E.5 – XML-Text der CAEX-Datei zur Referenzierung von aus mehreren AML logic XML Dokumenten bestehenden Logikinformationen	101
Bild E.6 – Referenzierung einer Variablen	102
Bild E.7 – XML-Text der CAEX-Datei zur Referenzierung einer Variablen	103

Bild E.8 – Referenzierung eines Logikelements	103
Bild E.9 – XML-Text der CAEX-Datei zur Referenzierung eines Logikelements	104
Bild E.10 – Referenzierung einer Variablen eines bereits referenzierten Logikmodells	105
Bild E.11 – XML-Text der CAEX-Datei zur Referenzierung einer Variablen eines bereits referenzierten Logikmodells	106
Bild F.1 – Beispiel für ein Fertigungssystem	107
Bild F.2 – Beispiel für eine Verriegelungsquellengruppe und eine Verriegelungszielgruppe	108
Bild F.3 – Referenzierung von Verriegelungsinformationen ohne Verriegelungsbedingung	110
Bild F.4 – XML-Text der CAEX-Datei zur Referenzierung von Verriegelungsinformationen ohne Verriegelungsbedingung	111
Bild F.5 – Referenzierung von Verriegelungsinformationen mit Verriegelungsbedingung	113
Bild F.6 – XML-Text der CAEX-Datei zur Referenzierung von Verriegelungsinformationen mit Verriegelungsbedingung	115
Bild F.7 – Verknüpfung einer logischen Schnittstelle mit einer physikalischen Schnittstelle (Erweiterung von Bild F.5)	115
Bild F.8 – XML-Text der CAEX-Datei zur Verknüpfung einer logischen Schnittstelle mit einer physikalischen Schnittstelle (Erweiterung von Bild F.6)	116

Tabellen

Tabelle 1 – Abkürzungen	16
Tabelle 2 – Rollenklasse InterlockingTargetGroup	18
Tabelle 3 – Rollenklasse InterlockingSourceGroup	18
Tabelle 4 – Rollenklasse LogicModelObject	19
Tabelle 5 – Schnittstellenklasse PLCopenXMLInterface	21
Tabelle 6 – Schnittstellenklasse VariableInterface	21
Tabelle 7 – Schnittstellenklasse AMLLogicXMLInterface	21
Tabelle 8 – Schnittstellenklasse LogicModelInterface	22
Tabelle 10 – Schnittstellenklasse BehaviourLogicModelInterface	22
Tabelle 12 – Schnittstellenklasse LogicModelElementInterface	23
Tabelle 13 – Schnittstellenklasse VariableInterface	23
Tabelle 14 – Schnittstellenklasse InterlockingConnector	24
Tabelle 15 – Schnittstellenklasse InterlockingVariableInterface	24
Tabelle 16 – Metainformationen zu jedem IML-Exportwerkzeug	26
Tabelle C.1 – Zuordnung des Startpunkts eines Gantt-Diagramms zu IML-Elementen	55
Tabelle C.2 – Zuordnung der Balken eines Gantt-Diagramms zu IML-Elementen	55
Tabelle C.3 – Zuordnung der Pfeile eines Gantt-Diagramms zu IML-Elementen	56
Tabelle C.4 – Zuordnung des Balkens eines Gantt-Diagramms mit mindestens einem nachfolgenden Balken zu IML-Elementen	57
Tabelle C.5 – Zuordnung des Balkens eines Gantt-Diagramms mit mindestens einem vorhergehenden Balken zu IML-Elementen	58
Tabelle C.6 – Zuordnung des Startpunkts eines Vorgangsknotennetzes zu IML-Elementen	60
Tabelle C.7 – Zuordnung der Knoten eines Vorgangsknotennetzes zu IML-Elementen	60

Tabelle C.8 – Zuordnung der Pfeile eines Vorgangsknotennetzes zu IML-Elementen	61
Tabelle C.9 – Zuordnung der Knoten für nachfolgende Aktivitäten eines Vorgangsknotennetzes zu IML-Elementen	62
Tabelle C.10 – Zuordnung der Knoten für vorhergehende Aktivitäten eines Vorgangsknotennetzes zu IML-Elementen	63
Tabelle C.11 – Zuordnung der Zeitachse eines Zeitverlaufsdigramms zu IML-Elementen	65
Tabelle C.12 – Zuordnung von Ressourcen und Ressourcenzuständen eines Zeitverlaufsdigramms zu IML-Elementen	65
Tabelle C.13 – Zuordnung der Lebenslinien eines Zeitverlaufsdigramms zu IML-Elementen	66
Tabelle C.14 – Zuordnung von Zeit- und Ressourcensignalen zu IML-Elementen	68
Tabelle C.15 – Speicherung von der Ablaufsprache in AML logic XML	70
Tabelle C.16 – Speicherung der Variablen eines Funktionsblockdiagramms in der AML logic XML	71
Tabelle C.17 – Speicherung der Variablen eines mathematischen Ausdrucks in der AML logic XML	72
Tabelle C.18 – Speicherung der Variablenzuordnungen eines mathematischen Ausdrucks in der AML logic XML	73
Tabelle C.19 – Speicherung eines mathematischen Ausdrucks in der AML logic XML	73
Tabelle D.1 – Speicherung des Beispiels für ein Gantt-Diagramm „Aktivitäten ohne Vorgänger- und Nachfolger-Beziehung“	74
Tabelle D.2 – Speicherung des Beispiels für ein Gantt-Diagramm „Ablauf von Aktivitäten“	75
Tabelle D.3 – Speicherung des Beispiels für ein Gantt-Diagramm „Sequenzierung von Aktivitäten mit Divergenz“	76
Tabelle D.4 – Speicherung des Beispiels für ein Gantt-Diagramm „Ablauf von Aktivitäten mit Konvergenz“	77
Tabelle D.5 – Speicherung des Beispiels für ein Vorgangsknotennetz „Aktivitäten ohne Vorgänger- und Nachfolger-Beziehung“	79
Tabelle D.6 – Speicherung des Beispiels für ein Vorgangsknotennetz „Ablauf von Aktivitäten“	80
Tabelle D.7 – Speicherung des Beispiels für ein Vorgangsknotennetz „Ablauf von Aktivitäten mit Divergenz“	81
Tabelle D.8 – Speicherung des Beispiels für ein Vorgangsknotennetz „Ablauf von Aktivitäten mit Konvergenz“	82
Tabelle D.9 – Beispiel für die Speicherung eines Zeitverlaufsdigramms „Umwandlung eines Zustandswechsels in einen nachfolgenden Zustand“	84
Tabelle D.10 – Beispiel für die Zuordnung eines Zeitverlaufsdigramms „Zwei mit einer Verzögerung von drei Sekunden ausgelöste externe Signale“	85
Tabelle D.11 – Beispiel für die Speicherung eines Zeitverlaufsdigramms „Vom Zustand einer Ressource ausgelöstes Signal, das von einem anderen Zustand verwendet wird“	87
Tabelle D.12 – Beispiel für die Speicherung von der Ablaufsprache	89
Tabelle D.13 – Beispiel für die Speicherung von Funktionsblockdiagrammen	91

Datenaustauschformat für Planungsdaten industrieller Automatisierungssysteme – Automation Markup Language (AML) – Teil 4: Logik

Einleitung

Das in der IEC 62714 definierte Datenaustauschformat AML (Automation Markup Language) ist ein XML-Schema basiertes Datenformat, das entwickelt wurde, um den Datenaustausch zwischen Planungswerkzeugen in einer heterogenen Planungswerkzeuglandschaft zu unterstützen. IEC 62714-1 gibt einen Überblick zu diesem Format.

Das Ziel von AML besteht darin, Planungswerkzeuge der verschiedenen Disziplinen der vorhandenen heterogenen Werkzeuglandschaft, z. B. Maschinen- und Anlagenbau, elektrische Konstruktion, Verfahrenstechnik, Prozessleittechnik, HMI-Entwicklung, SPS-Programmierung, Roboterprogrammierung usw. miteinander zu verbinden.

AML speichert Planungsinformationen entsprechend dem objektorientierten Paradigma und ermöglicht die Modellierung physikalischer und logischer Anlagenkomponenten als Datenobjekte, in denen unterschiedliche Aspekte enthalten sind. Ein Objekt kann aus Unterobjekten bestehen und kann zugleich selbst Teil einer größeren Komposition bzw. Aggregation sein. Typische Objekte in der Anlagenautomatisierung umfassen Informationen zur Topologie, Geometrie, Kinematik und Logik, wobei die Logik ihrerseits Abläufe, Verhalten und Steuerung beinhaltet.

AML kombiniert bestehende Industriedatenformate, die zur Speicherung und zum Austausch von verschiedenen Aspekten der Planungsinformationen entwickelt wurden. Diese Datenformate werden innerhalb ihrer eigenen Spezifikationen „so wie sie sind“ (en: „as is“) verwendet, und es wurden aus ihnen keine speziellen AML-Zweige entwickelt.

Der Kern von AML ist das Top-Level-Datenformat CAEX (Datenformat der höchsten Ebene), welches die verschiedenen Datenformate miteinander verbindet. Zu diesem Zweck verfügt AML über eine inhärent verteilte Dokumentenarchitektur.

Bild 1 gibt einen Überblick über die Grundarchitektur von AML sowie über die Verteilung der Topologie, Geometrie, Kinematik und Logikinformationen.

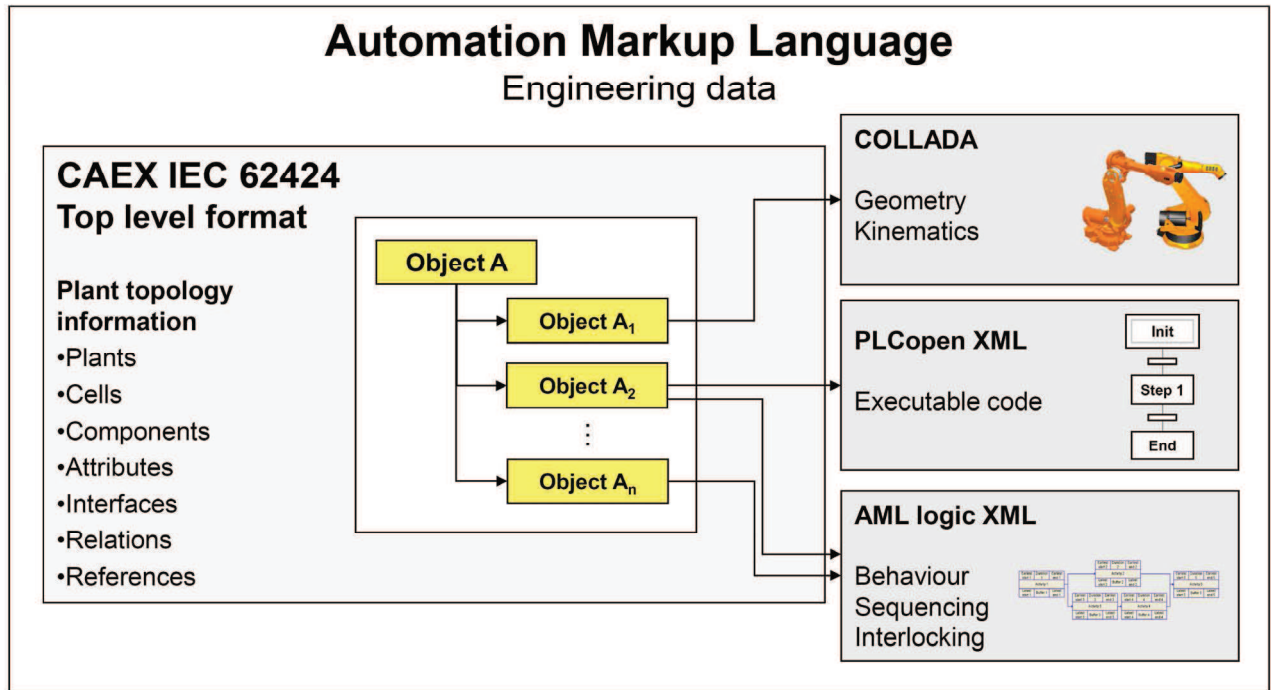


Bild 1 – Überblick über das Planungsdatenaustauschformat AML

Automation Markup Language	Automation Markup Language (AML)
Engineering data	Planungsdaten
Top level format	Toplevel-Format
Plant topology information	Anlagen-Topologiedaten
Plants	Anlagen
Cells	Zellen
Components	Komponenten
Attributes	Attribute
Interfaces	Schnittstellen
Relations	Beziehungen
References	Referenzierungen
Object A	Objekt A
Collada	Collada
Geometric Kinematics	Geometrie kinematik
PLCopen XML	PLCopen XML
Executable codes	Ablauffähiger Code
AML logic XML	XML der AML-Logik
Behaviour	Verhalten
Sequencing	Sequenzierung
Interlocking	Verriegelung

Aufgrund der verschiedenen Aspekte von AML besteht IEC 62714 aus verschiedenen Teilen, die sich auf unterschiedliche Aspekte von AML konzentrieren:

- IEC 62714-1: Architektur und allgemeine Anforderungen
Dieser Teil beschreibt die allgemeine Architektur von AML, die Modellierung von Planungsdaten, Klassen, Instanzen, Beziehungen, Referenzen, Hierarchien, grundlegende AML-Bibliotheken sowie erweiterte AML-Konzepte.
- IEC 62714-2: Rollenklassenbibliotheken
Dieser Teil spezifiziert zusätzliche AML-Bibliotheken.
- IEC 62714-3: Geometrie und Kinematik
Dieser Teil spezifiziert die Modellierung von Geometrie- und Kinematikinformationen.
- IEC 62714-4: Logik
Dieser Teil spezifiziert die Modellierung und Referenzierung von Logikinformationen.

Um die Verknüpfung weiterer Datenstandards mit AML zu ermöglichen, werden möglicherweise zukünftig weitere Teile ergänzt.

Abschnitt 5 beschreibt die logikbezogenen Erweiterungen der Rollen- und Schnittstellen-Klassenbibliothek.

Abschnitt 6 beschreibt, wie Metainformationen zum Quellwerkzeug direkt im Dokument IEC 61131-10 gespeichert werden.

Abschnitt 7 enthält eine normative Festlegung zur Referenzierung von Logikinformationen in AML logic XML Dokumenten.

Abschnitt 8 enthält eine normative Festlegung zum AML logic XML Schema, mit dem Logikmodelle gespeichert werden können.

Abschnitt 9 enthält eine normative Festlegung zur Referenzierung von Verriegelungsinformationen in AML logic XML Dokumenten.

Anhang A enthält einen informativen Überblick über diesen Teil der Norm.

Anhang B enthält eine normative Festlegung der betrachteten Logikmodelle.

Anhang C enthält die normativen Festlegungen zur Speicherung der Logikmodelle in AML logic XML Dokumenten.

Anhang D enthält Beispiele für die Speicherung von Logikmodellen in AML logic XML Dokumenten.

Anhang E beschreibt die Referenzierungsmethoden für Logikinformationen.

Anhang F beschreibt die Referenzierungsmethoden für Verriegelungsinformationen.

Anhang G enthält eine informative XML-Darstellung der in diesem Teil der IEC 62714 definierten Bibliotheken.

Anhang H enthält eine normative XML-Darstellung des in diesem Teil der IEC 62714 definierten AML logic XML Schemas.

1 Anwendungsbereich

Dieser Teil der IEC 62714 legt die Integration von Logikinformationen als Bestandteil eines AML-Modells für den Austausch zwischen Planungswerkzeugen für die Anlagenautomatisierung mittels AML fest. Hierzu werden eine bestimmte Objektsemantik sowie Objektschnittstellensemantik entwickelt. Zur Unterstützung des iterativen Planungsprozesses von Produktionssystemen integriert AML Logikmodelle aus verschiedenen Planungsphasen und -Disziplinen und macht sie untereinander austauschbar und transformierbar, wodurch ein konsistenter und verlustfreier Datenaustausch möglich wird.

Die Logikinformationen ihrerseits lassen sich in drei Arten unterteilen: Ablauf-, Verhaltens- und

sicherheitsinformationen. Dabei beschreiben die Ablaufinformationen die Anweisungen für das geregelte Objekt, die Verhaltensinformationen die Reaktion des geregelten Objekts auf die Ablaufinformationen und auf andere externe Einflussfaktoren. Die Sicherheitsinformationen beschreiben die Anweisungen für das geregelte Objekt, mit denen verhindert werden soll, dass sich aus der Nutzung des Objekts gefährdende Auswirkungen für Menschen und die Umwelt ergeben. Daher können einem Objekt einer Produktionssystemstruktur bis zu drei Arten von Logikinformationen zugewiesen sein.

Logikinformationen werden im Allgemeinen in von den Objekten des AML-Modells referenzierten Logikmodellen organisiert und gespeichert. Damit AML im Rahmen des Planungsprozesses von Produktionssystemen anwendbar ist, unterstützt diese Auszeichnungssprache allgemeine Logikmodelle, welche die verschiedenen Phasen des Planungsprozesses abdecken. Für die folgenden drei Logikmodelle wird eine mit IML (en: Intermediate Modelling Layer) bezeichnete gemeinsame Basis festgelegt und so die Transformation eines Logikmodells in ein anderes ermöglicht. Die vorwärts gerichtete Transformation unterstützt den Informationsanreicherungsprozess und verringert oder vermeidet die erneute Eingabe von Daten zwischen den austauschenden Planungswerkzeugen. Theoretisch kann dieser Workflow bis hin zu einem lauffähigen Programm angewendet werden. Aber auch die rückwärts gerichtete Transformation ergibt Sinn, da der Planungsprozess kein Wasserfallprozess ist. Aufgrund der möglichen Änderungen von Anforderungen und Fehlern bei der Entwicklung enthält er Iterationen und Schleifen. Da die Logikmodelle aber unterschiedliche Modellierungsmächtigkeit aufweisen, ergibt sich aus der rückwärts gerichteten Transformation ein Informationsverlust, wenn sie von den Importern und den Exportern der Software-Werkzeuge nicht ordnungsgemäß verarbeitet werden.

- In den frühen Phasen des Planungsprozesses werden hauptsächlich Gantt-Diagramme verwendet, um die Zeitplanung und Dauer der Herstellungsprozesse auf einer hohen Abstraktionsebene darzustellen.
- Vorgangsknotennetze dienen in den frühen Phasen des Planungsprozesses in erster Linie zur Darstellung der Zeitplanung und der wechselseitigen Beziehungen von Herstellungsprozessen. Dabei können auch Zeitplanungsinformationen gespeichert werden.
- Zeitverlaufsdiagramme werden in späteren Phasen des Planungsprozesses verwendet, um die Geräte mit ihren Zuständen und Zeitbeziehungen zu beschreiben. Dabei werden echte Signale eingeführt.

IML ermöglicht auch die Abbildung anderer Logikmodelle, beispielsweise ereignisgesteuerte Logikmodelle, wie Zustandsdiagramme.

Über diese auf IML beruhenden Logikmodelle hinaus ermöglicht AML auch die Nutzung anderer relevanter Logikmodelle. Aber diese bringen eine eigene formale, maschinenlesbare und verarbeitbare Darstellung mit und brauchen nicht auf IML abgebildet zu werden. Bei diesen handelt es sich um die folgenden:

- Die Ablaufsprache (SFC, en: Sequential Function Charts) zählt zu den Programmiersprachen der IEC 61131-3 und wird im Wesentlichen im späteren Verlauf der Planung von Produktionssystemen verwendet.
- Funktionsblockdiagramme (FBD) zählen ebenfalls zu den Programmiersprachen der IEC 61131-3 und werden im Wesentlichen im späteren Verlauf der Planung von Produktionssystemen verwendet.
- Mathematische Ausdrücke dienen hauptsächlich im späteren Verlauf des Planungsprozesses zur Beschreibung von mathematischen, physikalischen, chemischen oder logischen Beziehungen eines Objekts.

Insbesondere für Sicherheitsinformationen bietet AML ein Konzept zur Speicherung von Ursache-Wirkungstabellen (Logikmodell) mithilfe von FBD.

Die Grundlage für die Speicherung von Logikmodellen ist die IEC 61131-10. Durch die Definition des AML logic XML Schemas passt dieser Teil der Norm die Anwendung von IEC 61131-10 an. Mithilfe dieses Schemas werden die Logikmodelle gespeichert.

In der Mehrdokumentenarchitektur von AML werden die Logikinformationen in eigenen AML logic XML Dokumenten gespeichert, die auf dem Datenformat nach IEC 61131-10 basieren. Aus diesem Grund wird die Modellierung von Logikinformationen in zwei Schritte unterteilt. Im ersten Schritt wird ein die Logikinfor-

tionen enthaltendes Objekt in CAEX, bezeichnet als Toplevel-Format von AML, modelliert. Im zweiten Schritt muss ein AML logic XML Dokument bereitgestellt werden, das die betreffenden Logikinformationen enthält. Daher wird eine Beziehung zwischen dem AML-Objekt und dem AML logic XML Dokument oder einem darin enthaltenen Element hergestellt. Hierzu werden eine Objekt- und eine Objektschnittstellensemantik benötigt.

Dieser Teil der Norm stellt die Semantik bereit, mit der ein Objekt im Toplevel-Format CAEX als ein für ein Logikmodell relevantes Objekt bestimmt wird, damit die referenzierten AML logic XML Dokumente sowie die referenzierten Logikinformationen in diesem Dokument klassifiziert werden können. Außerdem wird eine Semantik bereitgestellt, um in PLCopen XML Dokumenten gespeicherte Steuerungsprogramme zu referenzieren.

2 Normative Verweisungen

Die folgenden Dokumente, die in diesem Dokument teilweise oder als Ganzes zitiert werden, sind für die Anwendung dieses Dokuments erforderlich. Bei datierten Verweisungen gilt nur die in Bezug genommene Ausgabe. Bei undatierten Verweisungen gilt die letzte Ausgabe des in Bezug genommenen Dokuments (einschließlich aller Änderungen).

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

Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation 04 February 2004 (available at <<http://www.w3.org/TR/2004/REC-xml-20040204/>>) [viewed on 2016-07-28]

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

Mathematical Markup Language (MathML) Version 2.0 (Second Edition), W3C Recommendation 21 October 2003 (available at <<https://www.w3.org/TR/MathML2/>>)

IEC 61131-10 (In Bearbeitung in SC 65B), *PROGRAMMABLE CONTROLLERS – Part 10: PLC open XML exchange format*

RFC5646, *Tags for Identifying Languages*, IETF (available at <<https://tools.ietf.org/html/rfc5646>>)

RFC4122, *A Universally Unique IDentifier (UUID) URN Namespace* (available at <<https://tools.ietf.org/html/rfc4122>>)

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

3 Begriffe und Abkürzungen

3.1 Begriffe

Für die Anwendung dieses Dokuments gelten die Begriffe nach IEC 62714-1 und die folgenden Begriffe.

3.1.1

AML-Logikobjekt

AML-Objekt, das Logikinformationen enthält

3.1.2

AML logic XML Dokument

externes XML-Dokument auf der Grundlage des AML logic XML Schemas, das von einem AML-Logikobjekt referenziert werden kann

3.1.3

AML logic XML Schema

XML-Schema zur Speicherung der Logikmodelle von AML

3.1.4

Verhaltensinformationen

Informationen, mit denen die Reaktion eines geregelten Systems auf Ablaufinformationen und andere externe Einflussfaktoren beschrieben werden

3.1.5

Verhaltenslogikmodell

Darstellung von Verhaltensinformationen

3.1.6

Verriegelungsinformationen

Informationen zur Beschreibung von Anweisungen für ein geregeltes Objekt, mit denen verhindert werden soll, dass sich aus der Nutzung des Objekts gefährdende Auswirkungen für Menschen und die Umwelt ergeben

3.1.7

Verriegelungslogikmodell

Darstellung von Verriegelungsinformationen

3.1.8

Verriegelungsquellengruppe

Gruppe von AML-Objekten, die anzeigen, wenn Aktionen erforderlich sind, um die Funktionssicherheit herzustellen

3.1.9

Verriegelungszielgruppe

Gruppe von AML-Objekten, die Aktionen ausführen, um die Funktionssicherheit herzustellen

3.1.10

Logikmodellelement

Element eines Logikmodells

3.1.11

Logikinformationen

Ablaufinformationen, Verhaltensinformationen oder Verriegelungsinformationen

3.1.12

Logikmodell

Darstellung von Logikinformationen

3.1.13

mathematischer Ausdruck

sinnvolle Kombination aus Zahlen, Variablen und mathematischen Symbolen zur Darstellung syntaktisch korrekter gegenseitiger Abhängigkeiten

Anmerkung zum Begriff 1: Ein mathematischer Ausdruck kann ebenso ein einfacher Ausdruck wie ein komplexes Gleichungssystem sein.

3.1.14

Ablaufinformationen

Informationen, mit denen die Anweisungen für ein geregeltes System beschrieben werden

3.1.5

Ablauflogikmodell

Darstellung von Ablaufinformationen

3.2 Abkürzungen

Für die Anwendung dieses Dokuments gelten die Abkürzungen der IEC 62714-1 und die Abkürzungen in Tabelle 1.

Tabelle 1 – Abkürzungen

SFC	Ablaufsprache (en: Sequential Function Chart)
FBD	Funktionsblockdiagramm
IML	Intermediate Modelling Layer
XSLT	Extensible Stylesheet Language Transformations

4 Konformität

Um Konformität mit dem vorliegenden Dokument in Bezug auf die Unterstützung von AML zu beanspruchen, müssen die Anforderungen von Abschnitt 5, 6, 7, 8 und 9 erfüllt sein. Im Anwendungsbereich von AML muss ein AML logic XML Dokument der Spezifikation von PCOpen® XML 2.0 oder 2.0.1 oder der Spezifikation nach IEC 61131-10 (in Vorbereitung durch SC 65B) entsprechen.

Diese Norm legt keine Einzelheiten zum Datenaustauschverfahren oder Umsetzungsanforderungen für Import-/Export-Werkzeuge fest.

5 Teil 4 bezogene Erweiterungen der AML-Bibliotheken

5.1 Allgemeine Angaben

In diesem Abschnitt werden die Erweiterungen der AML-Standardrollenklassen und der AML-Standardschnittstellenklassen festgelegt. Diese Klassen sind Bestandteil der AML-Standardbibliothek und eine spezifische Erweiterung für diesen Teil von IEC 62714. Alle beschriebenen Attribute und Schnittstellen sind Bestandteil der Standardbibliothek und können in der Instanzhierarchie entfernt werden, falls sie nicht benötigt werden.

5.2 AutomationMLBaseRoleClassLib

5.2.1 Allgemeine Angaben

ANMERKUNG Die AML Basisrollenklassenbibliothek hat die Version 2.2.2 und basiert auf der Version 2.2.0 von IEC 62714-1.

Bild 2 stellt die normative AutomationMLBaseRoleClassLib als Objektbaum dar. Einzelheiten zu jeder Rollenklasse sind in 5.2.2 und 5.2.4 enthalten. Der XML-Text der AutomationMLBaseRoleClassLib wird in G.1 angegeben.



Bild 2 – AutomationMLBaseRoleClassLib

5.2.2 Rollenklasse InterlockingTargetGroup

Die Rollenklasse „InterlockingTargetGroup“ muss nach den Angaben in Tabelle 2 verwendet werden.

Tabelle 2 Rollenklasse InterlockingTargetGroup

Klassenname	InterlockingTargetGroup	
Beschreibung	Die Rollenklasse „InterlockingTargetGroup“ muss für Verriegelungszielgruppen verwendet werden. Einzelheiten werden in Abschnitt 9 angegeben. Beispiele können Anhang F entnommen werden.	
Übergeordnete Klasse	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group	
Pfad für Elementreferenzierung	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group/InterlockingTargetGroup	
Attribute	keine	
Schnittstellen	Schnittstelle des Typs „InterlockingConnector“	Die Schnittstellen müssen dem in 5.3.11 angegebenen Typ „InterlockingConnector“ entsprechen. Diese Schnittstellen müssen für das im Abschnitt 9 angegebenen Verriegelungskonzept erforderlich sein. Die Verwendung der Schnittstelle muss verbindlich sein.
	Schnittstelle des Typs „InterlockingLogicModelInterface“	Die Schnittstellen müssen dem in 5.3.8 angegebenen Typ „InterlockingLogicModelInterface“ entsprechen. Diese Schnittstellen müssen bei dem im Abschnitt 9 angegebenen Verriegelungskonzept zur Referenzierung des Verriegelungslogikmodells verwendet werden. Die Verwendung der Schnittstelle muss optional sein.

5.2.3 Rollenklasse InterlockingSourceGroup

Die Rollenklasse „InterlockingSourceGroup“ muss nach den Angaben in Tabelle 3 verwendet werden.

Tabelle 3 – Rollenklasse InterlockingSourceGroup (1 von 2)

Klassenname	InterlockingSourceGroup	
Beschreibung	Die Rollenklasse „InterlockingSourceGroup“ muss für Verriegelungsquellengruppen verwendet werden. Einzelheiten werden in Abschnitt 9 angegeben. Beispiele können Anhang F entnommen werden.	
Übergeordnete Klasse	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group	
Pfad für Elementreferenzierung	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group/InterlockingSourceGroup	
Attribute	keine	
Schnittstellen	Schnittstelle des Typs „InterlockingConnector“	Die Schnittstellen müssen dem in 5.3.11 angegebenen Typ „InterlockingConnector“ entsprechen. Diese Schnittstellen müssen für das im Abschnitt 9 angegebenen Verriegelungskonzept erforderlich sein. Die Verwendung der Schnittstelle muss verbindlich sein.

Tabelle 3 (2 von 2)

	Schnittstelle des Typs „InterlockingLogicModelInterface“	Die Schnittstellen müssen dem in 5.3.8 angegebenen Typ „InterlockingLogicModelInterface“ entsprechen. Diese Schnittstellen müssen bei dem im Abschnitt 9 angegebenen Verriegelungskonzept zur Referenzierung des Verriegelungslogikmodells verwendet werden. Die Verwendung der Schnittstelle muss optional sein.
	Schnittstelle des Typs „InterlockingVariableInterface“	Die Schnittstellen müssen dem in 5.3.12 angegebenen Typ „InterlockingVariableInterface“ entsprechen. Diese Schnittstelle muss bei dem im Abschnitt 9 angegebenen Verriegelungskonzept zur Referenzierung der Ausgabevariable in einem Logikmodell verwendet werden, das Verriegelungsinformationen enthält. Die Verwendung der Schnittstelle muss optional sein.

5.2.4 Rollenklasse LogicModelObject

Die Rollenklasse „LogicModelObject“ muss entsprechend der Festlegung in Tabelle 4 verwendet werden.

Tabelle 4 – Rollenklasse LogicModelObject

Klassenname	LogicModelObject	
Beschreibung	Die Rollenklasse „LogicModelObject“ muss ein AML-Logikobjekt kennzeichnen, das ein Logikmodell instanziiert.	
Übergeordnete Klasse	AutomationMLBaseRoleClassLib/AutomationMLBaseRole	
Pfad für Elementreferenzierung	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/LogicModelObject	
Attribute	keine	
Schnittstellen	Schnittstelle des Typs „SequencingLogicModelInterface“	Die Schnittstellen müssen dem in 5.3.6 angegebenen Typ „SequencingLogicModelInterface“ entsprechen. Diese Schnittstellen müssen verwendet werden, um die im Abschnitt 7 bezeichnete Logikmodellsemantik anzugeben. Die Verwendung der Schnittstelle muss optional sein.
	Schnittstelle des Typs „BehaviourLogicModelInterface“	Die Schnittstellen müssen dem in 5.3.7 angegebenen Typ „BehaviourLogicModelInterface“ entsprechen. Diese Schnittstellen müssen verwendet werden, um die im Abschnitt 7 bezeichnete Logikmodellsemantik anzugeben. Die Verwendung der Schnittstelle muss optional sein.

ANMERKUNG Diese Objekte können in CAEX weiter detailliert und referenziert werden.

5.3 AutomationMLInterfaceClassLib

5.3.1 Allgemeine Angaben

ANMERKUNG Die Klassenbibliothek dieser AML-Schnittstelle hat die Version 2.2.2 und basiert auf der Version 2.2.0 der IEC 62714-1.

Bild 3 stellt die normative AutomationMLInterfaceClassLib als Objektbaum dar. Einzelheiten zu jeder Schnittstellenklasse sind in 5.3.2 und 5.3.12 enthalten. Der XML-Text der AutomationMLInterfaceClassLib wird in G.2 angegeben.

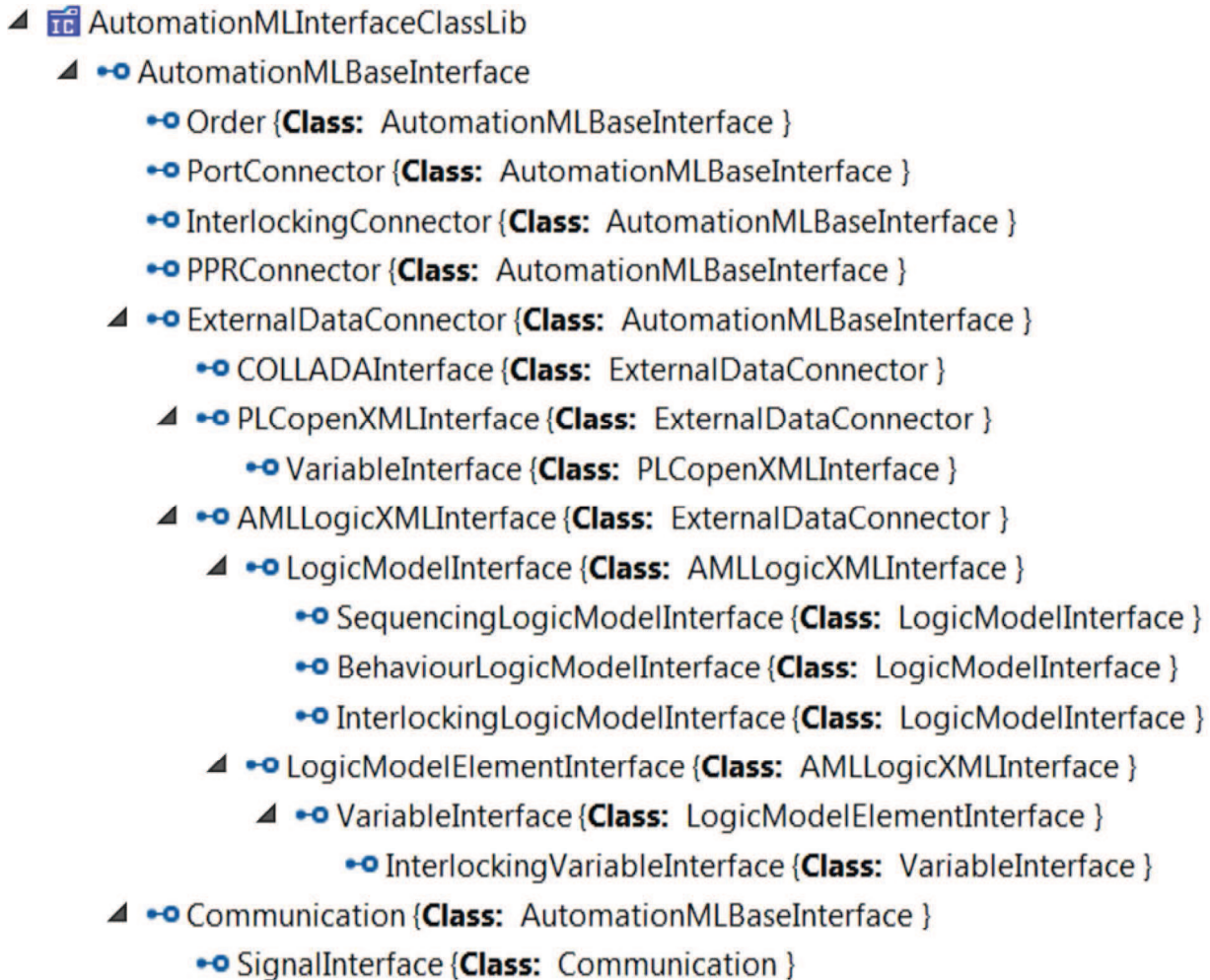


Bild 3 – AutomationMLInterfaceClassLib

5.3.2 InterfaceClass PLCopenXMLInterface

Die Schnittstellenklasse „PLCopenXMLInterface“ muss nach den Angaben in Tabelle 5 verwendet werden.

Tabelle 5 – Schnittstellenklasse PLCopenXMLInterface

Klassenname	PLCopenXMLInterface
Beschreibung	Die Schnittstellenklasse „PLCopenXMLInterface“ muss verwendet werden, um externe PLCopen XML-Dokumente der PLCopenXML-Version 2.0 oder 2.0.1 zu referenzieren.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/ PLCopenXMLInterface
Attribute	keine

5.3.3 Schnittstellenklasse VariableInterface

Die Schnittstellenklasse „VariableInterface“ muss nach den Angaben in Tabelle 6 verwendet werden.

Tabelle 6 – Schnittstellenklasse VariableInterface

Klassenname	VariableInterface
Beschreibung	Die Schnittstellenklasse „VariableInterface“ muss verwendet werden, um Signale oder Variablen zu veröffentlichen, die in PLCopen XML-Dokumenten der PLCopenXML-Version 2.0 oder 2.0.1 festgelegt sind.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/ PLCopenXMLInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/ PLCopenXMLInterface/VariableInterface
Attribute	keine

5.3.4 Schnittstellenklasse AMLLogicXMLInterface

Die Schnittstellenklasse „AMLLogicXMLInterface“ muss nach den Angaben in Tabelle 7 verwendet werden.

Tabelle 7 – Schnittstellenklasse AMLLogicXMLInterface

Klassenname	AMLLogicXMLInterface
Beschreibung	Alle Ableitungen der Schnittstellenklasse „AMLLogicXMLInterface“ müssen verwendet werden, um ein Logikmodell oder Logikmodellelemente eines externen AML logic XML Dokuments zu referenzieren.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/ AMLLogicXMLInterface
Attribute	keine

5.3.5 Schnittstellenklasse LogicModelInterface

Die Schnittstellenklasse „LogicModelInterface“ muss nach den Angaben in Tabelle 8 verwendet werden.

Tabelle 8 – Schnittstellenklasse LogicModelInterface

Klassenname	LogicModelInterface
Beschreibung	Alle Ableitungen der Schnittstellenklasse „LogicModelInterface“ müssen verwendet werden, um ein Logikmodell zu referenzieren, das in einem Funktionsblock eines externen AML logic XML Dokuments gespeichert ist. Einzelheiten werden in Abschnitt 7 angegeben. Beispiele können Anhang E entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMMLLogicXMLInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMMLLogicXMLInterface/LogicModelInterface
Attribute	keine

5.3.6 Schnittstellenklasse SequencingLogicModelInterface

Die Schnittstellenklasse „SequencingLogicModelInterface“ muss nach den Angaben in Tabelle 9 verwendet werden.

Tabelle 9 – Schnittstellenklasse SequencingLogicModelInterface

Klassenname	SequencingLogicModelInterface
Beschreibung	Die Schnittstellenklasse „SequencingLogicModelInterface“ muss verwendet werden, um ein Ablaufinformationen enthaltendes Logikmodell zu referenzieren, das in einem Funktionsblock eines externen AML logic XML Dokuments gespeichert ist. Einzelheiten werden in Abschnitt 7 angegeben. Beispiele können Anhang E entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMMLLogicXMLInterface/LogicModelInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMMLLogicXMLInterface/LogicModelInterface/SequencingLogicModelInterface
Attribute	keine

5.3.7 Schnittstellenklasse BehaviourLogicModelInterface

Die Schnittstellenklasse „BehaviourLogicModelInterface“ muss nach den Angaben in Tabelle 10 verwendet werden.

Tabelle 10 – Schnittstellenklasse BehaviourLogicModelInterface

Klassenname	BehaviourLogicModelInterface
Beschreibung	Die Schnittstellenklasse „BehaviourLogicModelInterface“ muss verwendet werden, um ein Verhaltensinformationen enthaltendes Logikmodell zu referenzieren, das in einem Funktionsblock eines externen AML logic XML Dokuments gespeichert ist. Einzelheiten werden in Abschnitt 7 angegeben. Beispiele können Anhang E entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMMLLogicXMLInterface/LogicModelInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMMLLogicXMLInterface/LogicModelInterface/BehaviourLogicModelInterface
Attribute	keine

5.3.8 Schnittstellenklasse InterlockingLogicModelInterface

Die Schnittstellenklasse „InterlockingLogicModelInterface“ muss nach den Angaben in Tabelle 11 verwendet werden.

Tabelle 11 – Schnittstellenklasse InterlockingLogicModelInterface

Klassenname	InterlockingLogicModelInterface
Beschreibung	Die Schnittstellenklasse „InterlockingLogicModelInterface“ muss verwendet werden, um ein Verriegelungsinformationen enthaltendes Logikmodell zu referenzieren, das in einem Funktionsblock eines externen AML logic XML Dokuments gespeichert ist. Diese Schnittstellenklasse darf nur von Verriegelungsquellen und Verriegelungszielgruppen verwendet werden. Einzelheiten werden in Abschnitt 9 angegeben. Beispiele können Anhang F entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface/LogicModelInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface/LogicModelInterface/InterlockingLogicModelInterface
Attribute	keine

5.3.9 Schnittstellenklasse LogicModelElementInterface

Die Schnittstellenklasse „LogicModelElementInterface“ muss nach den Angaben in Tabelle 12 verwendet werden.

Tabelle 12 – Schnittstellenklasse LogicModelElementInterface

Klassenname	LogicModelElementInterface
Beschreibung	Die Schnittstellenklasse „LogicModelElementInterface“ muss verwendet werden, um ein Element in einem Logikmodell zu referenzieren, das in einem Funktionsblock eines externen AML logic XML Dokuments gespeichert ist. Einzelheiten werden in Abschnitt 7 angegeben. Beispiele können Anhang E entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface/LogicModelElementInterface
Attribute	keine

5.3.10 Schnittstellenklasse VariableInterface

Die Schnittstellenklasse „VariableInterface“ muss nach den Angaben in Tabelle 13 verwendet werden.

Tabelle 13 – Schnittstellenklasse VariableInterface (1 von 2)

Klassenname	VariableInterface
Beschreibung	Die Schnittstellenklasse „VariableInterface“ muss verwendet werden, um eine Variable in einem Logikmodell zu referenzieren, das in einem Funktionsblock eines externen AML logic XML Dokuments gespeichert ist. Einzelheiten werden in Abschnitt 7 angegeben. Beispiele können Anhang E entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface/LogicModelElementInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface/LogicModelElementInterface/VariableInterface

Tabelle 13 (2 von 2)

Attribute	Direction (type="xs:string")	<p>Dieses Attribut muss verwendet werden, um eine Richtung des VariableInterface zu bezeichnen. Erlaubte Werte: „In“ (Eingabe), „Out“ (Ausgabe) oder „InOut“ (Ein-/Ausgabe).</p> <p>Der Attributwert muss „In“ (Eingabe) lauten, wenn die Variable von CAEX verwendet wird. Der Attributwert muss „Out“ (Ausgabe) lauten, wenn der Informationsfluss von CAEX zur Variable läuft.</p> <p>Der Attributwert muss „InOut“ (Ein-/Ausgabe) lauten, wenn beide Richtungen möglich sind.</p> <p>Die Verwendung des Attributs muss optional sein.</p>
------------------	------------------------------	---

5.3.11 Schnittstellenklasse InterlockingConnector

Die Schnittstellenklasse „InterlockingConnector“ muss nach den Angaben in Tabelle 14 verwendet werden.

Tabelle 14 – Schnittstellenklasse InterlockingConnector

Klassenname	InterlockingConnector
Beschreibung	Die Schnittstellenklasse „InterlockingConnector“ muss verwendet werden, um Relationen zwischen einer Verriegelungsquellengruppe und einer Verriegelungszielgruppe zu modellieren. Einzelheiten werden in Abschnitt 9 angegeben. Beispiele können Anhang F entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/InterlockingConnector
Attribute	keine

5.3.12 Schnittstellenklasse InterlockingVariableInterface

Die Schnittstellenklasse „InterlockingVariableInterface“ muss nach den Angaben in Tabelle 15 verwendet werden.

Tabelle 15 – Schnittstellenklasse InterlockingVariableInterface (1 von 2)

Klassenname	InterlockingVariableInterface
Beschreibung	Die Schnittstellenklasse „InterlockingVariableInterface“ muss verwendet werden, um die Ausgangsvariable in einem Logikmodell zu referenzieren, das Verriegelungsinformationen enthält, die in einem Funktionsblock eines externen AML logic XML Dokuments gespeichert sind. Diese Variable stellt das Ergebnis einer Funktion dar, mit der die Verriegelungsbedingung beschrieben wird. Diese Schnittstellenklasse darf nur von Verriegelungsquellengruppen und Verriegelungszielgruppen verwendet werden. Einzelheiten werden in Abschnitt 9 angegeben. Beispiele können Anhang F entnommen werden.
Übergeordnete Klasse	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface/LogicModelElementInterface/VariableInterface
Pfad für Elementreferenzierung	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/AMLLogicXMLInterface/LogicModelElementInterface/VariableInterface/InterlockingVariableInterface

Tabelle 15 (2 von 2)

Attribute	SafeConditionEquals (type="xs:boolean")	<p>Das Attribut „SafeConditionEquals“ muss verwendet werden, um das Ergebnis einer Funktion anzugeben, mit der die Verriegelungsbedingung definiert wird. Die erlaubten Werte müssen „true“ (wahr) oder „false“ (falsch) sein und angeben, welcher Wert einer Booleschen Variable den sicheren Zustand einer Verriegelungsquellengruppe angibt.</p> <p>Die Verwendung des Attributs muss optional sein. Der Standardwert muss „true“ (wahr) lauten.</p>
-----------	--	---

6 Metainformationen zu IML-Exportwerkzeugen

Um den Datenaustausch zwischen einem Quellwerkzeug und einem Zielwerkzeug zu vereinfachen, ist es hilfreich, die Informationen zu den Exportwerkzeugen unmittelbar im IML-Dokument zu speichern. Daher gelten die folgenden Festlegungen:

ANMERKUNG Im Planungsprozess ist es hilfreich, die Informationen zum Projekt, zu dem die Daten gehören, und den Zeitpunkt des Datenexports zu übertragen. Damit wird der Implementierungsprozess der Schnittstellen unterstützt.

- Ein IML-Dokument muss die Informationen zu jedem Exportwerkzeug bereitstellen, mit dem das IML-Dokument geschrieben wurde.
- In einer Datenaustausch-Werkzeugkette müssen die letzten beteiligten Exportwerkzeuge diese Informationen im IML-Dokument speichern.
- Jedes IML-Dokument muss mindestens ein XML-Element „WriterHeader“ (Erstellerinformationen) enthalten.
- In einer Datenaustausch-Werkzeugkette müssen alle beteiligten Exportwerkzeuge diese Informationen in gleicher Art und Weise im IML-Dokument speichern. Daher kann das Dokument Informationen über mehrere Werkzeuge einer Werkzeugkette für den Datenaustausch enthalten. Ein Werkzeug darf die Erstellerinformationen anderer Werkzeuge entfernen. Dies verhindert möglicherweise den iterativen Datenaustausch mit anderen Werkzeugen. Daher wird das Entfernen der Erstellerinformationen anderer Werkzeuge nicht empfohlen.
- Diese Informationen müssen in XML-Elementen des XML-Elements „WriterHeader“ gespeichert werden.
- Die Metainformationen müssen folgende Angaben bereitstellen:
 - Bezeichnung des Exportwerkzeugs;
 - eindeutige ID des Exportwerkzeugs (z. B. UUID bzw. URL des Werkzeugs);
 - Name des Anbieters des Exportwerkzeugs;
 - URL des Anbieters des Exportwerkzeugs;
 - vollständige Versionsinformationen des Exportwerkzeugs (einschließlich der Hauptversions-, Nebenversionsnummer sowie der Buildnummer); und
 - Datum und Uhrzeit des letzten Schreibvorgangs in das IML-Dokument.
- Zusätzlich können die folgenden optionalen Informationen enthalten sein:
 - Regions- und Spracheinstellungen des Exportwerkzeugs nach RFC 5646;
 - Bezeichnung des Projekts, mit dem die exportierten Daten im Zusammenhang stehen;
 - Eindeutige ID des Projekts, mit dem die exportierten Daten im Zusammenhang stehen; sowie
 - weitere Angaben.
- Die Informationen oben müssen mithilfe der XML-Elemente gespeichert werden, die in Tabelle 16 für jedes Exportwerkzeug abgebildet sind.

Tabelle 16 – Metainformationen zu jedem IML-Exportwerkzeug

XML-Elementname	Typ	Kardinalität	Beispiel
WriterName	xs:string	1	„Werkzeug XYZ“
WriterID	xs:string	1	„4F1D5716-1C2A-4CBC-8C40-91A91F7E09A4“
WriterVendor	xs:string	1	„Anbieter XYZ“
WriterVendorURL	xs:string	1	„http://www.anbieter_xyz.org“
WriterVersion	xs:string	1	„0.2.123 PreAlpha“
LastWritingDateTime	xs:dateTime	1	„2011-05-25T09:30:47“
WriterCulturalSetting	xs:string	0..1	„de-DE“
WriterProjectTitle	xs:string	0..1	„Projekt_Y“
WriterProjectID	xs:string	0..1	„6BA300EA-DF77-4E78-80F9-2B050E2C6799“
AdditionalInformation	xs:string	0..1	„Der Anbieter dieses Werkzeugs hieß zuvor ‚Anbieter ABC‘.“

7 Referenzierung von AML logic XML Dokumenten

7.1 Allgemeine Angaben

Nach der verteilten Dokumentstruktur von AML ist es erforderlich, die unterschiedlichen Informationen der verschiedenen Dokumente zueinander in Beziehung zu setzen. Dieser Abschnitt befasst sich mit der Referenzierung von AML-Objekten (innerhalb eines CAEX-Dokuments) mit den in den AML logic XML Dokumenten gespeicherten Logikinformationen. Ein informativer Überblick über die Referenzierungsmethoden ist in Anhang E enthalten.

7.2 Referenzierung von Logikinformationen

Im Hinblick auf die Referenzierung von Logikinformationen gelten die folgenden Festlegungen:

- Die Referenzierung von Logikinformationen in einem AML logic XML Dokument durch ein AML-Objekt muss mithilfe eines CAEX-„ExternalInterface“-Elements über die in 5.3 angegebenen AML-Schnittstellenklassen „LogicModelInterface“ und „LogicModelElementInterface“ sowie deren Ableitungen modelliert werden.
 - Logikinformationen müssen anhand ihrer URI innerhalb des Attributs „refURI“ dieser CAEX-„ExternalInterface“-Elemente referenziert werden.
 - Der Wert des Attributs „refURI“ muss auf die Logikinformationen im AML logic XML Dokument verweisen und der URI-Syntax entsprechen. Dabei wird das AML logic XML-Attribut „uuid“ als Fragment verwendet.

ANMERKUNG 1 Ein Beispiel für den Wert von refURI ist „file:///behaviour.xml#74e94dc8-b5a2-490c-bf0f-f0fdaa4515ef“.

ANMERKUNG 2 In diesem Teil der IEC 62714 werden UUIDs in einer Kurzform, wie „UUID1“, „UUID100“ usw. dargestellt. Dies dient der besseren Lesbarkeit und soll eine echte UUID abbilden.

ANMERKUNG 3 Es wird nicht empfohlen, nur das AML logic XML Dokument selbst zu referenzieren, da das Dokument verschiedene Logikmodelle mit unterschiedlichen Logikinformationen enthalten kann.

- Jedes „LogicModelElementInterface“ und seine Ableitungen müssen mithilfe eines „InternalLink“-Elements auf eine entsprechende Ableitung von „LogicModelInterface“ verweisen.

ANMERKUNG 1 Eine Ableitung von „LogicModelInterface“ kann mithilfe eines „InternalLink“-Elements mit mehr als einem „LogicModelElementInterface“-Element und seiner Ableitungen verbunden werden.

ANMERKUNG 2 Die Variablen können dann mithilfe von „InternalLink“-Elementen beispielsweise den Signalen anderer AML-Objekte oder den Variablen anderer Logikmodelle zugewiesen werden.

- Logikmodelle müssen eindeutig unterschieden werden können, d. h. ein AML-Logikobjekt darf höchstens ein CAEX-„ExternalInterface“-Element der gleichen AML-Schnittstellenklasse „SequencingLogicModelInterface“, „BehaviourLogicModelInterface“ oder einer Ableitung derselben aufweisen. Jede Verriegelungsquellengruppe oder Verriegelungszielgruppe darf nur ein CAEX-„ExternalInterface“-Element der AML-Schnittstellenklasse „InterlockingLogicModelInterface“ aufweisen.

ANMERKUNG So sind beispielsweise bei einem AML-Logikobjekt keine unterschiedlichen Detailstufen eines Logikmodells zulässig.

8 Beschreibung des AML logic XML Schemas

8.1 Überblick über das Schema

8.1.1 Verwendung des Schemas aus IEC 61131-10

In diesem Abschnitt wird das „AML logic XML Schema“ beschrieben, das formal in Anhang H festgelegt wird.

Das AML logic XML Schema importiert das Schema nach IEC 61131-10 und verwendet teilweise dessen Typdefinitionen erneut, um Variablen, Funktionsblöcke, Abläufe usw. verwendende Logikmodelle darzustellen.

Das Schema nach IEC 61131-10 unterstützt zwei Erweiterungsmechanismen, mit denen die Einbindung von Informationen in einem Austauschformat ermöglicht wird. Der erste Mechanismus ist das Element „AddData“, das unter verschiedene andere Elemente verschachtelt werden kann. Seine Nutzung ist für die Erweiterung anderer Elemente vorgesehen, um diese mit Informationen anzureichern, ohne dass er allerdings für den Austausch des Elements zwischen verschiedenen Werkzeugen erforderlich wäre. Der zweite Mechanismus besteht in der Nutzung eines XML-Schemaverfahrens mit der Bezeichnung „abstrakte Typsubstitution“. Aus dem abstrakten Typ kann ein neues Element abgeleitet und dort als Substitution für den Elementtyp verwendet werden, wo der abstrakte Typ angegeben wird.

Das AML logic XML Schema folgt beiden Erweiterungsmechanismen von IEC 61131-10 und den Namenskonventionen für Elemente. Der Überblick über die Schemastruktur ist in Bild 4 im UML-Klassendiagramm dargestellt. Klassen, deren Bezeichnungen mit „IEC61131_1-Ed1_0“ beginnen, stellen aus dem Schema nach IEC 61131-10 importierte Typen dar. Darüber hinaus stellen Klassen, deren Bezeichnungen kursiv (in den gelb unterlegten Feldern) gedruckt sind, abstrakte Typen dar, die durch einen ihrer erweiterten nicht abstrakten Typen ersetzt werden müssen.

- Wie aus Bild 4 hervorgeht, werden die anwenderdefinierten Datentypdeklarationen und ihre Referenzierungen aus der Variablendeklaration vom AML logic XML Schema vollständig wiederverwendet.
- Die Logikbeschreibungen und ihre Basis-Typen aus IEC 61131-10 werden ebenfalls vom AML logic XML Schema wiederverwendet. Siehe hierzu Bild 4. Damit stehen alle Logikbeschreibungen aus IEC 61131-10 im AML logic XML zur Verfügung. Außerdem fügt das AML logic XML Schema durch die „Substitution abstrakter Typen“, also durch den zweiten aus IEC 61131-10 folgenden Erweiterungsmechanismus, spezifische Logikmodelle wie IML und mathematische Ausdrücke hinzu.
- Das AML logic XML Schema erweitert nach der Darstellung in Bild 4 die grundlegenden SCF-Elemente von IEC 61131-10, um den in A.3 bezeichnete Intermediate Modelling Layer (IML) abzubilden.



Logic descriptions of IEC 61131-10 and its extension base are fully imported and reused	Logikbeschreibungen der IEC 61131-10 und ihre extension base sind vollständig importiert und weiterverwendet
User-defined data type declaration and reference mechanism of IEC 61131-10 are fully imported and reused.	Benutzerdefinierte Datentypdeklarationen und Referenzierungsmechanismen der IEC 61131-10 sind vollständig importiert und weiterverwendet
“IML” partially reuses and extends SFC elements of IEC 61131-10	“IML” verwendet teilweise und erweitert die SFC Elemente der IEC 61131-10

8.1.2 Schemaversionierung

Die AML logic XML Schemadefinition weist ein Versionsattribut auf. Die erste Zahl bezeichnet dabei die Ausgabennummer dieser IEC 62714-4. Die zweite Zahl ist für kleinere Änderungen innerhalb einer Ausgabe reserviert. Der Dateiname des Schemas weist ebenfalls die Ausgabennummer aus.

Zusätzlich hat auch das Root-Element „AMLLogic“ ein Attribut „schemaVersion“. Bei der Erstellung eines XML-Dokuments auf der Grundlage dieses Schemas muss seine Versionsnummer auf den Wert dieses Attributs gesetzt werden.

8.2 Root-Element „AMLLogic“

8.2.1 Allgemeine Angaben

Das Root-Element „AMLLogic“ stellt eine Art Projekt oder Bibliothek dar, in der/dem Logikinformationen gespeichert sind. Sein Inhalt ist in Bild 5 abgebildet.

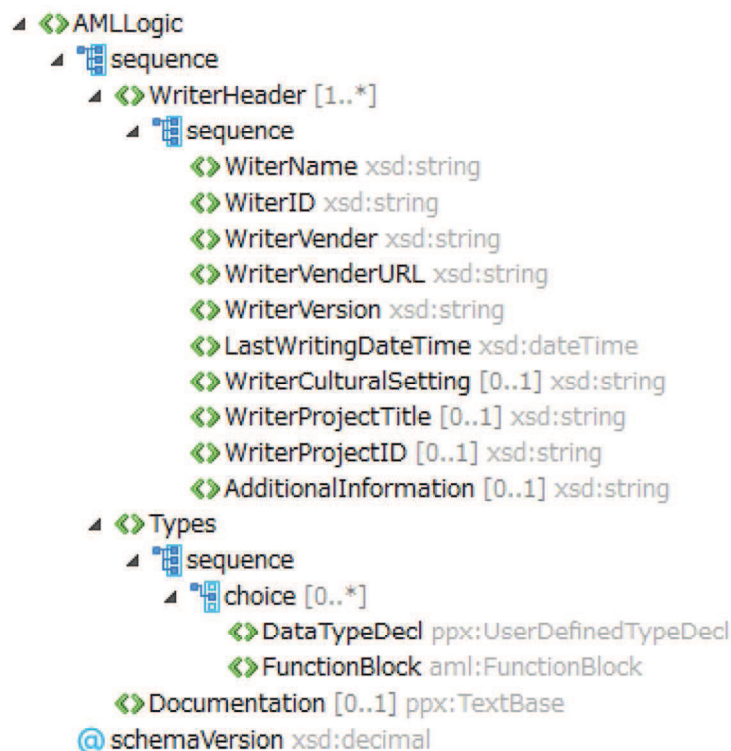


Bild 5 – Root-Element „AMLLogic“

8.2.1.1 Attribute

Das Attribut „schemaVersion“ sollte die Versionsnummer des AML logic XML Schema enthalten, mit der auch die Versionsnummer des XML-Dokuments übereinstimmen sollte.

8.2.1.2 Untergeordnetes Element „WriterHeader“

Wie im Abschnitt 6 erläutert, enthält das Element „WriterHeader“ Metadaten zu dem Werkzeug, mit dem das XML-Dokument erstellt wird.

8.2.1.3 Untergeordnetes Element „Types“

Das Element „Types“ ist ein Platzhalter für die Deklaration des anwenderdefinierten Datentyps und/oder des Funktionsblocks.

Das Element „DataTypeDecl“ des Typs „ppx:UserDefinedTypeDecl“ stellt die Deklaration des anwenderdefinierten Datentyps in der Sprachspezifikation nach IEC 61131-3 dar. Damit kann auch jede Art von Datentyp, die in der Sprachspezifikation von IEC 61131-3 enthalten ist, mithilfe des AML logic XML festgelegt werden. Die detaillierte Spezifikationen des Typs „ppx:UserDefinedTypeDecl“ sollte im Dokument IEC 61131-10 nachgesehen werden.

Das Element „FunctionBlock“ vom Typ „aml:FunctionBlock“ gibt eine Art von Funktionsblock für die AML-Logikbeschreibung an, in der nur die erforderlichen Merkmale des Funktionsblocks nach IEC 61131-3 unterstützt werden. Sein Inhalt wird in 8.3 erläutert.

8.2.1.4 Untergeordnetes Element „Documentation“

Das Element „Documentation“ enthält eine allgemeine Beschreibung des Elements „AMLLogic“. Der Typ „ppx:TextBase“ ist ein abstrakter Typ, der aus IEC 61131-10 importiert wurde und durch den Klartext von „ppx:SimpleText“ ersetzt wird.

8.3 Komplexer Typ „FunctionBlock“

8.3.1 Allgemeine Angaben

Der komplexe Typ „FunctionBlock“ stellt (wie aus Bild 6 hervorgeht) eine Art von Funktionsblock für die AML-Logikbeschreibung dar, bei dem nur die erforderlichen Merkmale des Funktionsblocks aus IEC 61131-3 unterstützt werden.

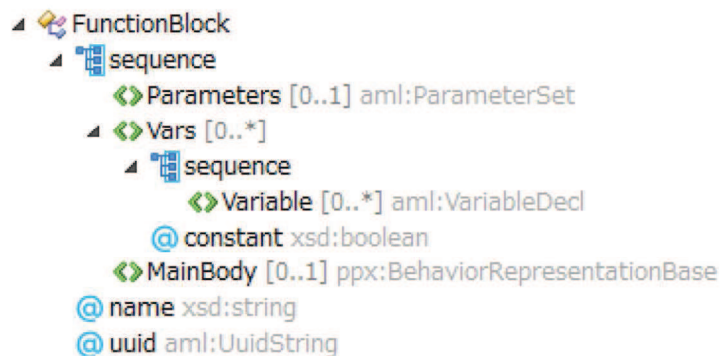


Bild 6 – Komplexer Typ „FunctionBlock“

8.3.1.1 Attribute

Das Attribut „name“ von „xsd:string“ gibt den Typnamen des Funktionsblocks an.

Das Attribut „uuid“ des Typs „aml:UuidString“ gibt den universellen eindeutigen Bezeichner der Typdefinition des Funktionsblocks an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.

8.3.1.2 Untergeordnetes Element „Parameters“

Das Element „Parameters“ vom Typ „aml:ParameterSet“ gibt die Deklaration der Eingangs-, Ausgangs- oder Eingangs-/Ausgangsvariable des Funktionsblocks an. Der Typ „ParameterSet“ wird detailliert in 8.3.2 erläutert.

8.3.1.3 Untergeordnetes Element „Vars“

Das Element „Vars“ deklariert die internen Variablen des Funktionsblocks.

Das Element „Variable“ des Typs „aml:VariableDecl“ enthält die Variablendeklaration. Der komplexe Typ „aml:VariableDecl“ wird in 8.3.3 erläutert.

Das Attribut „constant“ legt fest, dass die Variablendeklaration im Element „Vars“ Variablen mit konstanten Werten festlegen.

8.3.1.4 Untergeordnetes Element „MainBody“

Das Element „MainBody“ des Typs „ppx:BehaviorRepresentationBase“ stellt ein Logikmodell dar, das bei Ausführung des Funktionsblocks evaluiert werden muss. Jeder von „ppx:BehaviorRepresentationBase“ abgeleitete komplexe, nicht abstrakte Typ kann gegen den Typ des Elements ausgetauscht werden. Damit stehen auch alle im Schema nach IEC 61131-10 definierten Logikmodelle in diesem Funktionsblock des AML logic XML zur Verfügung, wie beispielsweise die Ablaufsprache (SFC), das Funktionsblockdiagramm (FBD), der Kontaktplan(LD) und der strukturierte Text (ST).

8.3.2 Komplexer Typ „ParameterSet“

Der komplexe Typ „ParameterSet“ (siehe Bild 7) stellt einen Deklarationssatz bestehend aus Eingangsvariablen, Ausgangsvariablen und Eingangs-/Ausgangsvariablen dar.

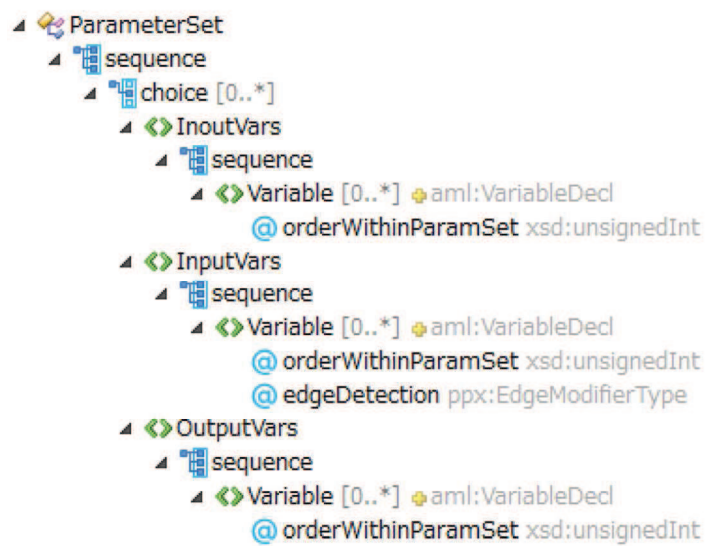


Bild 7 – Komplexer Typ „ParameterSet“

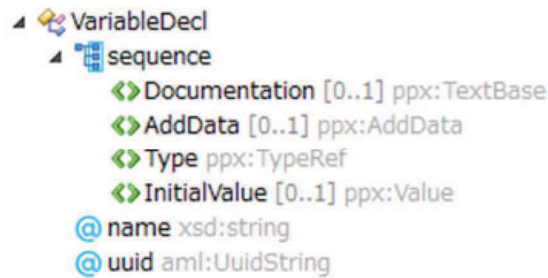
Das Element „InoutVars“ bezeichnet den Deklarationsabschnitt für Eingangs-/Ausgangsvariablen. Auf das Element „Variable“ in „InoutVars“ bezogen wird sein Typ ausgehend vom Typ „aml:VariableDecl“ erweitert, der in 8.3.3 erläutert wird. Das Attribut „orderWithinParamSet“ wird dabei hinzugefügt. Der Grund für diese Erweiterung besteht darin, dass die Reihenfolge der Parameter eines Funktionsblocks von Bedeutung ist, die Reihenfolge der Elemente eines Elementtyps im XML-Dokument dagegen nicht als signifikant sichergestellt ist.

Das Element „InputVars“ enthält einen Deklarationsabschnitt für die Eingangsvariablen des Funktionsblocks. Auf das Element „Variable“ in „InputVars“ bezogen wird sein Typ ausgehend vom Typ „aml:VariableDecl“ erweitert, der ebenfalls in 8.3.3 erläutert wird. Das Attribut „edgeDetection“ wird dabei zusätzlich zum Attribut „orderWithinParamSet“ hinzugefügt.

Das Element „OutputVars“ enthält einen Deklarationsabschnitt für die Ausgangsvariablen des Funktionsblocks. Auf das Element „Variable“ in „OutputVars“ bezogen wird sein Typ ausgehend vom Typ „aml:VariableDecl“ erweitert, der ebenfalls in 8.3.3 erläutert wird. Das Attribut „orderWithinParamSet“ wird dabei ebenso hinzugefügt.

8.3.3 Komplexer Typ „VariableDecl“

Der komplexe Typ „VariableDecl“ (siehe Bild 8) stellt einen Deklarationsabschnitt für Variablen dar.

**Bild 8 – Komplexer Typ „VariableDecl“**

Das Element „Documentation“ des Typs „ppx:TextBase“ enthält eine Beschreibung der Variable.

Das Element „AddData“ des Typs „ppx:AddData“ kann verwendet werden, um jeden Typ von zusätzlichen Daten zu speichern.

Das Element „Type“ des Typs „ppx:TypeRef“ enthält die Referenzierung der Typdefinition der Variable. Dabei kann es sich um eine Typbezeichnung oder eine Instanzendeklaration handeln, wie „ARRAY [0..8] OF INT“. Für Einzelheiten des komplexen Typs „ppx:TypeRef“ sollte auf die Spezifikation von IEC 61131-10 verwiesen werden.

Das Element „InitialValue“ des Typs „ppx:Value“ enthält den Ausgangswert der Variable. Dabei kann es sich, je nach Datentyp der Variable, um einen einzelnen Wert, einen Feldwert für einen Feldtyp oder, bei einem strukturierten Datentyp, um einen strukturierten Wert handeln. Für Einzelheiten des komplexen Typs „ppx:Value“ sollte auf die Spezifikation von IEC 61131-10 verwiesen werden.

Das Attribut „name“ des Typs „xsd:string“ gibt den Namen der Variable an.

Das Attribut „uuid“ des Typs „aml:UuidString“ gibt den universellen eindeutigen Bezeichner der Variable. Der einfache Typ „UuidString“ wird in 8.9 erläutert.

8.4 Komplexer Typ „IML“

8.4.1 Allgemeine Angaben

Der komplexe Typ „IML“ (siehe Bild 9) stellt den Intermediate Modelling Layer mehrerer bestimmter Logikmodelle nach der Beschreibung in A.3 dar. Dieser komplexe Typ kann als Ersatz für den Typ des Elements „MainBody“ im Funktionsblock der AML logic XML verwendet werden, da er nach der Beschreibung in Bild 4 „ppx:BehaviorRepresentationBase“ erweitert.

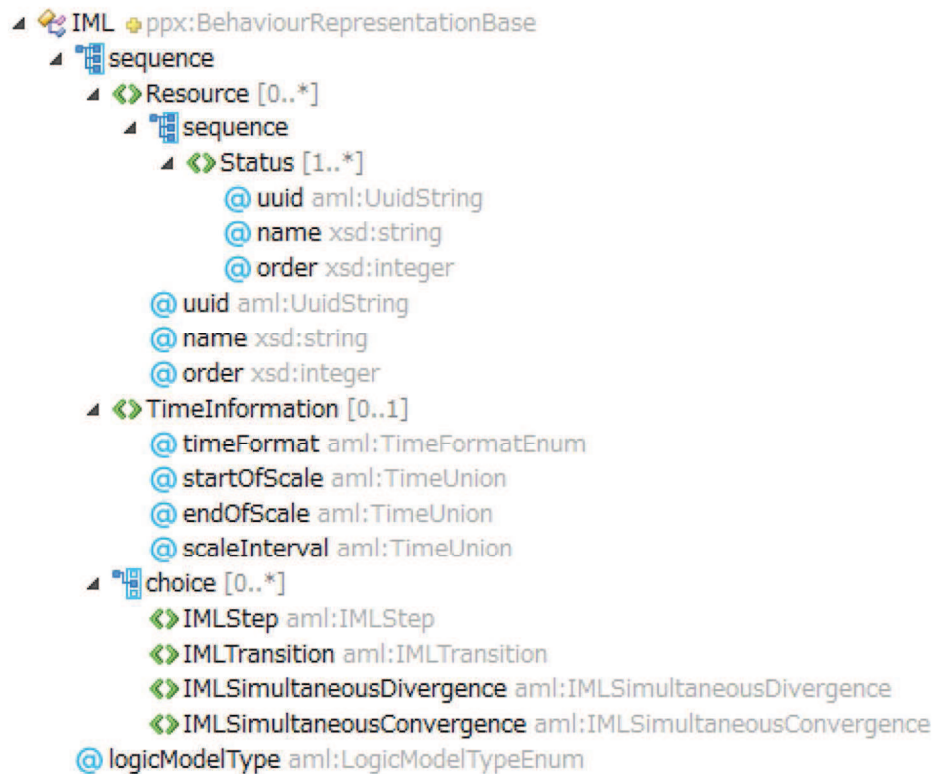


Bild 9 – Komplexer Typ „IML“

8.4.1.1 Attribute

Das Attribut „logicModelType“ des Typs „aml:LogicModelTypeEnum“ legt fest, welche Art von Logikmodell mit der IML dargestellt wird. Der einfache Typ „LogicModelTypeEnum“ wird in 8.6 erläutert.

8.4.1.2 Untergeordnetes Element „Resource“

Das Element „Resource“ gibt zusätzliche Logikinformationen an, die nach der Beschreibung in B.4.4 für das Zeitverlaufdiagramm erforderlich sind.

- Das Attribut „uuid“ gibt den universellen eindeutigen Bezeichner des „Resource“-Elements an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.
- Das Attribut „name“ gibt den Namen des „Resource“-Elements an.
- Das Attribut „order“ gibt die Reihenfolge des „Resource“-Elements in der IML an. Kleinere Zahlen sollten früher aufgeführt werden.

„Status“-Elemente geben mögliche Zustände des „Resource“-Elements an.

- Das Attribut „uuid“ gibt den universellen eindeutigen Bezeichner des „Status“-Elements an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.
- Das Attribut „name“ gibt den Namen des „Status“-Elements an.
- Das Attribut „order“ gibt die Reihenfolge des „Status“-Elements im „Resource“-Element an. Kleinere Zahlen sollten früher aufgeführt werden.

8.4.1.2 Untergeordnetes Element „TimeInformation“

„TimeInformation“ gibt die logische Zeitskala an, die insbesondere für Gantt- oder Zeitverlaufdiagramme verwendet wird.

- Das Attribut „timeFormat“ des Typs „aml:TimeFormatEnum“ stellt das Format des Zeitausdrucks in dieser IML dar. Der einfache Typ „aml:TimeFormatEnum“ wird in 8.8 erläutert.

- Das Attribut „startOfScale“ des Typs „aml:TimeUnion“ gibt den Startpunkt der Zeitskala in dem durch das Attribut „timeFormat“ oben festgelegten Format an. Der einfache Typ „aml:TimeUnion“ wird in 8.7 erläutert.
- Das Attribut „endOfScale“ des Typs „aml:TimeUnion“ gibt den Endpunkt der Zeitskala in dem durch das Attribut „timeFormat“ oben festgelegten Format an. Der einfache Typ „aml:TimeUnion“ wird in 8.7 erläutert.
- Das Attribut „scaleInterval“ des Typs „aml:TimeUnion“ gibt das Intervall der Zeitskala in dem durch das Attribut „timeFormat“ oben festgelegten Format an. Der einfache Typ „aml:TimeUnion“ wird in 8.7 erläutert.

8.4.1.4 Auswahl des „IML****“-Elements

Zur Auswahl des „choice“-Elements können „IMLStep“, „IMLTransition“, „IMLSimultaneousDivergence“ oder „SimultaneousConvergence“ verwendet werden. Diese Schemastruktur entspricht im Grunde der des komplexen Typs „SFC“ aus IEC 61131-10. Allerdings sind die anderen Möglichkeiten für diese Auswahl auf das begrenzt, was für die IML-Darstellung erforderlich ist. Außerdem wird der Typ jedes anderen Elements ausgehend vom ursprünglichen Element aus IEC 61131-10 erweitert.

8.4.2 Komplexer Typ „IMLStep“

Der komplexe Typ „IMLStep“ (siehe Bild 10) gibt eine Phase oder einen Zustand einer Aktivität an. Er erweitert den Typ „ppx:Step“ aus IEC 61131-10, mit dem ein Ablaufschritt (STEP) in der Ablaufsprache definiert wird.

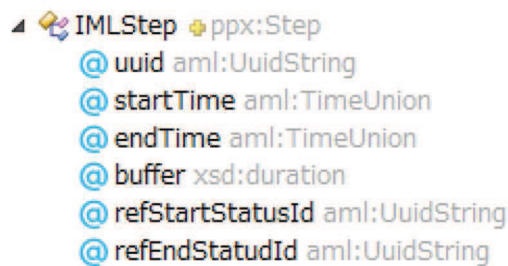


Bild 10 – Komplexer Typ „IMLStep“

Die folgenden Attribute werden als Erweiterung des Basis-Typs „ppx:Step“ aus IEC 61131-10 hinzugefügt.

- Das Attribut „uuid“ des Typs „aml:UuidString“ gibt den universellen eindeutigen Bezeichner der Variable an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.
- Das Attribut „startTime“ des Typs „aml:TimeUnion“ gibt die Startzeit der Phase der Aktivität an, die diesem IMLStep entspricht. Der einfache Typ „aml:TimeUnion“ wird in 8.7 erläutert. Die Zeit muss in dem mit dem Attribut „timeFormat“ des Elements „TimeInformation“ festgelegten Format angegeben werden. Das Element wird in 8.4.1.3 erläutert.
- Das Attribut „endTime“ des Typs „xds:duration“ gibt den Endpunkt der Phase der Aktivität an, die diesem IMLStep entspricht. Der einfache Typ „aml:TimeUnion“ wird in 8.7 erläutert. Die Zeit muss in dem mit dem Attribut „timeFormat“ des Elements „TimeInformation“ festgelegten Format angegeben werden. Das Element wird in 8.4.1.3 erläutert.
- Das Attribut „buffer“ des Typs „xsd:duration“ gibt die Zeitspanne zwischen „startTime“ und „endTime“ an.
- Das Attribut „refStartStatusId“ des Typs „aml:UuidString“ gibt die Referenzierung zum Startzustand des Segments des Pseudokurvenverlaufs an, die das Element „IMLStep“ darstellt, falls die IML ein Zeitverlaufdiagramm definiert. Konkret wird hiermit der Wert des Attributs „uuid“ eines bestimmten „Status“-Elements innerhalb des „Resource“-Elements im „IML“-Element angegeben.
- Das Attribut „refEndStatusId“ des Typs „aml:UuidString“ gibt die Referenzierung des Zielzustandes des Segments des Pseudokurvenverlaufs an, die das Element „IMLStep“ darstellt, falls die IML ein Zeitverlaufdiagramm definiert. Konkret wird hiermit der Wert des Attributs „uuid“ eines bestimmten „Status“-Elements innerhalb des „Resource“-Elements im „IML“-Element angegeben.

8.4.3 Komplexer Typ „IMLTransition“

Der komplexe Typ „IMLTransition“ (siehe Bild 11 – Komplexer Typ „IMLTransition“) stellt eine Art Vorgänger-Nachfolger-Beziehung zwischen „IMLStep“-Elementen dar. Er erweitert den Typ „ppx:Transition“ aus IEC 61131-10, mit dem ein Zustandsübergang (TRANSITION) in der Ablaufsprache definiert wird.



Bild 11 – Komplexer Typ „IMLTransition“

Die folgenden Attribute werden als Erweiterung des Basis-Typs „ppx:Transition“ aus IEC 61131-10 hinzugefügt.

- Das Attribut „uuid“ des Typs „aml:UuidString“ gibt den universellen eindeutigen Bezeichner des Elements „IMLTransition“ an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.
- Das Attribut „refTime“ gibt die Zeit an, zu der dieser Übergang ausgelöst werden sollte. Der einfache Typ „aml:TimeUnion“ wird in 8.7 erläutert. Die Zeit muss in dem mit dem Attribut „timeFormat“ des Elements „TimeInformation“ festgelegten Format angegeben werden. Das Element wird in 8.4.1.3 erläutert.
- Das Attribut „refEndSegmentId“ des Typs „aml:UuidString“ stellt eine Referenzierung des Elements „IMLStep“ anhand eines UUID dar, dessen Ende diesen Übergang auslösen sollte.

8.4.4 Komplexer Typ „IMLSimultaneousDivergence“

Der komplexe Typ „IMLSimultaneousDivergence“ erweitert „ppx:SimultaneousDivergence“ aus IEC 61131-10 (siehe Bild 12), das einen Startpunkt einer parallelen Verzweigung in der Ablaufsprache angibt.

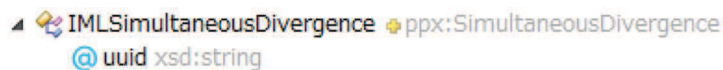


Bild 12 – Komplexer Typ „IMLSimultaneousDivergence“

Die folgenden Attribute werden als Erweiterung des Basis-Typs „ppx:SimultaneousDivergence“ aus IEC 61131-10 hinzugefügt.

- Das Attribut „uuid“ des Typs „aml:UuidString“ gibt den universellen eindeutigen Bezeichner des Elements „IMLSimultaneousDivergence“ an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.

8.4.5 Komplexer Typ „IMLSimultaneousConvergence“

Der komplexe Typ „IMLSimultaneousConvergence“ erweitert „ppx:SimultaneousConvergence“ aus IEC 61131-10 (siehe Bild 13), das einen Endpunkt einer parallelen Verzweigung in der Ablaufsprache angibt.

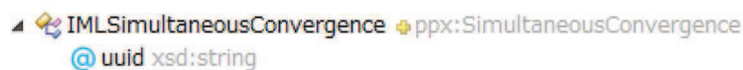


Bild 13 – Komplexer Typ „IMLSimultaneousConvergence“

Die folgenden Attribute werden als Erweiterung des Basis-Typs „ppx:SimultaneousConvergence“ aus IEC 61131-10 hinzugefügt.

- Das Attribut „uuid“ des Typs „aml:UuidString“ gibt den universellen eindeutigen Bezeichner des Elements „IMLSimultaneousConvergence“ an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.

8.5 Komplexer Typ „MathematicalExpression“

8.5.1 Allgemeine Angaben

Der komplexe Typ „MathematicalExpression“ (siehe Bild 14) stellt eine logische Beschreibung als mathematischen Ausdruck dar. Sie kann als Ersatz für das Element „MainBody“ des Funktionsblocks in der AML logic XML verwendet werden, da sie nach der Beschreibung in Bild 4 den abstrakten komplexen Typ „ppx:BehaviorRepresentationBase“ erweitert.



Bild 14 – Komplexer Typ „MathematicalExpression“

8.5.1.1 Attribute

Das Attribut „name“ des Typs „string“ gibt den Namen des mathematischen Ausdrucks an.

Das Attribut „uuid“ gibt den universellen eindeutigen Bezeichner des mathematischen Ausdrucks an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.

8.5.1.2 Untergeordnetes Element „VariableMapping“

Das Element „VariableMapping“ gibt die Zuordnung von in MathML definierten Variablen zu den Variablen an, die in dem Funktionsblock deklariert werden, zu dem dieses Logikmodell gehört.

„refMathMLVariableName“ gibt den Namen der in MathML definierten Variable an.

„refFBVariableUUID“ gibt der UUID der im gleichen Funktionsblock deklarierten Variable an. Der einfache Typ „UuidString“ wird in 8.9 erläutert.

Das Attribut „direction“ gibt anhand einer Aufzählungszeichenkette die Zuordnungsrichtung an, bei der nur „In“ oder „Out“ zugelassen ist.

8.5.1.3 Untergeordnetes Element „MathML“

Das Element „MathML“ des Typs „string“ gibt die XML-Textdaten von MathML an. Es wird einfach als Klartext gespeichert. Das Parsing mit MathML-Schemata liegt außerhalb des Anwendungsbereichs des AML logic XML Schemas.

8.6 Einfacher Typ „LogicModelTypeEnum“

Der einfache Typ „LogicModelTypeEnum“ ist ein eingeschränkter „string“-Typ, der nur eine der folgenden Aufzählungszeichenketten zulässt: „GanttChart“, „ActivityOnNodeNetwork“, „TimingDiagram“ oder „Unkown“. Mit diesem Typ wird angegeben, welche Art von Logikmodelltyp die IML darstellt.

8.7 Einfacher Typ „TimeUnion“

Der einfache Typ „TimeUnion“ ist eine Vereinigungsmenge, deren Mitglieder die folgenden zeitbezogenen einfachen Typen des XML-Schemas umfassen. Zur Spezifikation des detaillierten Werteformats jedes integrierten Typs muss die Spezifikation des XML-Schemas referenziert werden.

- xsd:duration
- Das Werteformat muss „PnYnMnDTnHnMnS“ lauten, beispielsweise „P10DT15H“ (10 Tage und 15 Stunden), „PT1M20S“ (1 Minute und 20 Sekunden), „PT95.4“ (95,4 Sekunden) usw.
- xsd:time
- Das Werteformat muss „hh:mm:ss“ lauten, beispielsweise einfach „15:45:30“, oder „15:45:30+02:00“, um eine Zeitzonendifferenz zu UTC anzugeben.
- xsd:dateTime
- Das Werteformat muss „JJJJ-MM-TTThh:mm:ss“ lauten, beispielsweise einfach „2017-05-30T09:00:00“, oder „2017-05-30T09:00:00+02:00“, um eine Zeitzonendifferenz zu UTC anzugeben.
- xsd:date
- Das Werteformat muss „JJJJ-MM-TT“ lauten, beispielsweise „2017-09-24“.

8.8 Einfacher Typ „TimeFormatEnum“

Der einfache Typ „TimeFormatEnum“ ist ein eingeschränkter „string“-Typ, der nur eine der folgenden Aufzählungszeichenketten zulässt: „xsd:duration“, „xsd:time“, „xsd:dateTime“ oder „xsd:date“. Mit diesem Typ wird angegeben, welches Zeitformat für den in 8.7 erläuterten Typ „aml:TimeUnion“ verwendet werden muss.

8.9 Einfacher Typ „UuidString“

Der einfache Typ „UuidString“ ist ein eingeschränkter „string“-Typ zur Darstellung des UUID als Text. Eine Syntaxeinschränkung wird dabei faktisch nicht definiert.

ANMERKUNG Es wird empfohlen, einen UUID als GUID (nach der Spezifikation in RFC 4122) zu implementieren.

9 Verknüpfung von AML-Objekten mit Verriegelungsinformationen

9.1 Allgemeine Angaben

Verriegelungsinformationen sind ein bedeutender Teil der Planung von Produktionssystemen, mit dem die Sicherheit deren Verhaltens umgesetzt werden soll. Dieser Abschnitt befasst sich mit dem Konzept der Verriegelung, das zwei Detailstufen zur Modellierung und Speicherung von Verriegelungsinformationen in AML umfasst. Ein informativer Überblick über das Verriegelungskonzept ist in Anhang F enthalten.

ANMERKUNG In diesem Teil bezieht sich die Verriegelung auf sicherheitsrelevante Abhängigkeiten zwischen Anlagen, beispielsweise Lichtschranken, die verhindern, dass Menschen durch einen Roboter verletzt werden. Dagegen befasst sich dieser Teil nicht mit den funktionsrelevanten Abhängigkeiten zwischen Anlagen, dass also beispielsweise bei zwei zusammenarbeitenden Robotern einer warten muss, bis der andere einen Vorgang abgeschlossen hat.

9.2 Referenzierung von Verriegelungsinformationen

Auf der ersten Detailstufe des Verriegelungskonzepts werden Verriegelungsquellengruppen und Verriegelungszielgruppen verknüpft, um funktionale Abhängigkeiten unter AML-Objektgruppen bzw. AML-Objekten zu beschreiben. Hierzu gelten die folgenden Festlegungen:

- Eine Verriegelungszielgruppe muss durch die Verwendung des AML-Gruppenkonzepts nach IEC 62714-1 erstellt werden.
 - Zur Identifizierung der Gruppe als Verriegelungszielgruppe muss die Rollenklasse „InterlockingTargetGroup“ verwendet werden, die in 5.2.2 beschrieben wird.

ANMERKUNG Die Objekte einer Verriegelungszielgruppe führen die Aktionen aus, die zur Sicherstellung der funktionalen Sicherheit erforderlich sind.

- Eine Verriegelungsquellengruppe muss durch die Verwendung des AML-Gruppenkonzepts nach IEC 62714-1 erstellt werden.
 - Zur Identifizierung der Gruppe als Verriegelungsquellengruppe muss die Rollenklasse „InterlockingSourceGroup“ verwendet werden, die in 5.2.3 beschrieben wird.

ANMERKUNG Die Objekte einer Verriegelungsquellengruppe geben an, wann Aktionen zur Sicherstellung der funktionalen Sicherheit erforderlich sind.

- Jede Verriegelungszielgruppe und jede Verriegelungsquellengruppe muss über ein CAEX-„ExternalInterface“-Element der AML-Schnittstellenklasse „InterlockingConnector“ verfügen, das in 5.3.11 beschrieben wird.
 - Das CAEX-„ExternalInterface“-Element der Verriegelungsquellengruppe muss über ein „InternalLink“-Element mit dem CAEX-„ExternalInterface“-Element der Verriegelungszielgruppe verknüpft werden.

ANMERKUNG 1 Ein bestimmter Zustand der Verriegelungsquellengruppe hat Auswirkungen auf die Verriegelungszielgruppe.

ANMERKUNG 2 Ein AML-Objekt kann, muss aber nicht, in beliebig vielen Verriegelungsgruppen enthalten sein.

ANMERKUNG 3 Ein AML-Objekt kann zur gleichen Zeit sowohl in einer Verriegelungsquellengruppe als auch in einer Verriegelungszielgruppe enthalten sein.

ANMERKUNG 4 Die Zuweisung zwischen beiden Gruppen ist nicht auf eine 1:1-Beziehung beschränkt. Es kann mehrere Verriegelungsquellengruppen geben, die mit einer oder mit mehreren Verriegelungszielgruppen verbunden sind.

ANMERKUNG 5 Die Beziehungen zwischen den verschiedenen Gruppen sind unabhängig.

Auf der weiten Detailstufe des Verriegelungskonzepts wird die einfache Zuordnung von AML-Objekten zu den Verriegelungsquellengruppen und Verriegelungszielgruppen (aus der ersten Detailstufe) durch eine Funktion erweitert, mit der die Verriegelungsbedingung der Verriegelungsquellengruppen beschrieben wird. Hierzu gelten die folgenden Festlegungen:

- Die Funktion muss durch einen Funktionsblock des Typs „FBD“ dargestellt werden, der in Abschnitt 8 beschrieben wird.
- Dieser Funktionsblock muss nach den Angaben in 5.3.8 vom CAEX-„ExternalInterface“-Element der AML-Schnittstellenklasse „InterlockingLogicModelInterface“ referenziert werden.
- Der Funktionsblock muss in einer eindeutigen Booleschen Variablen resultieren, die das Ergebnis bzw. die Bewertung dieses Funktionsblocks beschreibt.

ANMERKUNG Der Wert der Booleschen Variable kann „true“ (wahr) oder „false“ (falsch) annehmen. Dies gibt an, ob der sichere Zustand der Verriegelungsquellengruppe anhand eines der Werte „true“ (wahr) oder „false“ (falsch) dargestellt wird.

- Die eindeutige Boolesche Variable muss anhand einer Verriegelungsquellengruppe referenziert werden, die nach den Angaben in 5.3.12 ein CAEX-„ExternalInterface“-Element der AML-Schnittstellenklasse „InterlockingVariableInterface“ modelliert.
 - Dieses CAEX-„ExternalInterface“-Element kann über ein Attribut „SafeConditionEquals“ verfügen, in dem der Boolesche Wert zur Angabe des sicheren Zustands gespeichert werden muss. Der Standardwert lautet „true“ (wahr).
- Anhand des im Abschnitt 7 beschriebenen Mechanismus müssen die „ExternalInterface“-Elemente der Eingangs- und Ausgangsvariablen des Funktionsblocks mithilfe von „InternalLink“-Elementen mit dem „ExternalInterface“-Element des Verriegelungslogikmodells verbunden werden.
 - Bei der Modellierung der Reaktion der betreffenden Verriegelungszielgruppen müssen die gleichen Festlegungen beachtet werden, die auch für die Verriegelungsquellengruppen festgelegt wurden.

Anhang A (informativ)

Logikinformationen in AML

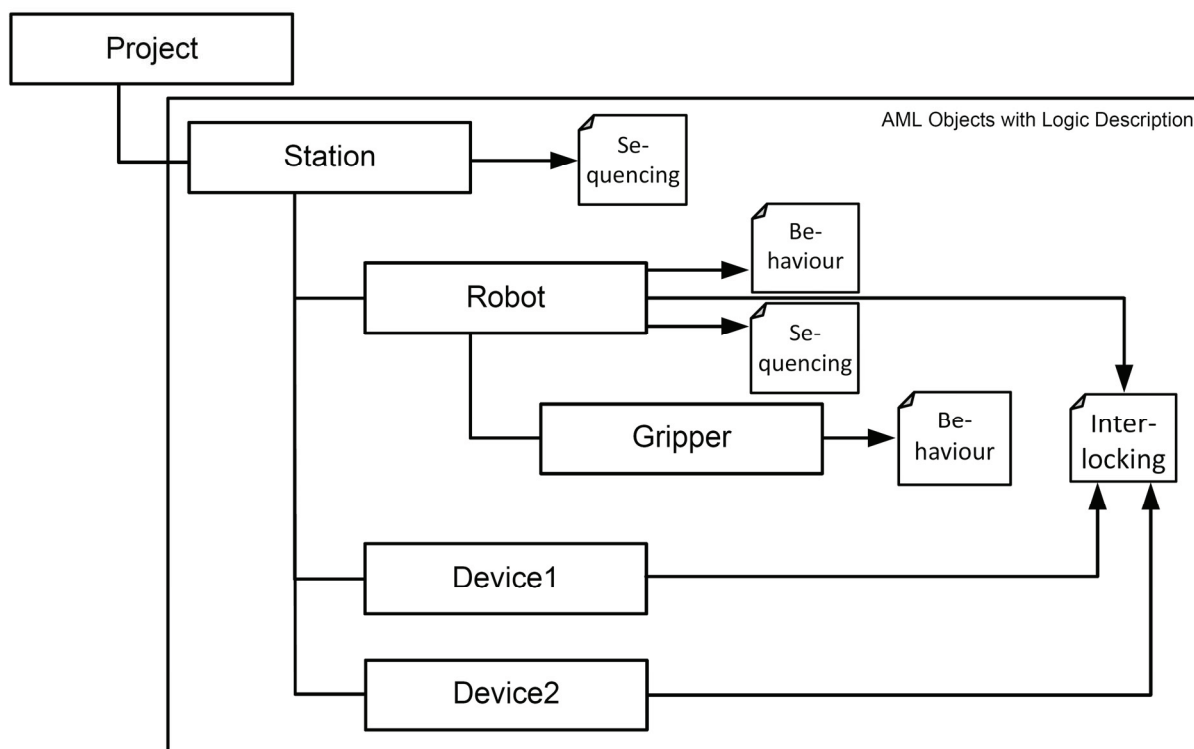
A.1 Logikinformationen in der Planung von Produktionssystemen

Dieser Teil befasst sich mit der Speicherung von Logikinformationen, einem wesentlichen Punkt des elektrischen Designs, der HMI-Entwicklung, der Steuerungs- und Robotersteuerungsprogrammierung, der Simulation und der virtuellen Inbetriebnahme.

Zur Unterstützung verschiedener Phasen im iterativen Planungsprozess von Produktionssystemen und der unterschiedlichen Detailstufen muss AML in der Lage sein, die Logikinformationen verschiedener Werkzeuge und Bereiche abzudecken. So müssen verschiedene Arten Logikinformationen gespeichert werden, die zu einem ganzen Produktionssystem oder zu betreffenden Einzelkomponenten gehören. Diese Vielfalt an Daten kann in zwei Gruppen unterteilt werden: Ablaufinformationen und Verhaltensinformationen. Beide Informationsarten dienen unterschiedlichen Zwecken, die sich allerdings je nach Sichtweise und Nutzung der Elemente in AML überschneiden können (siehe Bild A.1).

- Ablaufinformationen sind Informationen, mit denen die Anweisungen für ein geregeltes System (oder Subsystem) beschrieben werden. Auf der höchsten Detailstufe werden sie häufig durch ein Programm abgebildet, das in einer oder mehreren Regeleinheiten ausgeführt wird, beispielsweise in einer Roboter- oder Programmlogiksteuerung. Aber die Entwicklung von Abläufen beginnt – bei Modellen mit einer hohen Abstraktionsstufe – früher Planungsprozess, und zwar bei Modellen, welche die erforderlichen Vorgänge einer größeren Maßstabseinheit darstellen (Zelle, Linie, Anlage usw.).
- Verhaltensinformationen sind Informationen, mit denen die Reaktion eines geregelten Systems (oder Subsystems) auf Ablaufinformationen und andere externe Wechselwirkungen beschrieben werden. Sie werden häufig anhand eines bestimmten Modells dargestellt, das auf externe Eingänge reagiert, beispielsweise das Verhalten eines Greifers oder Ventils. Diese externen Eingänge würden das Verhalten eines Greifers oder die Anzeige seiner Zustände auslösen.
- Neben den Ablauf- und Verhaltensinformationen gibt es mit den Verriegelungsinformationen einen dritten ergänzenden Typ von Logikinformationen (siehe Bild A.1). Sie sind ein bedeutender Teil der Planung von Produktionssystemen, mit dem die Sicherheit deren Verhaltens umgesetzt werden soll. Sie beeinflussen nicht nur die sichere Interaktion mit Menschen und der Umgebung sondern tragen auch zur Vermeidung instabiler Systemzustände bei, die unter Umständen zu schädigenden Einflüssen für Menschen und der Umgebung oder zu Schäden an Maschinen und Produkten führen.

Eine Regelstrecke (oder ein Subsystem) kann durch mehr als nur eine Art der oben genannten Informationen in AML dargestellt werden, beispielsweise ein Roboter (siehe Bild A.1).



Legend:

- Reference
- Hierarchical relation
- IEC 61131-10 document

Bild A.1 – Beispiel für eine Systemdarstellung mit verschiedenen Informationsarten in AML

Project	Projekt
Stations	Station
Sequencing	Abfolge
AML Objects with Logic Description	AML-Objekte mit Logikbeschreibung
Robot	Roboter
Behaviour	Verhalten
Gripper	Greifer
Interlocking	Verriegelung
Device	Gerät
Legend	Legende
Reference	Referenzierung
Hierarchical relation	Hierarchische Beziehung
IEC 61131-10 document	IEC 61131-10

A.2 Logikmodelle in der Planung von Produktionssystemen

Damit AML im Rahmen des Planungsprozesses von Produktionssystemen anwendbar ist, muss diese Auszeichnungssprache übliche Logikmodelle unterstützen und deren Logikinformationen speichern, welche die einzelnen Phasen des Planungsprozesses abdecken (siehe Bild A.2). Bei diesen Logikmodellen handelt es sich um die folgenden:

- In den frühen Stadien des Planungsprozesses werden hauptsächlich Gantt-Diagramme verwendet, um die Zeitplanung und Dauer der Herstellungsprozesse auf einer hohen Abstraktionsebene darzustellen (siehe B.2).
- Vorgangsknotennetze dienen in den frühen Stadien des Planungsprozesses in erster Linie zur Darstellung der Zeitplanung und der wechselseitigen Beziehungen von Herstellungsprozessen. Dabei können auch Zeitplanungsinformationen gespeichert werden (siehe B.3).
- Zeitverlaufsdiagramme werden in späteren Phasen des Planungsprozesses verwendet, um die Geräte mit ihren Zuständen und Zeitbeziehungen zu beschreiben (siehe B.4). Dabei werden echte Signale eingeführt und die Möglichkeit, Ablaufinformationen auszudrücken.
- Die Ablaufsprache (SFC) zählt zu den Programmiersprachen aus IEC 61131-3 und wird im Wesentlichen im späteren Verlauf der Planung von Produktionssystemen verwendet (siehe B.5). Sowohl Ablauf- als auch Verhaltensinformationen können modelliert werden.
- Funktionsblockdiagramme (FBD) zählen ebenfalls zu den Programmiersprachen aus IEC 61131-3 und werden im Wesentlichen gleichfalls im späteren Verlauf der Planung von Produktionssystemen verwendet (siehe B.6). Sowohl Verriegelungs- als auch Verhaltensinformationen können modelliert werden.
- Mathematische Ausdrücke dienen hauptsächlich im späteren Verlauf der Planung zur Beschreibung von mathematischen, physikalischen, chemischen oder logischen Beziehungen eines Geräts (siehe B.7).

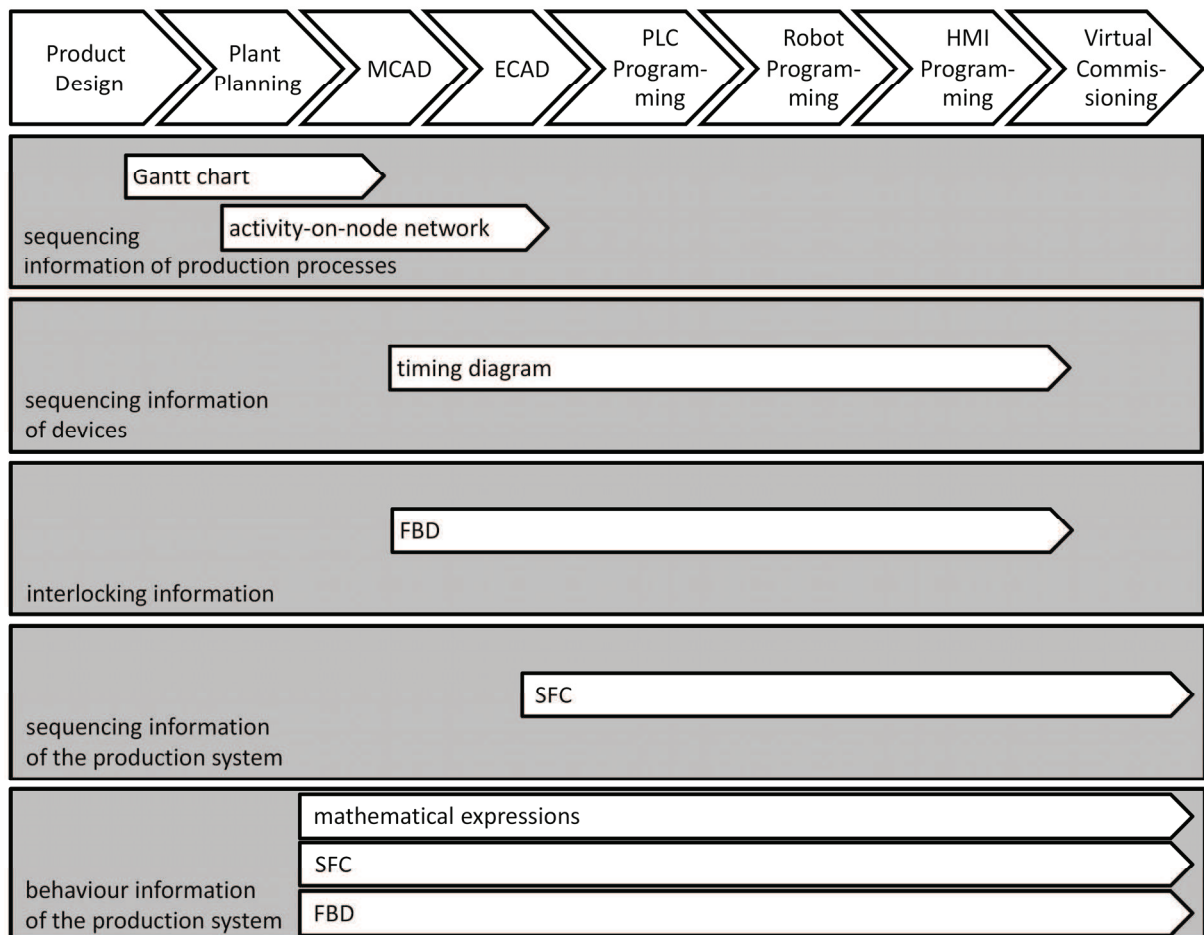


Bild A.2 – Logikmodelle in AML

Product design	Produktentwurf
Plant planning	Anlagenplanung
MCAD	MCAD

ECAD	ECAD
PLC Programming	PLC-Programmierung
Robot Programming	Roboterprogrammierung
HMI Programming	HMI-Programmierung
Virtual Commissioning	Virtuelle Inbetriebnahme
Gantt chart	Gantt-Grafik
sequencing information of production processes	Sequenzierungsdaten des Produktionsprozesses
activity-on-node network	Vorgangsknotennetz
sequencing information of devices	Sequenzierungsdaten der Geräte
timing diagram	Zeitdiagramm
interlocking information	Verriegelungsdaten
FBD	Funktionsblockdiagramm (FBD)
sequencing information of the production system	Sequenzierungsdaten des Produktionssystems
SFC	Sequentielle Funktionstabelle (SFC)
behaviour information of the production system	Verhaltensdaten des Produktionssystems
mathematical expressions	Mathematische Ausdrücke

A.3 Speicherung von Logikmodellen in der AML logic XML

Es ist beabsichtigt, die Umwandlung von einem Logikmodell in ein anderes zu ermöglichen. Dies ist von entscheidender Bedeutung, da es sich bei der Planung von Produktionssystemen um einen Datenanreicherungsprozess handelt. Dabei werden die Ablaufinformationen für ein Produktionssystem zunächst nur grob mithilfe eines Gantt-Diagramms beschrieben und müssen im nächsten Schritt detaillierter dargestellt werden. Damit wird aus dem Gantt-Diagramm ein Vorgangsknotennetz: Die Informationen aus dem Gantt-Diagramm werden übernommen, und die komplexen Zeitablaufbedingungen können hinzugefügt werden. Theoretisch kann dieser Workflow bis hin zu einem ablauffähigen Programm fortgesetzt werden, wobei sich die Informationsdichte bei jedem Schritt erhöht. Dies führt dazu, dass eine erneute Eingabe von Daten innerhalb des anschließend verwendeten Planungswerkzeugs nicht erforderlich ist. Da die Logikmodelle aber unterschiedliche Modellierungsmächtigkeiten aufweisen, ergeben sich – bei vorwärts und rückwärtsgerichteter Transformation – in bestimmten Fällen Informationsverluste, beispielsweise bei der Transformation eines Vorgangsknotennetzes in ein Gantt-Diagramm.

Ein Überblick über den Anwendungsbereich dieses Teils der Norm ist in Bild A.3 abgebildet.

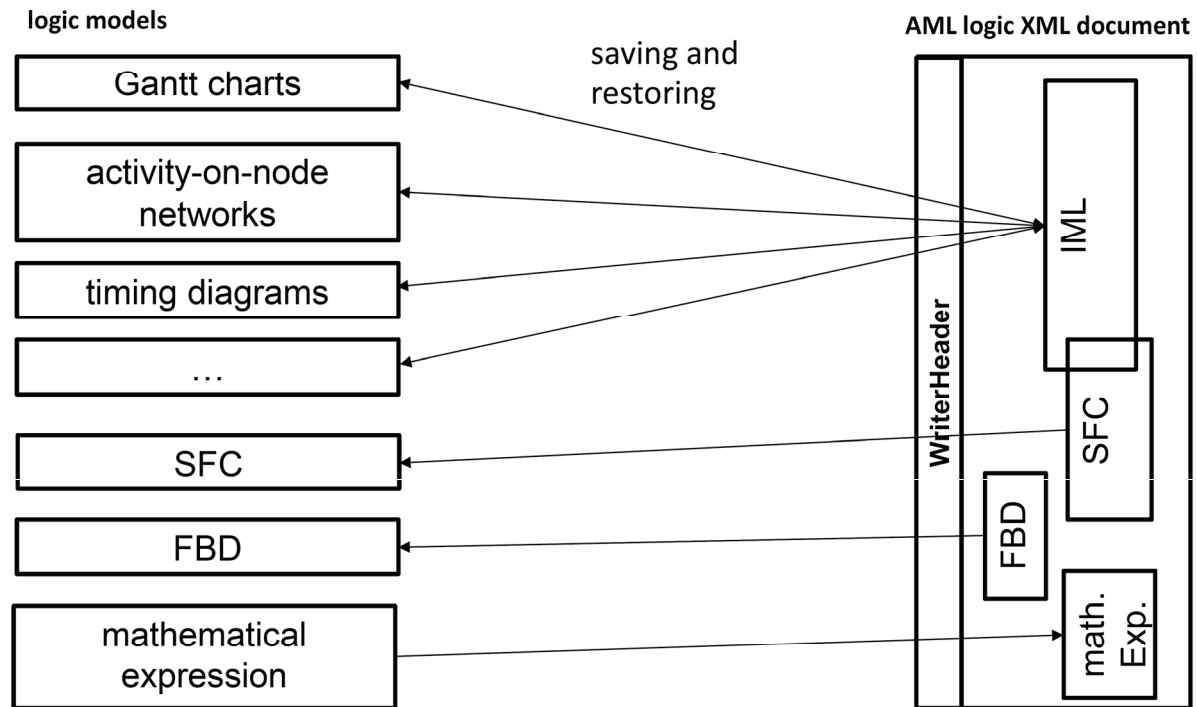


Bild A.3 – Speicherung von Logikmodellen in der AML logic XML

logic models	Logikmodelle
AML logic XML document	XML-Dokument der AML-Logik
saving and restoring	Sicherung + Wiederherstellung
Gantt charts	Gantt-Grafiken
activity-on-node networks	Vorgangsknotennetze
timing diagrams	Zeitdiagramme
SFC	SFC
FBD	FBD
mathematical expressions	Mathematischer Ausdruck
WriterHeader	Kopfdaten über den Ersteller (WriterHeader)
IML	IML
math. Exp.	Math. Ausdr.

ANMERKUNG Zur Umwandlung einer Ablaufsprache im IML-Kontext in eine Ablaufsprache im Kontext von IEC 61131-10 können einige allgemein erhältliche XSLT-Dateien verwendet werden. Bei der Konvertierung einer Ablaufsprache des IML-Kontexts (IML SFC) in eine Ablaufsprache nach IEC 61131-10 können Informationen verloren gehen.

AML verwendet IEC 61131-10, um die Logikmodelle zu speichern. Um IEC 61131-10 von den Logikmodellen zu entkoppeln, wird in diesem Teil die Intermediate Modelling Layer (IML) festgelegt. Damit bildet IML ein allgemeines Modell für Logikmodelle, mit dem die Möglichkeiten der Vorwärts- und Rückwärtstransformation und der Erweiterbarkeit von AML für neue Logikmodelle verbessert werden. Beispiele hierfür sind ereignisgesteuerte Logikmodelle, wie Zustandsdiagramme.

Da es in diesem Teil der Norm nicht nur einfach um die Speicherung von Steuerungsprogrammen geht, wird die Verwendung von IEC 61131-10 adaptiert. Hierzu wird ein AML logic XML Schema verwendet (siehe Abschnitt 8), das die Speicherung der genannten Logikmodelle erlaubt. Die Modellierungselemente des AML

logic XML, mit denen die betreffenden Logikmodelle gespeichert werden können, sind in Bild A.4 dargestellt. Auf diese Modellierungselemente können die Logikmodelle abgebildet werden. Dies wird in Anhang C beschrieben. Anhang D enthält Beispiele.

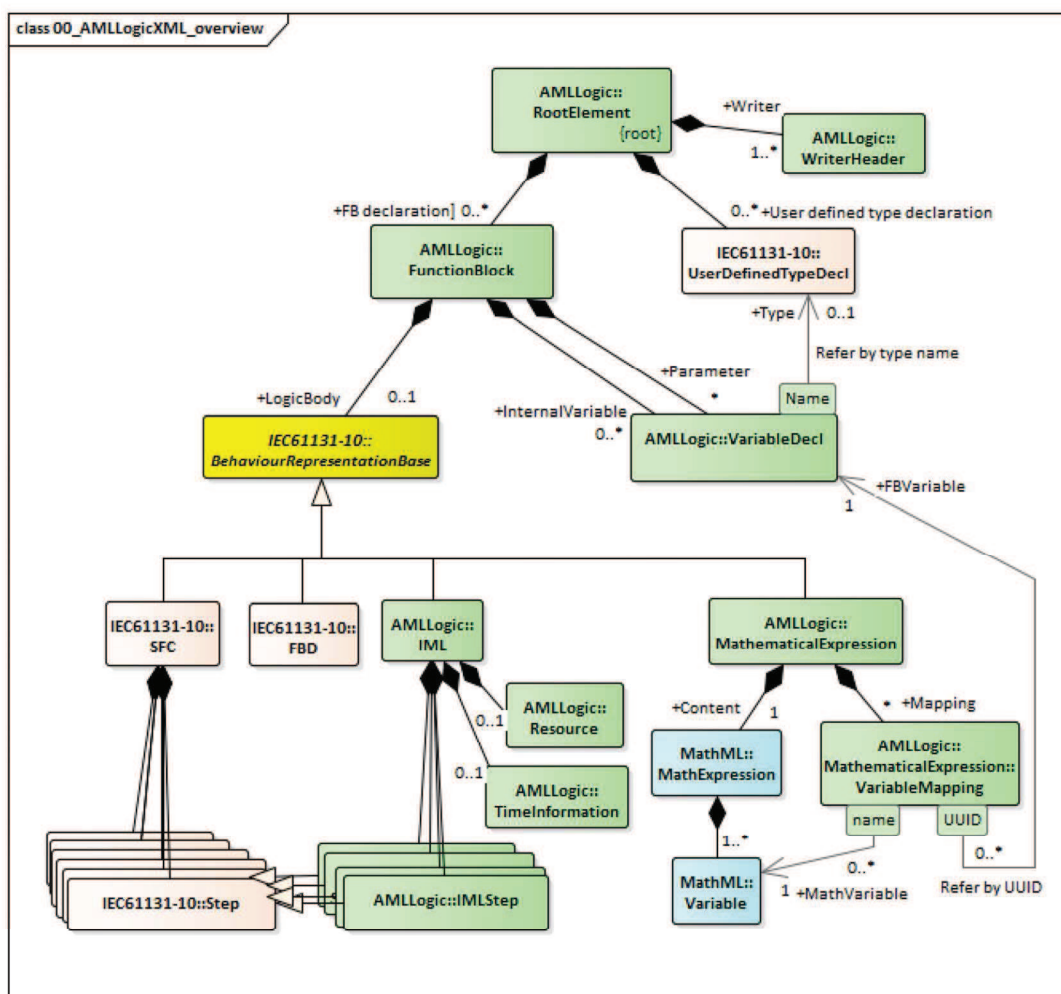


Bild A.4 – Modellierungselemente in der AML logic XML

+FB declaration	+FB-Deklaration
+Writer	+Ersteller
+User defined type declaration	+Anwenderdefinierte Typdeklaration
+ LogicBody	+Logiktext
+Type	+Typ
+Parameter	+Parameter
Refer by type name	Referenzierung nach Typname
+InternalVariable	+Interne Variable
+FBVariable	+FB-Variable
+Content	+Inhalt
+Mapping	+Zuordnung
+MathVariable	+Math. Variable
Refer by UUID	Referenzierung nach UUID

RootElement	RootElement
WriterHeader	WriterHeader
FunctionBlock	FunctionBlock
UserDefinedTypeDecl	UserDefinedTypeDecl
BehaviourRepresentationsBase	BehaviourRepresentationsBase
VariableDecl	VariableDecl
Name	Name
Step	Step
Resource	Resource
TimeInformation	TimeInformation
MathematicalExpression	MathematicalExpression
MathExpression	MathExpression
Variable	Variable
MathematicalExpression: VariableMapping	MathematicalExpression: VariableMapping

Darüber hinaus sind zur Ermöglichung eines ordnungsgemäßen Datenaustauschs im Anwendungsbereich von AML als „WriterHeader“ bezeichnete dokumentenbezogene Metadaten zum AML logic XML Dokument erforderlich, um das Dokument zu identifizieren und zu charakterisieren. Normative Festlegungen sind in Abschnitt 6 enthalten.

Da das AML logic XML auf IEC 61131-10 basiert, können auch die anderen darauf beruhenden Logikmodelle genutzt werden. In diesem Teil der Norm werden nur die Ablaufsprache (SFC) und die Funktionsblockdiagramme (FBD) als für den Anwendungsbereich dieser Norm relevant angesehen.

Auch ein als mathematischer Ausdruck dargestelltes Logikmodell kann im AML logic XML Dokument gespeichert werden. Dabei enthält entweder das Logikmodell selbst die Verhaltensinformationen des Systems, oder es kann verwendet werden, um einen mathematischen Ausdruck bereitzustellen, der auf die anderen Logikmodelle verweist.

A.4 Referenzierung von Logikinformationen in AML

In der Mehrdokumentearchitektur von AML werden die Logikinformationen in getrennten AML logic XML Dokumenten gespeichert, die auf dem Datenformat nach IEC 61131-10 basieren. Aus diesem Grund werden die Modellierungs-Logikinformationen in zwei Schritte unterteilt. Zunächst wird ein die Logikinformationen enthaltendes Objekt in dem mit CAEX bezeichneten Toplevel-Format von AML modelliert. Im zweiten Schritt muss ein AML logic XML Dokument bereitgestellt werden, das die betreffenden Logikinformationen enthält. Schließlich wird eine Beziehung zwischen dem CAEX-Objekt und dem AML logic XML Dokument oder einem darin enthaltenen Objekt hergestellt. Normative Festlegungen für die Referenzierung aller drei Informationsarten werden in Abschnitt 5 und in Abschnitt 7 definiert. Ein informativer Überblick über das Verriegelungskonzept ist in Anhang E enthalten.

Für die Referenzierung von Verriegelungsinformationen müssen weitere Festlegungen berücksichtigt werden. Diese werden normativ im Abschnitt 9 definiert. Ein informativer Überblick ist in Anhang F enthalten.

Anhang B
(informativ)

Logikmodelle in AML

B.1 Allgemeine Angaben

In diesem Anhang werden die Logikmodelle festgelegt, die in diesem Teil von IEC 62714 berücksichtigt werden. Hierzu zählen die grafischen Elemente und ihre strukturellen Regeln, die Logikinformationen, die sie darstellen und wie diese Informationen den Modellelementen zugeordnet werden.

ANMERKUNG Dieser Teil stellt keine Informationen zur Ausführungslogik bereit, d. h. zur Art und Weise, in der die Modellelemente ausgeführt werden.

B.2 Gantt-Diagramme

B.2.1 Allgemeine Angaben

Gantt-Diagramme sind grafische Darstellungen von Zeitplänen, die üblicherweise zur Modellierung von teilweise angeordneten Gruppierungen nacheinander und gleichzeitig ablaufender Aktivitäten verwendet werden. Dazu zählen auch die Ausführungsreihenfolge und -dauer dieser Aktivitäten. Sie enthalten keine Angaben zur Modellierung von alternierenden Pfaden und Zyklen.

ANMERKUNG Gantt-Diagramme werden auch als Balkengrafiken bezeichnet.

B.2.2 Grafische Elemente

In einem Gantt-Diagramm müssen die folgenden grafischen Elemente verwendet werden (siehe Bild B.1):

- eine als allgemeine Zeitachse ausgeführte Zeitleiste (horizontale Achse);
- eine vertikale Achse mit allen Namen der betreffenden Balken;
- keine oder beliebig viele horizontale Balken;
- keine oder beliebig viele gerichtete Pfeile.

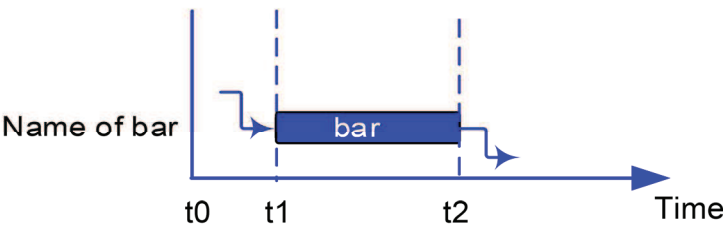


Bild B.1 – Modellelemente von Gantt-Diagrammen

Name of Bar	Name des Balkens
bar	Balken
Time	Zeit

B.2.3 Diagrammstruktur

Für ein Gantt-Diagramm muss die folgende Struktur verwendet werden:

- Jedes Gantt-Diagramm muss durch zwei orthogonale Achsen dargestellt werden. Die horizontale Achse stellt die Zeit dar, die zur Abbildung der linearen Zeit mit einer Laufrichtung versehen ist. Die vertikale Achse enthält eine Liste mit den Balkennamen.
- Jeder Balken muss in der grafischen Darstellung auf einer eigenen Zeile stehen und auf der vertikalen Achse einen entsprechenden Namen aufweisen.
- Die zeitbezogenen Start- und Endpunkte des Balkens müssen der horizontalen Achse entsprechen.
- Die Balken können mithilfe von Pfeilen mit anderen Balken verbunden werden.
- Ein Pfeil muss an der rechten Seite eines Balkens beginnen und an der linken Seite eines anderen Balkens enden.

B.2.4 Logikinformationen

Gantt-Diagramme müssen die folgenden Arten von Logikinformationen umfassen:

- Angaben zur Aktivität:
 - Name;
 - Startzeitpunkt;
 - Endzeitpunkt;
 - Dauer.
- Keine oder beliebig viele Beziehungsdaten zur vorangegangenen Aktivität.
- Keine oder beliebig viele Beziehungsdaten zur nachfolgenden Aktivität.

B.2.5 Logikinformationen in Gantt-Diagrammen

Im Hinblick auf die Darstellung der Logikinformationen in einem Gantt-Diagramm gelten die folgenden Festlegungen (siehe Bild B.2):

- Eine Aktivität muss durch genau einen Balken dargestellt werden.
 - Der Name der Aktivität muss durch den Namen des Balkens auf der vertikalen Achse dargestellt werden.
 - Der Startzeitpunkt der Aktivität muss durch einen zeitlich definierten Startpunkt des Balkens auf der horizontalen Achse dargestellt werden.
 - Der Endzeitpunkt der Aktivität muss durch einen zeitlich definierten Endpunkt des Balkens auf der horizontalen Achse dargestellt werden.
 - Die Dauer der Aktivität muss durch die Länge des Balkens angegeben werden.
- Angaben zur Vorgänger-Nachfolger-Beziehung zwischen Aktivitäten müssen durch einen Pfeil angegeben werden.

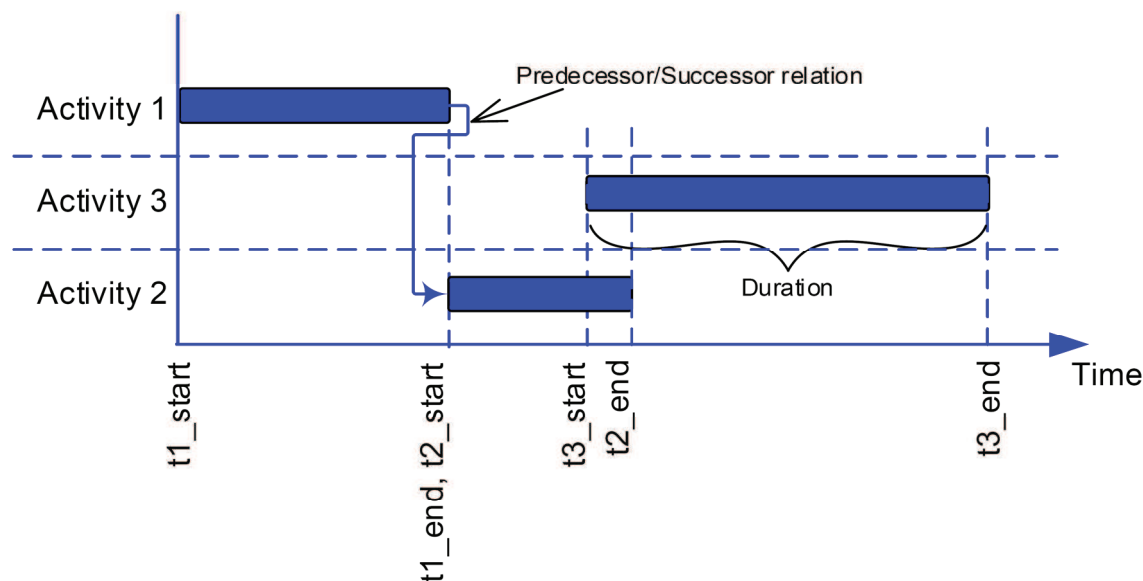


Bild B.2 – Daten in Gantt-Diagrammen

Activity	Aktivität
Predecessor/Successor relation	Vorgänger-Nachfolger-Beziehung
start	Start t1, Start t2, Start t3
end	Ende t1, Ende t2, Ende t3
Time	Zeit
Duration	Dauer
Activity	Aktivität
Predecessor/Successor relation	Vorgänger-Nachfolger-Beziehung
start	Start t1, Start t2, Start t3
end	Ende t1, Ende t2, Ende t3
Time	Zeit
Duration	Dauer

B.3 Vorgangsknotennetze

B.3.1 Allgemeine Angaben

Mithilfe von Vorgangsknotennetzen werden zeitliche und kausale Zusammenhänge von gegenseitig abhängigen Aktivitäten beschrieben und analysiert. Die Ende-Start-Beziehung zwischen zwei Knoten gibt an, dass ein Knoten nur dann starten kann, wenn der vorherige Knoten beendet ist. Vorgangsknotennetze enthalten keine Angaben zur Modellierung von alternativen Pfaden und Zyklen.

B.3.2 Grafische Elemente

In einem Vorgangsknotennetz müssen die folgenden grafischen Elemente verwendet werden (siehe Bild B.3):

- keine oder beliebig viele durch Rechtecke dargestellte Knoten, die jeweils in sieben Felder unterteilt sind;
- keine oder beliebig viele gerichtete Pfeile.

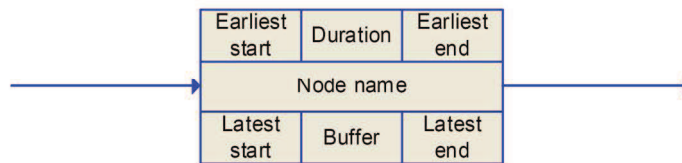


Bild B.3 – Modellelemente von Vorgangsknotennetzen

Earliest start	Frühester Start
Duration	Dauer
Earliest end	Frühestes Ende
Node name	Name des Knotens
Latest start	Spätester Start
Buffer	Puffer
Latest end	Spätestes Ende

B.3.3 Knotenstruktur

Für einen Knoten muss die folgende Struktur verwendet werden:

- Der früheste Start eines Knotens (t_{es}) sollte als Zahl im Feld in der linken oberen Ecke des Knotens eingetragen werden.
- Das früheste Ende eines Knotens (t_{ee}) sollte als Zahl im Feld in der rechten oberen Ecke des Knotens eingetragen werden.
- Die Dauer eines Knotens (t_d) sollte als Zahl im Feld in der Mitte oben des Knotens eingetragen werden. Sie errechnet sich nach der Gleichung $t_d = t_{ee} - t_{es}$ aus dem frühesten Start und dem frühesten Ende des Knotens.
- Der Name des Knotens sollte im Feld in der Mitte des Knotens eingetragen werden.
- Der späteste Start eines Knotens (t_{ls}) sollte als Zahl im Feld in der linken unteren Ecke des Knotens eingetragen werden.
- Das späteste Ende eines Knotens (t_{le}) sollte als Zahl im Feld in der rechten unteren Ecke des Knotens eingetragen werden.
- Der Pufferwert eines Knotens (t_b) sollte als Zahl im Feld in der Mitte unten des Knotens eingetragen werden. Sie errechnet sich nach der Gleichung $t_b = t_{le} - t_{ee}$ aus dem frühesten Ende und dem spätesten Ende des Knotens.

B.3.4 Netzstruktur

Für ein Vorgangsknotennetz muss die folgende Struktur verwendet werden:

- Die Knoten können mithilfe von Pfeilen mit anderen Knoten verbunden werden.
- Ein Pfeil muss an der rechten Seite eines Knotens beginnen und an der linken Seite eines anderen Knotens enden.

B.3.5 Logikinformationen

Vorgangsknotennetze müssen die folgenden Arten von Logikinformationen umfassen:

- Angaben zur Aktivität:
 - Name;
 - frühester Start;

- spätestester Start;
- frühestes Ende;
- spätestes Ende;
- Dauer;
- Puffer.
- Angaben zur Vorgänger-Nachfolger-Beziehung zwischen Aktivitäten.

B.3.6 Logikinformationen in Vorgangsknotennetzen

Im Hinblick auf die Darstellung der Logikinformationen in einem Vorgangsknotennetz gelten die folgenden Festlegungen (siehe Bild B.4):

- Eine Aktivität muss durch genau einen Knoten dargestellt werden.
 - Die Bezeichnung der Aktivität muss durch den Namen des Knotens angegeben werden.
 - Der früheste Start der Aktivität muss durch den frühesten Start des Knotens angegeben werden.
 - Der späteste Start der Aktivität muss durch den spätesten Start des Knotens angegeben werden.
 - Das früheste Ende der Aktivität muss durch das früheste Ende des Knotens angegeben werden.
 - Das späteste Ende der Aktivität muss durch das späteste Ende des Knotens angegeben werden.
 - Die Dauer der Aktivität muss durch die Dauer des Knotens angegeben werden.
 - Die Verzögerung der Aktivität muss durch den Puffer des Knotens angegeben werden.
- Angaben zur Vorgänger-Nachfolger-Beziehung zwischen Aktivitäten müssen durch einen Pfeil angegeben werden.

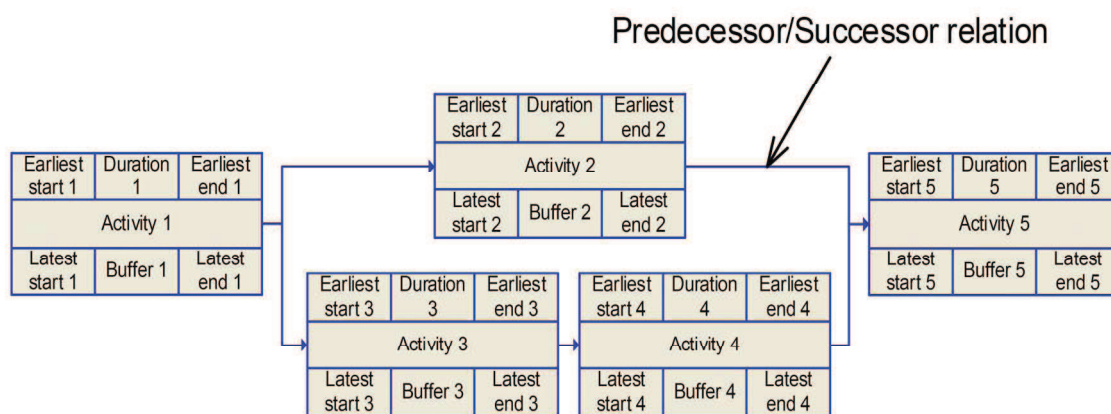


Bild B.4 – Daten in Vorgangsknotennetzen

Earliest start	Frühester Start
Duration	Dauer
Earliest end	Frühestes Ende
Node name	Name des Knotens
Latest start	Spätester Start
Buffer	Puffer
Latest end	Spätestes Ende