

Datenaustausch in der Anlagenplanung mit AutomationML

Rainer Drath
Herausgeber

Datenaustausch in der Anlagenplanung mit AutomationML

Integration von CAEX, PLCopen XML
und COLLADA

 Springer

Herausgeber
Dr.-Ing. Rainer Drath
ABB Forschungszentrum Ladenburg
Wallstadter Str. 59
68526 Ladenburg
Deutschland
rainer.drath@de.abb.com

ISBN 978-3-642-04673-5 e-ISBN 978-3-642-04674-2
DOI 10.1007/978-3-642-04674-2
Springer Heidelberg Dordrecht London New York

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Alle Grafiken/Abbildungen, die unter Verwendung der Software XMLSpy, Copyright 2003-2009 Altova GMBH erstellt wurden, erscheinen mit freundlicher Genehmigung der Altova GmbH.

© Springer-Verlag Berlin Heidelberg 2010

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Einbandentwurf: WMXDesign GmbH, Heidelberg

Gedruckt auf säurefreiem Papier

Springer ist Teil der Fachverlagsgruppe Springer Science+Business Media (www.springer.com)

Vorwort

Vorwort von Prof. Alexander Fay

Ingenieurwissenschaftliche Bücher behandeln im Allgemeinen technische Lösungen oder Methoden, um technische Lösungen zu erstellen. Das vorliegende Buch aber behandelt ein Beschreibungsmittel zum Austausch von Engineering-Daten. Ein ungewöhnliches Thema, aber ein wichtiges und lohnendes, nimmt doch der Anteil des Engineering-Aufwands bei der Konzeption und Realisierung von Maschinen und Anlagen stetig zu. Das Engineering als arbeitsteiliger und zunehmend regional verteilter Arbeitsprozess erfordert Mechanismen zur Sicherstellung einer konsistenten Datenbasis, auf die alle Projektbeteiligten Zugriff haben, idealerweise unabhängig davon, welche Engineering-Werkzeuge sie für die Bearbeitung ihres Anteils am Gesamtprojekt nutzen und in welcher Weise die Daten persistent gespeichert werden. Bislang dominieren noch proprietäre Datenformate bestimmter Engineering-Werkzeuge, aus denen heraus die Engineering-Daten nur verlustbehaftet in andere Werkzeuge übertragen werden können.

An dieser Stelle setzt AutomationML an: es bietet einen definierten, strukturierten Rahmen und die informationstechnischen Mittel, um verschiedene Sichten auf eine zu automatisierende Anlage in einem konsistenten Modell zu beschreiben. Dieses Modell kann gewerke-übergreifend erstellt und genutzt werden. Insbesondere erlaubt es eine werkzeug-unabhängige Beschreibung der Dynamik einer Anlage, sowohl der möglichen Bewegungen aller geometrischen Elemente (Kinematik) als auch der gewollten Bewegungen, d. h. der gewünschten Bewegungsfolgen, die durch die Automatisierungstechnik zu realisieren und sicherzustellen sind. Aus automatisierungstechnischer Sicht sind damit alle Informationen gegeben, um die Automatisierungslösung zielgerichtet zu erstellen. Doch AutomationML nützt nicht darüber hinaus allen anderen am Anlagenprojekt Beteiligten, die sich einen Eindruck vom späteren Anlagenverhalten verschaffen wollen.

Der große Vorteil von AutomationML liegt darin, dass hierin verschiedene bereits in ihrem Anwendungsgebiet bewährte aktuelle Standards miteinander kombiniert werden. Das Rückgrat bildet CAEX (Computer Aided Engineering eXchange) gemäß IEC 62424 als objektorientiertes statisches Anlagenmodell, in das Geometrie- und Kinematik-Beschreibungen nach COLLADA (COLLABorative Design

Activity) und Verhaltensbeschreibungen entsprechend PLCopen XML integriert und miteinander verknüpft werden. So konnte AutomationML durch ein engagiertes Team innerhalb erstaunlich kurzer Zeit zu beachtlicher Reife entwickelt werden. Auf dieser Basis haben die AutomationML-Partner bereits gezeigt, wie Beschreibungen roboterbasierter Fertigungszellen zwischen verschiedenen Software-Werkzeugen ausgetauscht werden können. AutomationML hat das Potential, den Datenaustausch im Anlagenbau signifikant zu vereinfachen, zum Nutzen von Engineering-Dienstleistern, Anlagenbauern, Anlagenbetreibern und Herstellern.

Allen, die wissen wollen, wie sie an dieser Entwicklung partizipieren können, sei dieses Buch sehr empfohlen.

Institut für Automatisierungstechnik
Helmut-Schmidt-Universität Hamburg
Hamburg, im Juli 2009

Prof. Dr.-Ing. Alexander Fay

Vorwort von Anton Hirzle

Die Komplexität und der Kostendruck in der Automatisierungstechnik nehmen ständig zu. Ein wichtiges Mittel zur Beherrschung beider Herausforderungen ist die Standardisierung der Komponenten, Systeme und Prozesse.

Das Engineering einer automatisierten Fertigungsanlage nimmt derzeit rund 40–50% des Investments im steuerungstechnischen Bereich in Anspruch. Deshalb erscheint es besonders lohnend, diesen kostenintensiven Prozess genauer zu untersuchen. Hierbei stellt man schnell fest, dass ein erheblicher Anteil des Aufwands auf das Übertragen der Inhalte von einem Tool zum anderen anfällt.

So kann man beispielsweise als Projektingenieur der Steuerungstechnik nicht ohne weiteres auf die Inhalte der vorgelagerten Planungsphase der „Digitalen Fabrik“ zugreifen, um diese in den folgenden Engineering-Schritten weiter zu detaillieren. Die Übertragung von kinematisierten 3D-Daten ist mit bislang verfügbaren offenen Datenformaten gar nicht möglich – hier müssen die Ingenieure aufwändig Hand anlegen.

Eine Untersuchung der verfügbaren Datenformate im Jahre 2006 zeigte, dass es zu dieser Zeit kein durchgängiges und gleichzeitig frei verfügbares Datenformat gab. Im selben Jahr initiierte Daimler die Bildung einer Arbeitsgruppe, bestehend aus namhaften Firmen der Fertigungsindustrie, die selbst Toolhersteller und Betroffene im Engineering-Prozess waren, mit dem Ziel ein durchgängiges, neutrales Datenformat zu entwickeln. Mittlerweile ist aus dieser einst geschlossenen Arbeitsgruppe ein Industrieverein entstanden, der jedem interessierten Unternehmen die Möglichkeit bietet, sich als Mitglied an der Weiterentwicklung des Formats zu beteiligen.

AutomationML stellt dabei einen ganzheitlichen Ansatz dar. Das Besondere dabei ist die Kombination bewährter Datenformate, die frei zugänglich und etabliert sind – und nicht die Neuerfindung eines Datenformats. Da sich die Toolhersteller

auf die Leistungsmerkmale ihrer Werkzeuge und nicht auf den Datenaustausch konzentrieren wollen, stellte es auch nie ein Problem dar, dass teilweise direkte Wettbewerber gemeinsam an einem Tisch saßen. Alle hatten die gleichen Probleme beim Datentransfer und -handling.

AutomationML soll die enge Bindung von Engineering-Daten an ihre Werkzeuge lösen. Durch AutomationML wird dem Anwender die Möglichkeit eröffnet, stets das Tool zu nutzen, welches für seine Aufgabe am besten geeignet ist, anstatt für ihn fremde, kostenintensive Tools seiner Auftraggeber mittragen zu müssen. Das Ziel ist, keine Inhalte mehr von Tool zu Tool händisch zu übertragen. Vorarbeiten z. B. aus der Digitalen Fabrik sollen mit AutomationML nahtlos genutzt und weiter ausdetailliert werden können.

AutomationML ermöglicht somit Ingenieurbüros, wettbewerbsfähig zu bleiben und weiterhin mit spezialisierten, kleinen aber leistungsfähigen Tools am Markt teilzuhaben. Für den Anlagenbauer bzw. den Anlagenbetreiber ist eine erhebliche Qualitätssteigerung möglich, da im Engineering-Prozess durch Automatismen und den Wegfall des händischen Übertragens der Inhalte nun viel weniger Fehler auftreten. Nicht zuletzt stellt AutomationML die Möglichkeit einer effizienten Darstellung der „Virtuellen Inbetriebnahme“ dar.

AutomationML ist in der Fertigungsindustrie entstanden, aber nicht auf diese beschränkt. So ist der Einsatz in der Prozessindustrie, Luft- und Raumfahrt, Energieerzeugung und -verteilung ebenso denkbar. Alle Interessenten sind herzlich eingeladen, hier mitzuarbeiten.

Das vorliegende Buch bietet sowohl Managern, Entwicklern als auch Anwendern einen Einblick in die Möglichkeiten und den Nutzen von AutomationML.

Senior Manager
Automation Technology and Simulation
Daimler AG

Anton Hirzle

Danksagung

Das vorliegende Buch entstand in der Zeit vom Sommer 2008 bis Sommer 2009 und wäre ohne die tatkräftige Arbeit aller Mitautoren nicht zustande gekommen - an dieser Stelle möchte ich mich herzlich bei ihnen bedanken.

Eine Besonderheit von AutomationML färbt auch auf dieses Buch ab: Wettbewerber sitzen an einem Tisch, entwickeln gemeinsam ein offenes Datenformat und schreiben ein Buch darüber. Dieser Geist ist spürbar: die Schwerpunkte von AutomationML werden aus der Sicht verschiedener Firmen, Branchen und Anwendungen heraus offen beleuchtet. Dies bietet dem Leser eine Fülle von Ansatzpunkten.

Besonders danken möchte ich Michael Lebrecht und Alexander Alonso Garcia, die das Thema AutomationML ins Leben gerufen haben und bis heute leiten. Eine wichtige Rolle spielte dabei Dirk Weidemann, der mit bemerkenswerter Professionalität und viel Humor das AutomationML-Projektmanagement unterstützte. Ich danke zudem ganz besonders Steffen Lips, der mit außergewöhnlichem Engagement in seiner Freizeit das Thema Geometrie und Kinematik bearbeitet, anschaulich niedergeschrieben und zudem mit einem Werkzeug zur Konvertierung von Geometrien und Kinematiken bereichert hat.

Weiterhin möchte ich an dieser Stelle auch allen anderen danken, die durch fachliche Diskussionen einen Beitrag zum Gelingen dieser Arbeit geleistet haben. Besonders zu erwähnen sind dabei Volker Miegel, Sönke Kock, Christian Zeidler, Georg Gutermuth, Anton Hirzle, Christoph Winterhalter und Michael John, die das Thema AutomationML aus den verschiedenen Aspekten heraus zu beleuchten und voranzubringen verstanden und durch eine Vielzahl von Gesprächen und Ideen unterstützt und bereichert haben. Dies gilt ebenso für Josef Prinz von INPRO, der seine Arbeiten zur Bewertung von AutomationML und zur Kombination mit der NE 100 zur Verfügung gestellt hat.

Speziell möchte ich mich bei Prof. Epple und Prof. Fay bedanken, die das drängende Thema des Datenaustausches zwischen den Phasen des Engineering bereits vor Jahren erkannt und das Thema vielfältig und fortwährend befruchtet haben.

Von besonderem Wert für AutomationML sind die Softwarewerkzeuge und zugehörige Dokumentationen, die u. a. von Malte Pirsch, Sebastian Breithor, Steffen Lips und Michala Weisensee erstellt wurden – sie waren die Grundlage für die Entwicklung, Überprüfung und Vermittlung vieler AutomationML-Aspekte. Mit dem

kostenfrei zur Verfügung gestellten AutomationML-Editor, der AutomationML-Engine sowie Konverterwerkzeugen wird nicht nur die Tragfähigkeit von AutomationML untermauert, sondern zudem eine dokumentierte Referenzimplementierung vorgestellt, die Werkzeugherstellern beim Entwickeln von AutomationML-Schnittstellen helfen kann.

Einen ganz persönlichen Dank möchte ich jedoch den Familien und Ehepartnern aller Mitautoren aussprechen, die an vielen Wochenenden und Abenden ihre Unterstützung gegeben und Freiräume für die Arbeit an diesem Buch geschaffen haben. Dies gilt insbesondere für Susanne Brzezinski, die zusätzlich als externe Reviewerin sowohl das Buch als auch den Herausgeber mit vielen wertvollen Hinweisen und ausgezeichnetem Gespür für Sprache sehr bereichert hat.

Dr.-Ing. Rainer Drath, Herausgeber

Inhaltsverzeichnis

1	Einleitung	1
	<i>Dirk Weidemann und Rainer Drath</i>	
1.1	Lesefahrplan: welche Kapitel Sie nicht lesen müssen	1
1.2	Motivation und Problembeschreibung	2
1.2.1	Motivation	2
1.2.2	Problembeschreibung	3
1.2.3	Ansätze	6
1.3	Initiatoren	9
1.4	Ziele von AutomationML	10
1.4.1	Übersicht	10
1.4.2	Offenheit	12
1.4.3	Datenaustausch im Engineering	12
1.4.4	Hoher Abdeckungsgrad	13
1.4.5	Hohe Marktdurchdringung	14
1.4.6	Kombination bewährter Datenformate	14
1.4.7	Erweiterbarkeit und Standardisierung	15
1.4.8	Abgrenzung	15
1.5	Vergleich von Planungsprozessen	16
1.5.1	Ein typischer Planungsprozess in der Automobilindustrie	16
1.5.2	Ein typischer Planungsprozess in der Prozessindustrie	19
1.5.3	Gemeinsamkeiten von Fertigungs- und Prozesstechnik	22
1.5.4	Problematik heterogener CAE-Systeme	23
1.6	AutomationML in a Nutshell: ein Architekturüberblick	25
1.6.1	Architekturanforderungen und Architekturübersicht	25
1.6.2	Beschreibung der Anlagentopologie	28
1.6.3	Geometrie- und Kinematikbeschreibung	30
1.6.4	Beschreibung von Abläufen und Verhalten	31
1.6.5	Schnittstellen- und Rollen-Bibliotheken	33
1.6.6	Erweiterte AutomationML-Konzepte	34
1.7	Anwendungen und Beispiele	35
1.8	Standardisierungsvorhaben	39
1.9	Möglichkeiten der Mitgestaltung	42
	Literatur	43

2 Grundarchitektur: das Objektmodell	45
<i>Rainer Drath und Miriam Schleipen</i>	
2.1 Die Architektur von AutomationML	45
2.2 Ein Wort zur Objektorientierung in der Anlagenplanung	46
2.3 Einführung in CAEX	48
2.3.1 Überblick über wesentliche CAEX-Elemente	48
2.3.2 CAEX-Bibliotheken	49
2.3.3 Die Instanz-Hierarchie	50
2.3.4 Das CAEX-Rollenkonzept	52
2.4 AutomationML-spezifische Festlegungen zu CAEX	54
2.4.1 Drei Wege zum Umgang mit der Instanz-Hierarchie	54
2.4.2 Objektidentifizierung	55
2.4.3 Unterstützung mehrerer Rollen	56
2.5 Beziehungen zwischen CAEX-Objekten	58
2.5.1 Überblick	58
2.5.2 Vater-Kind-Relationen	59
2.5.3 Vererbungsbeziehungen	60
2.5.4 Klassen-Instanz-Relationen	61
2.5.5 Relationen zwischen Instanzen	62
2.6 Referenzierung extern gespeicherter Informationen	64
2.6.1 Referenzierung von COLLADA- und PLCopen-XML-Daten	64
2.6.2 Relationen zwischen extern gespeicherten Informationen	64
2.7 AutomationML-Standardbibliotheken	67
2.7.1 AutomationML-Schnittstellenbibliothek	67
2.7.2 AutomationML-Basis-Rollenbibliothek	68
2.7.3 Fertigungstechnische Rollenbibliothek	69
2.7.4 Leittechnische Rollenbibliothek	69
2.8 Abbildung nutzerdefinierte Daten	70
2.8.1 Übersicht	70
2.8.2 Nutzerdefinierte Attribute	70
2.8.3 Nutzerdefinierte SystemUnit-Klassen	71
2.8.4 Nutzerdefinierte Rollenbibliotheken	72
2.8.5 Fazit	73
2.9 Erweiterte AutomationML-Konzepte	74
2.9.1 Überblick	74
2.9.2 AutomationML Port-Konzept	74
2.9.3 AutomationML Facetten-Konzept	77
2.9.4 AutomationML Gruppen-Konzept	79
2.9.5 Kombination aus Gruppen- und Facetten-Konzept	80
2.9.6 Ressource-Produkt-Prozess-Konzept	83
2.10 Import und Export von AutomationML-Objekten	91
Literatur	94

3 Beschreibung von Geometrie und Kinematik mit COLLADA 95
Steffen Lips

3.1 Übersicht zu COLLADA 1.5 95

3.2 Geometriebeschreibung 96

 3.2.1 Einführung 96

 3.2.2 Aufbau eines COLLADA-Dokuments 97

 3.2.3 Boundary Representation (BREP) 98

 3.2.4 Tessellierte Geometrien 103

 3.2.5 Modellieren von Produktbäumen 107

 3.2.6 Modellieren von Materialien 108

 3.2.7 Modellieren unterschiedlicher Detaillierungsgrade 111

3.3 Kinematikbeschreibung 111

 3.3.1 Anforderung an ein Kinematikbeschreibung 111

 3.3.2 Beschreibung von Gelenken 112

 3.3.3 Kinematische Modelle 113

 3.3.4 Abbildung von Formeln 115

 3.3.5 Artikulierte Systeme 116

 3.3.6 Vereinen von Kinematik und Geometrie 121

 3.3.7 Zusammengesetzte Kinematiken 123

 3.3.8 Verknüpfung von CAEX und COLLADA 124

3.4 Beispiele 128

 3.4.1 BREP: Flansch mit Loch 128

 3.4.2 Dreieckmodell: Flansch mit Loch 130

 3.4.3 Kinematik einer Schraube mit Formel 133

3.5 Zusammenfassung 133

Literatur 134

4 Verhaltensbeschreibung mit PLCopen XML 135
Lorenz Hundt, Arndt Lüder, Rainer Drath und Björn Grimm

4.1 Überblick 135

4.2 Beschreibungsmittel zur Verhaltensmodellierung 139

 4.2.1 Verhaltensinformationen einer Anlagenkomponente 139

 4.2.2 Beschreibungsmittel für Verhalten 140

 4.2.3 Beschreibungsmittel im Engineering-Prozess 141

 4.2.4 Gantt Charts 143

 4.2.5 PERT Charts 144

 4.2.6 Impuls-Diagramme 145

 4.2.7 Sequential Function Charts 147

 4.2.8 Logiknetzwerke 149

 4.2.9 State Charts 150

4.3 PLCopen XML 2.0 153

 4.3.1 Überblick 153

 4.3.2 Das AutomationML addData-Schema 156

4.4	Der Intermediate Modelling Layer IML	160
4.4.1	Motivation	160
4.4.2	IML-Modellelemente	161
4.4.3	IML-Definition und Klassendiagramm	163
4.4.4	Transformation von Gantt Charts nach IML	163
4.4.5	Transformation von PERT Charts nach IML	167
4.4.6	Transformation von Impuls-Diagrammen nach IML	169
4.4.7	Transformation von State Charts nach IML	174
4.4.8	Vergleich der Abbildungsvorschriften nach IML	177
4.4.9	Transformation von IML nach PLCopen XML	179
4.4.10	Vorgehensweise bei der Implementierung von IML	181
4.5	Verriegelungslogik	183
4.5.1	Übersicht	183
4.5.2	Signal- und Komponentengruppen	183
4.5.3	Beschreibung boolescher Verriegelungsbedingungen	186
4.5.4	Erweiterte Verriegelungskonzepte	187
4.6	Integration von Verhaltensbeschreibung in CAEX	188
4.6.1	Referenzierung von PLCopen-XML-Daten	188
4.6.2	Verknüpfung von Verhaltensbeschreibung	189
4.7	Zusammenfassung	191
	Literatur	192
5	Ansatz zur integrierten Prozessbeschreibung	195
	<i>Andreas Keibel</i>	
5.1	Einleitung	195
5.2	Übersicht und Motivation	196
5.2.1	Problembeschreibung	196
5.2.2	Anforderungen an AutomationML	198
5.2.3	Vision	200
5.2.4	Bestehende Datenformate zur Prozessdarstellung	201
5.3	Modellierung von Bearbeitungsprozessen	201
5.3.1	Übersicht	201
5.3.2	Basisanforderungen an eine Prozessbeschreibung	202
5.3.3	Die Eckpfeiler der Prozessbeschreibung	202
5.3.4	Von der Prozessbeschreibung zum Roboter-Code	204
5.4	Datentechnische Inhalte der Objekte	205
5.4.1	Übersicht	205
5.4.2	Modellierung des Bahn-Objektes	206
5.4.3	Modellierung des Werkzeug-Objektes	208
5.4.4	Modellierung des Prozess-Objektes	213
5.5	Beispielmodellierung mit AutomationML	214
5.6	Zusammenfassung	220
	Literatur	220

6	Praktische Anwendung	221
	<i>Rainer Drath, Dirk Weidemann, Steffen Lips,</i>	
	<i>Lorenz Hundt, Arndt Lüder und Miriam Schleipen</i>	
6.1	Überblick	221
6.2	Referenzwerkzeuge	224
6.2.1	Editieren und Visualisieren mit dem AutomationML Editor	224
6.2.2	AutomationML Logic Editor	229
6.2.3	AutomationML Engine	233
6.2.4	COLLADA Tools	239
6.3	Graphic Conditioner Pipeline Framework	239
6.3.1	Motivation	239
6.3.2	Anforderungen	240
6.3.3	Umsetzung des Graphic CPF	243
6.3.4	Fazit	247
6.4	Das Logic CPF	247
6.4.1	Übersicht	247
6.4.2	Rahmenapplikation	249
6.4.3	IML-DOM	250
6.4.4	Plugins	253
6.4.5	Beispiel	254
6.4.6	Erweiterungsmöglichkeiten	256
6.4.7	Aufbau der Pipeline-Konfigurationsdatei	257
6.4.8	Bearbeiten von Pipeline-Konfigurationsdateien	259
6.5	AutomationML-Beispiel „Philipp“	259
6.5.1	Wer oder was ist Philipp	259
6.5.2	Vorgehen zur Abbildung von Philipp mit AutomationML	260
6.5.3	Aufbau der Objektstruktur von Philipp	262
6.5.4	Definition und Implementierung der Objektschnittstellen	264
6.5.5	Integration externer Informationen	266
6.5.6	Verknüpfung der Objektschnittstellen	267
6.5.7	Erstellen von Bibliotheksobjekten	271
6.6	OPC-Konfiguration	273
6.6.1	Übersicht	273
6.6.2	Beispiel	275
6.7	Umgang mit großen CAEX-Dateien	278
6.7.1	Auslagerung von Bibliotheken	279
6.7.2	Aufteilung der Anlagenstruktur	280
6.7.3	Verteilung von Daten in eine Hierarchie von Verzeichnissen	282
6.8	Umgang mit großen COLLADA-Dokumenten	283

6.8.1	Verwendung eines Masterdokuments	283
6.8.2	Auslagerung der verschiedenen Detaillierungsgrade	284
6.8.3	Verteilung der Daten in einer Hierarchie von Verzeichnissen	285
6.9	Workflow-Empfehlungen	287
6.9.1	Übersicht	287
6.9.2	Vom manuellen zum voll automatisierten Datenaustausch	287
6.9.3	Randbedingungen zur Einführung von AutomationML	290
6.9.4	Unidirektionaler Datenaustausch zwischen zwei Werkzeugen	291
6.9.5	Bidirektionaler Datenaustausch zwischen zwei Werkzeugen	292
6.9.6	Sequentieller Workflow	294
6.9.7	Paralleler Workflow	295
6.9.8	AutomationML als zentrale Planungsdatenbasis?	296
6.9.9	Zwischenfazit	297
6.9.10	Empfehlungen zum Umgang mit Rollenbibliotheken	298
6.9.11	Empfehlungen zum Umgang mit SystemUnit- Bibliotheken	299
6.9.12	Zusammenfassung der Empfehlungen	302
	Literatur	304
7	Bewertung und Ausblick	307
	<i>Dirk Weidemann und Rainer Drath</i>	
7.1	Bewertung von AutomationML durch INPRO	307
7.2	Nächste Schritte	309
7.2.1	Schwerpunkte	309
7.2.2	Kommunikation und Gerätebeschreibung	311
7.2.3	Elektroplanung	311
7.2.4	Safety-Aspekte	313
7.2.5	Simulation und virtuelle Inbetriebnahme	313
7.2.6	Compliance-Prüfung	314
7.2.7	Projektierung und Konfiguration von MES mit AutomationML	314
7.3	Zusammenfassung und Ausblick	316
	Literatur	318
	Stichwortverzeichnis	321

Autorenverzeichnis



Dr.-Ing. Rainer Drath

Nach seiner Promotion als Automatisierungsingenieur an der technischen Universität Ilmenau und einem Stipendiumsaufenthalt in Japan ist Rainer Drath seit 2001 als wissenschaftlicher Mitarbeiter im Forschungszentrum der ABB AG in Ladenburg in Projekt- und Linienverantwortung tätig. Dort beschäftigt er sich u. a. mit dem Thema Engineering. Im AutomationML-Konsortium vertritt er die Themen AutomationML-Architektur, CAEX, Anlagentopologie, Bibliotheken sowie das Thema Logik.



Björn Grimm

Björn Grimm studierte Elektrotechnik und Informationstechnik an der Universität Karlsruhe (TH). Seit dem Jahre 2004 ist er bei der Daimler AG mit dem Entwurf innovativer Produktionskonzepte und Technologien sowie der Entwicklung von Engineering-Werkzeugen für die Automatisierungstechnik beschäftigt. Im AutomationML-Konsortium vertritt er das Thema Logik.

**Lorenz Hundt**

Lorenz Hundt ist seit 2006 wissenschaftlicher Mitarbeiter am Center Verteilte Systeme der Otto-von-Guericke Universität Magdeburg. Dort leitet er das europäische Zertifizierungslabor für EtherNet/IP Geräte der ODVA. Seine Forschungsarbeiten betreffen den Bereichen industrielle Kommunikation und die Entwicklung von neuen Konzepten zum Austausch von Engineering-Daten. Im AutomationML-Konsortium vertritt er das Thema Logik.

**Andreas Keibel**

Der 1967 gebürtige Hamburger war während und nach seinem Studium am Institut für Roboterforschung in Dortmund maßgeblich an der Entwicklung des Simulationssystems COSIMIR beteiligt. Nach seiner Promotion als Automatisierungsingenieur 2003 war er zunächst als Projektleiter für die Forschung-& Vorentwicklung der KUKA-Controls tätig und ist seit 2005 bei KUKA-Roboter verantwortlicher Projektleiter für die Applikationsentwicklung im Bereich digitales Engineering. Im AutomationML Konsortium vertritt er das Thema Robotik und Bearbeitungsprozesse.

**Steffen Lips**

Steffen Lips studierte Bauingenieurwesen an der Ruhr-Universität Bochum. Seit 2007 ist er Projektleiter bei der NetAllied Systems GmbH und u. a. verantwortlich für die technische Entwicklung von AutomationML im Bereich COLLADA. Im AutomationML-Konsortium vertritt er das Thema Geometrie und Kinematik.

**Dr.-Ing. habil. Arndt Lüder**

Arndt Lüder leitet das Center Verteilte Systeme der Otto-von-Guericke Universität Magdeburg. Nach dem Studium der Mathematik und Wirtschaftsmathematik promovierte er an der Martin-Luther-Universität Halle-Wittenberg. Seit 2001 arbeitet Dr. Lüder im Center Verteilte Systeme der Otto-von-Guericke Universität Magdeburg, wo er 2007 im Fachgebiet Automation habilitierte. Im AutomationML-Konsortium vertritt er das Thema Logik.

**Miriam Schleipen**

Miriam Schleipen studierte Informatik an der Universität Karlsruhe (TH). Seit 2005 ist sie am Fraunhofer IITB im Geschäftsfeld Leitsysteme beschäftigt. Ihre Tätigkeiten erstrecken sich unter anderem auf die Entwicklung von Konzepten und Methoden zur Verbesserung und Automatisierung des Engineerings von Leitsystemen/ Manufacturing Execution Systems (MES). Ihr wissenschaftlicher Schwerpunkt liegt auf der Adaptivität und Interoperabilität von Teilsystemen in Produktionsanlagen. Im AutomationML-Konsortium vertritt sie das Thema Dachformat und erweiterte Konzepte.

**Dirk Weidemann**

Nach seinem Studium der Mathematik und Informatik entwickelte Dirk Weidemann zunächst CAD-Systeme und technische Applikationen. Schon bald übernahm er Projekt- und Linienverantwortung. Seit 2006 ist er Standortleiter der Zühlke Engineering GmbH in Hannover. Innerhalb der AutomationML ist er seit der Gründung Mitglied des Projektmanagement-Teams.

Abkürzungen

BREP	Boundary Representation
CAD	Computer Aided Design
CAEX	Computer Aided Engineering Exchange
COLLADA	COLLABorative Design Activity
CPF	Conditioner Pipeline Framework
DCC-Tools	Digital Content Creation-Tools
DOM	Dokumenten-Objekt-Modell
EAI	Enterprise Application Integration
FBD	Function Block Diagrams
FPGA	Field Programmable Gate Array
GSM	Global System for Mobile communications
GUID	Global Unique Identifier
HMI	Human Machine Interface
LoD	Level of Detail
MES	Manufacturing Execution System
PLC	Programmable Logic Controller
PLM	Product Lifecycle Management
POU	Program Organization Unit
R&I-Fließbild	Rohrleitungs- und Instrumentierungsfließbild, eine bevorzugte grafische Darstellung einer verfahrenstechnischen Anlage
SASA	Side-Angle-Side-Angle
SASS	Side-Angle-Side-Side
SCADA	Supervisory Control and Data Acquisition
SID	scoped identifier
SIGGRAPH	Special Interest Group on Graphics and Interactive Techniques
SOA	Service-orientierte Architektur
ST	Structured Text
STO	Safe Torque Off
TCP	Tool Center Point
URI	Uniform Resource Identifier
USB	Universal Serial Bus
UTC	Universal Time Coordinated
W3C	World Wide Web Consortium
WPF	Windows Presentation Foundation

Verwendete Markenbegriffe

In diesem Buch wurden eine Reihe von Wortmarken verwendet. Diese befinden sich im Besitz der hier aufgeführten Eigentümer.

Wortmarke	Eigentümer
AutomationML	Siemens AG
Catia	Dassault Systèmes
COLLADA	Sony Computer Entertainment Inc.
Comos, ComosPT	Innotec GmbH
Cosimir	EF Robotertechnik GmbH
Delmia	Dassault Systèmes
Khronos	Khronos Group, Inc.
Microsoft Excel	Microsoft Corp.
Microsoft Project	Microsoft Corp.
Microsoft Visio	Microsoft Corp.
PROLIST	Bayer AG
RobCad	Tecnomatix Technologies Ltd
W3C	Massachusetts Institute of Technology (MIT), European Research Consortium of Informatics and Mathematics (ERCIM) or Keio University (Keio)
XMLSpy	Altova GmbH

Abbildungsverzeichnis

Abb. 1.1	Eine typische Engineering-Prozesskette im Karosseriebau	2
Abb. 1.2	Hohe Komplexität durch zahlreiche Software-Werkzeuge	5
Abb. 1.3	Anzahl von Schnittstellen bei bilateralem Datenaustausch	6
Abb. 1.4	Anzahl der Schnittstellen bei Nutzung offener Datenformate	7
Abb. 1.5	Initiale Ziele von AutomationML	11
Abb. 1.6	Schematischer Planungsprozess im Automobilbau	17
Abb. 1.7	Phase 1 – Fahrzeugentwicklung und digitale Fabrikplanung	17
Abb. 1.8	Phase 3 – Realisierung und Produktion	19
Abb. 1.9	Phasen bei der Planung verfahrenstechnischer Anlagen	20
Abb. 1.10	Bereiche innerhalb der Phase Verfahrensplanung	20
Abb. 1.11	Schnittstelle zwischen Verfahrens- und Leittechnikplanung	22
Abb. 1.12	Von der digitalen Fabrik zur Automatisierungstechnik	24
Abb. 1.13	AutomationML-Architektur	26
Abb. 1.14	Kernkonzepte der AutomationML-Toplevel-Architektur	28
Abb. 1.15	Beispiel für die Anlagenhierarchie einer Fertigungszelle	29
Abb. 1.16	Planar- und Kugelgelenk als Beispiele für Kinematiken	30
Abb. 1.17	Referenzierung von COLLADA-Dokumenten	31
Abb. 1.18	Unterstützte Diagrammtypen für Logikbeschreibungen	32
Abb. 1.19	Referenzierung eines PLCopen-XML-Dokumentes	32
Abb. 1.20	AutomationML-Schnittstellenbibliothek	33
Abb. 1.21	AutomationML-Basis-Rollenbibliothek	33
Abb. 1.22	AutomationML-Rollenbibliothek für die a) Fertigungstechnik und b) Leittechnik	34
Abb. 1.23	Beispiel für Gruppenobjekte	35
Abb. 1.24	Darstellung einer kinematisierten Zelle in verschiedenen Werkzeugen	36
Abb. 1.25	Transformation einer Ablaufdarstellung von Gantt (Excel) über Impulsdiagramme in einen SFC der IEC 61131-3	37
Abb. 1.26	Komponentenkonzepte im Automation Designer (Siemens) und im AutomationML Editor (Zühlke)	38
Abb. 1.27	Geplante IEC-Normenreihe für AutomationML	41
Abb. 1.28	Messepräsentation von AutomationML	43

Abb. 2.1	Toplevel-Architektur von AutomationML	46
Abb. 2.2	Ein Objektbaum in einem modernen Planungswerkzeug	47
Abb. 2.3	Grundlegende CAEX-Elemente	48
Abb. 2.4	Beispiel einer Instanz-Hierarchie	51
Abb. 2.5	CAEX-Modellierung eines Objektbaumes	51
Abb. 2.6	XML-Code des Objektbaumes	52
Abb. 2.7	Das CAEX-Rollenkonzept	53
Abb. 2.8	XML-Beispiel für das Rollenkonzept	53
Abb. 2.9	Klassische Einzel-Rollen-Zuordnung nach IEC 62424	56
Abb. 2.10	Zuordnung mehrerer Rollen mit AutomationML	57
Abb. 2.11	Relationsarten in CAEX	59
Abb. 2.12	XML-Beschreibung von Relationen	60
Abb. 2.13	Relation zwischen zwei Instanzen	63
Abb. 2.14	XML-Dokument für eine Relation zwischen Instanzen	63
Abb. 2.15	Beispiel für Referenzen	64
Abb. 2.16	XML-Text für das Referenzbeispiel	65
Abb. 2.17	Publikation von COLLADA-Schnittstellen in CAEX	65
Abb. 2.18	Veröffentlichte COLLADA-Schnittstellen in CAEX	66
Abb. 2.19	Publikation von PLCopen-XML-Schnittstellen in CAEX	66
Abb. 2.20	Veröffentlichte PLCopen-XML-Schnittstellen in CAEX	66
Abb. 2.21	AutomationML-Standard-Schnittstellenbibliothek	68
Abb. 2.22	AutomationML-Standard-Rollenbibliothek	68
Abb. 2.23	Fertigungstechnische Basis-Rollenbibliothek	69
Abb. 2.24	Leittechnische Basis-Rollenbibliothek	70
Abb. 2.25	Beispiel für nutzerdefinierte Attribute	71
Abb. 2.26	Nutzerdefinierte SystemUnit-Klassenbibliothek	72
Abb. 2.27	Erweiterte AutomationML-Rollenbibliothek	73
Abb. 2.28	Port-Konzept	75
Abb. 2.29	Beispiel für AutomationML-Ports	75
Abb. 2.30	Beispiel für AutomationML-Ports mit CAEX	76
Abb. 2.31	Beispiel für AutomationML-Facetten	78
Abb. 2.32	Beispiel für AutomationML-Facetten mit CAEX	78
Abb. 2.33	Beispiel für Gruppen	79
Abb. 2.34	Umsetzung des Gruppenobjekt-Beispiels mit CAEX	80
Abb. 2.35	Kombination von Gruppen- und Facetten-Konzept	81
Abb. 2.36	XML-Code zur Kombination von Gruppen- und Facetten-Konzept	82
Abb. 2.37	HMI-Faceplate B	83
Abb. 2.38	Generiertes HMI	83
Abb. 2.39	Grundelemente des Dreisichten-Konzeptes	84
Abb. 2.40	Beispielanlage für das PPR-Modell	86
Abb. 2.41	Standard-Rollenklassen für das Drei-Sichtenkonzept	87
Abb. 2.42	Teilbäume des Anwendungsbeispiels	87
Abb. 2.43	Standard-Schnittstellenklasse PPRConnector	87
Abb. 2.44	Verbindungen innerhalb der Beispielanlage	88

Abb. 2.45	Ressourcenzentrierte Sicht auf die Beispielanlage	88
Abb. 2.46	InstanceHierarchy der Beispielanlage in AutomationML	89
Abb. 2.47	Interne Elemente der Beispielanlage	90
Abb. 2.48	Verknüpfungen der Beispielanlage	90
Abb. 2.49	XML-Quelltext der Beispielanlage	92
Abb. 2.50	Import- und Export von AutomationML-Objekten	93
Abb. 3.1	Die 3D-Geometrie einer Anlagenkomponente „Roboter“	97
Abb. 3.2	Allgemeiner Aufbau einer COLLADA-Datei	97
Abb. 3.3	BREP Modell einer Hülse	98
Abb. 3.4	Orientierung von Edges	100
Abb. 3.5	Prinzipieller Aufbau des <brep> Elements	100
Abb. 3.6	Struktur der Begrenzungselemente	102
Abb. 3.7	Tessellierte Darstellung einer Hülse	103
Abb. 3.8	Aufbau einer polygon-basierten Geometrie in COLLADA	104
Abb. 3.9	Das <lines> Element	105
Abb. 3.10	Das <linestrips> Element	105
Abb. 3.11	Das <polygons> Element mit einem Loch	106
Abb. 3.12	Das <polylist> Element	106
Abb. 3.13	Das <triangles> Element	106
Abb. 3.14	Das <tristrips> Element	107
Abb. 3.15	Das <trifans> Element	107
Abb. 3.16	Geometrie einer Komponente „KR150-1“	108
Abb. 3.17	Visuelle Szene der Komponente „Roboter“	109
Abb. 3.18	Definition eines Effekts	110
Abb. 3.19	Definition eines Materials	110
Abb. 3.20	Binden eines Materials	110
Abb. 3.21	Verschiedene Detaillierungsgrade bzw. „Level of Detail“	111
Abb. 3.22	Definition von Gelenken	113
Abb. 3.23	Beispiel eines einfachen kinematischen Modells	114
Abb. 3.24	Ein einfacher Zyklus	115
Abb. 3.25	Kinematische Modell mit Formeln	117
Abb. 3.26	Vorabdefinierte Funktion	118
Abb. 3.27	Verwendung einer Funktion	118
Abb. 3.28	Artikuliertes System mit kinematischen Aspekten	120
Abb. 3.29	Artikuliertes System mit dynamischen Aspekten	121
Abb. 3.30	Kinematische Szene	122
Abb. 3.31	Koppelung zwischen Kinematik und Geometrie	123
Abb. 3.32	Artikuliertes System einer kombinierten Kinematik	124
Abb. 3.33	Referenz eines CAEX Objekts nach COLLADA	125
Abb. 3.34	XML-Quellcode für ein COLLADAInterface in CAEX	126
Abb. 3.35	<extra> Element für eine Komponente	127
Abb. 3.36	Relation zwischen zwei COLLADAInterfaces	127
Abb. 3.37	Verlinkung zwischen Roboter und Greifer	127
Abb. 3.38	Flansch mit Loch	128

Abb. 3.39	Schematische Darstellung der Topologie	129
Abb. 3.40	Grundorientierung der Wires	130
Abb. 3.41	Triangulierte Geometrie	131
Abb. 3.42	Harter und weicher Kantenübergang	131
Abb. 3.43	Schematische Darstellung einer Schraube	131
Abb. 3.44	Definition des kinematischen Modells	132
Abb. 4.1	Beschreibungsmittel für Verhalten in den Phasen des Anlagen-Engineerings	136
Abb. 4.2	Sequencing und Behaviour zur Verhaltensbeschreibung von Automatisierungsgeräten	137
Abb. 4.3	Beispiel für verschiedenen Logik-Informationsarten	140
Abb. 4.4	Nutzung einzelner Beschreibungsmittel während der Planung von Produktionssystemen	141
Abb. 4.5	Beispiel für ein Gantt Chart	143
Abb. 4.6	Beispiel für ein PERT Chart	144
Abb. 4.7	Beispiel für ein Impuls-Diagramm	146
Abb. 4.8	Beispiel eines SFCs nach IEC 61131-3	148
Abb. 4.9	Beispiel für ein Logiknetzwerk	150
Abb. 4.10	Beispiel für ein State Chart	151
Abb. 4.11	Anwendungsfälle zum Datenaustausch mit PLCopen XML	153
Abb. 4.12	PLCopen-XML-Darstellung von „MUX“	155
Abb. 4.13	addData XML-Schema für PLCopen XML	158
Abb. 4.14	Deklaration des AutomationML addData-Schemas in einem PLCopen-XML-Dokument	159
Abb. 4.15	Beispiel für addData-Schema	160
Abb. 4.16	Das Intermediate Modelling Layer IML	161
Abb. 4.17	Ein einfaches IML-Modell	161
Abb. 4.18	Datenmodell des IML	164
Abb. 4.19	Einfaches Beispiel eines IML-Systems	165
Abb. 4.20	Übersetzung eines Gantt-Chart-Balken nach IML	166
Abb. 4.21	Übersetzung einer Vorgänger/Nachfolger-Beziehung nach IML	166
Abb. 4.22	Übersetzung des Zeitverhaltens nach IML	166
Abb. 4.23	Übersetzung des Endschritts nach IML	167
Abb. 4.24	Transformationsbeispiel Gantt Chart nach SFC	168
Abb. 4.25	Übersetzung einer PERT-Chart-Aktivität nach IML	169
Abb. 4.26	Transformationsbeispiel PERT Diagramm nach SFC	170
Abb. 4.27	Übersetzung eines Resource State Flow nach IML	171
Abb. 4.28	Signale in Impuls-Diagrammen	172
Abb. 4.29	Darstellung eines Impuls-Diagramms in IML	173
Abb. 4.30	Transformation eines State Charts nach IML	176
Abb. 4.31	Transformation von IML nach PLCopen XML	180
Abb. 4.32	Transformation eines IML-Schrittes nach PLCopen XML	181
Abb. 4.33	Transformation einer IML Aktivität nach PLCopen XML	182

Abb. 4.34	Beispiel für eine Signal- und eine Komponentengruppe	184
Abb. 4.35	Vereinfachtes CAEX-Modell	185
Abb. 4.36	Prinzip Verknüpfung von Signal- und Komponentengruppen . .	185
Abb. 4.37	Funktionsblocknetzwerk	186
Abb. 4.38	Komplexe Verriegelungsbeschreibungen	187
Abb. 4.39	Verknüpfen von AutomationML-Objekten mit PLCopen-XML-Dokumenten	189
Abb. 4.40	Veröffentlichungsmechanismus von PLCopen-XML-Variablen in CAEX	189
Abb. 4.41	CAEX-Beispiel zur Veröffentlichung von Signalen	190
Abb. 4.42	Verknüpfungsprinzip von Verriegelungs- und Verhaltensbeschreibung	190
Abb. 4.43	Beispielhafte Verknüpfung von Verriegelungs- und Verhaltensbeschreibung	191
Abb. 5.1	Von der Aufgabenbeschreibung zur Bearbeitung	197
Abb. 5.2	Beispiel für einen Bearbeitungsprozess: einfache Schweißnaht	200
Abb. 5.3	Objekte und Relationen der Bearbeitungs- Prozessbeschreibung	203
Abb. 5.4	Ausführungsdreieck	205
Abb. 5.5	Beispiel: Pick & Place	213
Abb. 5.6	Prozessbeschreibungs-Bibliothek	215
Abb. 5.7	AutomationML-Standardschnittstellenklassen	216
Abb. 5.8	Rollenbibliothek zur Prozessbeschreibung	216
Abb. 5.9	Rumpf des CAEX-Objektmodells	217
Abb. 5.10	Beispielprojekt Bahnschweißen	218
Abb. 5.11	Verknüpfungen zwischen den Objekten	219
Abb. 5.12	Prozessgrößenverläufe per SFC definieren	219
Abb. 6.1	Zusammenspiel verschiedener Werkzeuge für AutomationML	223
Abb. 6.2	AutomationML Editor	225
Abb. 6.3	Eine Instanzhierarchie mit ihren Objekten	226
Abb. 6.4	Ergebnisse der Konsistenzprüfung	228
Abb. 6.5	AutomationML Logic Editor	230
Abb. 6.6	Gantt-View im Logic Editor	231
Abb. 6.7	Impulsdiagramm im Logic Editor	232
Abb. 6.8	Ein komplexer SFC	233
Abb. 6.9	Klassenmodell der AutomationML Engine bis CAEXObject . .	234
Abb. 6.10	Klassenmodell der AutomationML Engine ab CAEXObject . .	235
Abb. 6.11	XML-Beispiel für die AutomationML Engine	236
Abb. 6.12	Source-Code-Beispiel für die AutomationML Engine	237
Abb. 6.13	Informationsfragmente einer Komponente	242
Abb. 6.14	Schematische Darstellung einer Konvertierung	242

Abb. 6.15	Kommando zum Konvertieren nach COLLADA	243
Abb. 6.16	Roboter im Tool DELMIA V5	244
Abb. 6.17	Kommando zum Konvertieren von COLLADA	244
Abb. 6.18	Roboter im Tool eM-Engineer	245
Abb. 6.19	Kommando zum Konvertieren von COLLADA nach JT	246
Abb. 6.20	Roboter im Tool JT2GO	246
Abb. 6.21	Prinzip des Logic CPF	248
Abb. 6.22	Plugin-Architektur des Logic CPF	249
Abb. 6.23	Bedienoberfläche der Logic CPF Rahmenapplikation	250
Abb. 6.24	Klassendiagramm des IML-DOM	251
Abb. 6.25	Ablaufdiagramm einer vereinfachten Waschmaschine in Excel	254
Abb. 6.26	GUI zum Aufruf des Logic CPFs	255
Abb. 6.27	PLCopen-XML-Darstellung im Logic Editor	256
Abb. 6.28	Erweiterungsmöglichkeit des Logic CPFs	257
Abb. 6.29	XML-Schema der Pipelinekonfigurationsdatei	258
Abb. 6.30	Beispiel einer Pipeline-Konfigurationsdatei	258
Abb. 6.31	3D-Darstellung von Philipp	259
Abb. 6.32	Hierarchische Struktur des Philipp	260
Abb. 6.33	Verhalten im Philipp-Beispiel	261
Abb. 6.34	Erstellung einer InstanceHierarchy	263
Abb. 6.35	Erstellung eines <code><InternalElement></code>	263
Abb. 6.36	Festlegung von Objekt-Basiseigenschaften	263
Abb. 6.37	Objektstruktur im Philipp-Beispiel	264
Abb. 6.38	Aufbau und Signale einer Sensor-Aktor-Einheit des Philipp	265
Abb. 6.39	Verhalten der Sensor-Aktor-Einheit des Philipp	266
Abb. 6.40	Integration von Schnittstellen im Philipp-Beispiel	268
Abb. 6.41	Schnittstellen im Philipp-Beispiel	268
Abb. 6.42	Ausschnitt der PLCopen-XML-Datei zum Behaviour der Sensor-Aktor-Einheit	269
Abb. 6.43	Integration der PLCopen-XML-Datei der Sensor- Aktor-Einheit	269
Abb. 6.44	Verknüpfung der Schnittstellen der Sensor-Aktor-Einheit	270
Abb. 6.45	Beispiel eines InternalLink-Objekts	270
Abb. 6.46	Erstellung einer neuen SystemUnitClassLib	271
Abb. 6.47	Erstellung der Inhalte einer neuen <code><SystemUnitClass></code>	272
Abb. 6.48	Struktur der Klasse SAE_Class	272
Abb. 6.49	Rollenklasse OPCServer	273
Abb. 6.50	OPCtag-Schnittstelle	274
Abb. 6.51	Verbindungen in der OPC-Beispielanlage	275
Abb. 6.52	InstanceHierarchy der OPC-Beispielanlage	276
Abb. 6.53	XML-Quelltext der OPC-Beispielanlage	277
Abb. 6.54	Beispiel für den Umgang mit großen CAEX-Dateien	278
Abb. 6.55	Beispiele für die Referenzierung externer CAEX-Dateien	278
Abb. 6.56	Verteilung von Daten in mehrere CAEX-Dateien	279

Abb. 6.57	Auslagerung von Bibliotheken in eine separate CAEX-Datei . . .	279
Abb. 6.58	Beispiel zur Referenzierung einer externen Bibliothek	280
Abb. 6.59	Zerlegung einer Instanz-Hierarchie in mehrere Dateien	281
Abb. 6.60	Zentraldokument des Beispielprojektes	281
Abb. 6.61	Externes Dokument Firma01.aml des Beispielprojektes	282
Abb. 6.62	Abbildung der Projektstruktur mittels Dateiordner	282
Abb. 6.63	Aufbau des Masterdokuments	284
Abb. 6.64	Zerlegung eines COLLADA-Dokuments in mehrere Dateien . . .	285
Abb. 6.65	Masterdokument mit LoD-Unterstützung	286
Abb. 6.66	Vorschlag einer Dokumentenstruktur für COLLADA-Dokumente	287
Abb. 6.67	Unidirektionaler Datenaustausch zwischen benachbarten Phasen des Engineering-Workflows	292
Abb. 6.68	Bidirektionaler Datenaustausch mit AutomationML	293
Abb. 6.69	Sequentieller Workflow mit AutomationML	294
Abb. 6.70	Paralleler Workflow mit AutomationML	295
Abb. 6.71	AutomationML als zentrale Datenbasis	297
Abb. 6.72	Austausch von Bibliotheken – Anwendungsfall 1	300
Abb. 6.73	Austausch von Bibliotheken – Anwendungsfall 2	301
Abb. 6.74	Austausch von Bibliotheken – Anwendungsfall 3	302
Abb. 7.1	Systemübergreifender Datenaustausch mit AutomationML und INPRO Rollenbibliothek	308
Abb. 7.2	Vereinfachter Engineering-Prozess für Fertigungsanlagen	310
Abb. 7.3	MES-Datenquelle AutomationML	316

Formelverzeichnis

Formel 3.1	Formeln für ein Parallelogramm	116
Formel 3.2	Schraubenformel	133

Tabellenverzeichnis

Tab. 2.1	Vater-Kind-Beziehung innerhalb einer Klassenbibliothek	61
Tab. 2.2	Beispiel einer Vererbungsrelation zwischen Klassen	61
Tab. 2.3	Beispiel einer Klassen-Instanz-Relation	62
Tab. 2.4	AutomationML-Standard-Schnittstellenklassen	67
Tab. 2.5	AutomationML-Standard-Basis-Rollenbibliothek	69
Tab. 2.6	Nutzerdefinierter AutomationML-Port als Klasse	77
Tab. 3.1	Auflistung der geometrischen Elemente	99
Tab. 3.2	Auflistung der begrenzenden Elemente in der Reihenfolge ihrer Komplexität	99
Tab. 3.3	Liste aller unterstützten Kurvenbeschreibungen	101
Tab. 3.4	Liste aller unterstützten Oberflächenbeschreibungen	101
Tab. 3.5	Liste der Transformationselemente	108
Tab. 3.6	Liste der gängigen Gelenkarten, Abbildungen aus MW (2009)	112
Tab. 3.7	Kinematische Aspekte	119
Tab. 3.8	Dynamische Aspekte	120
Tab. 3.9	Attribute des Attributs Frame	125
Tab. 3.10	Definition Vertizen	129
Tab. 3.11	Definition Edges	129
Tab. 3.12	Definition Wires	130
Tab. 3.13	Definition Face	130
Tab. 4.1	Elemente des AutomationML addData-Schemata	159
Tab. 4.2	Elemente von IML	162
Tab. 4.3	Abbildung der einzelnen Modellelemente auf IML	178
Tab. 4.4	Übertragungsregeln von IML nach SFC	179
Tab. 4.5	Beispiele zur Referenzierung von Logikinformationen aus CAEX	188
Tab. 5.1	Das Bahn-Objekt und seine benötigten Parameter	206
Tab. 5.2	Das Werkzeug-Objekt und seine benötigten Informationen	208
Tab. 5.3	Prozessgrößen und ihre benötigten Informationen	209
Tab. 5.4	Aktionen und ihre benötigten Informationen	209
Tab. 5.5	Modellierung von geometrischen Einschränkungen von Werkzeugen	210

Tab. 5.6	Parameter zur Definition von Freiheitsgraden	210
Tab. 5.7	Beispiel – Prozessgrößen einer Lackierpistole	211
Tab. 5.8	Beispiel – Prozessmethoden einer Lackierpistole	211
Tab. 5.9	Beispiel – Freiheitsgrade einer Lackierpistole	211
Tab. 5.10	Beispiel – Grunddaten für eine Schweißpistole	211
Tab. 5.11	Beispiel – Prozessgrößen für eine Schweißpistole	212
Tab. 5.12	Beispiel – Aktionen und Methoden für eine Schweißpistole	212
Tab. 5.13	Beispieldefinition für eine Schweißpistole	212
Tab. 5.14	Beispielklassen für die Prozessbeschreibung	215
Tab. 6.1	Unterstützte Eingabeformate	240
Tab. 6.2	Unterstützte Ausgabeformate	240
Tab. 6.3	Konvertierungsschrittfolge Teil 1	244
Tab. 6.4	Konvertierungsschrittfolge Teil 2	245
Tab. 6.5	Konvertierungsschrittfolge Teil 3	246
Tab. 6.6	Hauptkomponenten des Logic CPF	248
Tab. 6.7	Methoden des IML-Systems	252
Tab. 6.8	Plugin Schnittstelle ITask	253
Tab. 6.9	Eigenschaften der Klasse PluginOption	254
Tab. 6.10	Schnittstellen für Teilobjekte von Philipp	265
Tab. 6.11	Verhaltensbezogene Schnittstellen der linken Hand	267
Tab. 6.12	Attribute der Rollenklasse OPCServer	274
Tab. 6.13	Attribute der Schnittstellenklasse OPCTag	274
Tab. 6.14	Zuordnung von OPC- und XML-Datentypen	274

Kapitel 1

Einleitung

Dirk Weidemann und Rainer Drath

1.1 Lesefahrplan: welche Kapitel Sie nicht lesen müssen

Dieses Buch gibt erstmalig einen umfassenden Überblick über AutomationML. Es richtet sich zum einen an Ingenieure und Entscheider der Prozess- und Fertigungsindustrie, die sich dem Thema Datenaustausch zwischen Engineering-Werkzeugen widmen möchten. Zum anderen finden Informatiker in diesem Buch viele hilfreiche Details und Beispiele für die Implementierung von AutomationML. Und gerade für Studenten, Lehrbeauftragte und Professoren in Hochschulen und Universitäten ist dieses Buch eine Fundgrube, da AutomationML zur Anwendung und Entwicklung neuer Methoden und Ansätze anregt, die mit heutigen Werkzeugen (noch) nicht realisierbar sind.

AutomationML bietet Stoff für Diplom-, Master- und Doktorarbeiten und kann den Transfer neuer Technologien wie „Semantik Web“ oder „Automation of Automation“ in die Industrie beitragen. Die Literatur zeigt hierzu bereits vielversprechende Ansätze, vgl. beispielsweise Runde et al. (2009), Güttel et al. (2009), Mühlhause et al. (2009), Rimmel u. Drumm (2009), Wollschläger et al. (2009) und Güttel et al. (2008). Welche positiven Effekte eine schrittweise industrielle Einführung neutraler Datenaustauschkonzepte haben kann belegen beispielsweise Kroll u. Still (2009) anhand der NE 100.

- **Kapitel 1** widmet sich der Motivation, den Initiatoren und den Zielen von AutomationML sowie den Möglichkeiten zur Mitgestaltung. Für einen schnellen Einstieg in AutomationML empfiehlt sich der Abschnitt 1.6 „AutomationML in a Nutshell – ein Architekturüberblick“.
- **Kapitel 2** gibt einen Überblick über die Architektur von AutomationML und erläutert, wie Anlagenhierarchien modelliert werden – sie bilden für AutomationML das objektorientierte Grundgerüst der Anlagenplanung.

D. Weidemann (✉)
Zühlke Engineering GmbH, Expo Plaza 3, 30539 Hannover, Deutschland
e-mail: dwe@zuehlke.com