

DIN EN 61499-1

ICS 35.240.50

Einsprüche bis 2012-02-05
Vorgesehen als Ersatz für
DIN EN 61499-1:2006-06

Entwurf

**Verteilte Funktionsbausteine für die Automatisierungstechnik –
Teil 1: Architektur (IEC 65B/799/CDV:2011);
Deutsche Fassung FprEN 61499-1:2011**

Function Blocks –
Part 1: Architecture (IEC 65B/799/CDV:2011);
German version FprEN 61499-1:2011

Blocs Fonctionnels –
Partie 1: Architecture (CEI 65B/799/CDV:2011);
Version allemande FprEN 61499-1:2011

Anwendungswarnvermerk

Dieser Norm-Entwurf mit Erscheinungsdatum 2011-12-05 wird der Öffentlichkeit zur Prüfung und Stellungnahme vorgelegt.

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

Stellungnahmen werden erbeten

- vorzugsweise als Datei per E-Mail an dke@din.de in Form einer Tabelle. Die Vorlage dieser Tabelle kann im Internet unter www.din.de/stellungnahme oder für Stellungnahmen zu Norm-Entwürfen der DKE unter www.dke.de/stellungnahme abgerufen werden;
- oder online im Norm-Entwurfs-Portal des DIN unter www.entwuerfe.din.de, sofern dort wiedergegeben;
- oder in Papierform an die DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE (Hausanschrift: 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 124 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE

Nationales Vorwort

Die Deutsche Fassung des europäischen Dokuments FprEN 61499-1:2011 „Verteilte Funktionsbausteine für die Automatisierungstechnik – Teil 1: Architektur“ (Entwurf in der Umfrage) ist unverändert in diesen Norm-Entwurf übernommen worden.

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. Dem entsprechend ist das internationale Dokument IEC 65B/799/CDV:2011 „Function Blocks – Part 1: Architecture“ unverändert in den Entwurf FprEN 61499-1:2011 übernommen worden.

Das internationale Dokument wurde vom SC 65B „Devices“ der Internationalen Elektrotechnischen Kommission (IEC) erarbeitet.

Bei der Abstimmung zu dem Europäischen Schluss-Entwurf bei CENELEC und dem Internationalen Schluss-Entwurf bei IEC [Final Draft International Standard (FDIS)] sind jeweils nur „JA/NEIN“-Entscheidungen möglich, wobei „NEIN“-Entscheidungen fundiert begründet werden müssen. Dokumente, die bei CENELEC als Europäische Norm angenommen und ratifiziert werden, sind unverändert als Deutsche Normen zu übernehmen.

Für diesen Norm-Entwurf ist das nationale Arbeitsgremium K 931 „Systemaspekte“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE (www.dke.de) zuständig.

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

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

Der Zusammenhang der zitierten Normen mit den entsprechenden Deutschen Normen 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.

Änderungen

Gegenüber DIN EN 61499-1:2006-06 wurden folgende Änderungen vorgenommen:

- a) ECC-Verhalten eindeutig und genauer beschrieben;
- b) lokale (temporäre) Variable in Algorithmen von Basisfunktionsbausteinen aufgenommen;
- c) Netzwerk- und Segment-Typen eingeführt;
- d) Methoden zur Interaktion mit speicherprogrammierbaren Steuerungen mit Programmen nach DIN EN 61131-3 hinzugefügt;
- e) Write- und Read-Management-Operationen vereinfacht.

**Verteilte Funktionsbausteine für die Automatisierungstechnik –
Teil 1: Architektur**

Inhalt

	Seite
1 Anwendungsbereich	6
2 Normative Verweisungen	7
3 Begriffe	8
4 Referenzmodelle	19
4.1 Systemmodell	19
4.2 Gerätemodell	20
4.3 Ressourcenmodell	21
4.4 Anwendungsmodell	22
4.5 Funktionsbausteinmodell	23
4.5.1 Eigenschaften von Funktionsbausteininstanzen	23
4.5.2 Festlegung von Funktionsbausteintypen	24
4.5.3 Ausführungsmodell für Basisfunktionsbausteine	25
4.6 Verteilungsmodell	27
4.7 Managementmodell	27
4.8 Betriebszustandsmodelle	29
5 Festlegung von Funktionsbaustein- und Unteranwendungstypen	29
5.1 Überblick	29
5.2 Basisfunktionsbausteine	31
5.2.1 Typdeklaration	31
5.2.1.1 Allgemeines	31
5.2.1.2 Deklaration der Ereignisschnittstelle	31
5.2.1.3 Deklaration des Algorithmus	32
5.2.1.4 Deklaration der Ausführungssteuerung der Algorithmen	32
5.2.2 Verhalten von Instanzen	33
5.2.2.1 Initialisierung	33
5.2.2.2 Algorithmusaufruf	34
5.2.2.3 Algorithmusausführung	35
5.3 Zusammengesetzte Funktionsbausteine	36
5.3.1 Festlegung von Typen	36
5.3.2 Verhalten von Instanzen	38
5.4 Unteranwendungen	39
5.4.1 Festlegung von Typen	39
5.4.2 Verhalten von Instanzen	40
5.5 Adapterschnittstellen	41
5.5.1 Allgemeine Prinzipien	41
5.5.2 Festlegung von Typen	42

	Seite
5.5.3 Verwendung	43
5.6 Ausnahme und Fehlerbehandlung	45
6 Dienstschnittstellen-Funktionsbausteine	45
6.1 Allgemeine Grundsätze	45
6.1.1 Allgemeines	45
6.1.2 Festlegung von Typen	46
6.1.3 Verhalten von Instanzen	48
6.2 Kommunikationsfunktionsbausteine	50
6.2.1 Festlegung von Typen	50
6.2.2 Verhalten der Instanzen	51
6.3 Managementfunktionsbausteine	52
6.3.1 Anforderungen	52
6.3.2 Festlegung von Typen	53
6.3.3 Verhalten von verwalteten Funktionsbausteinen	56
7 Konfiguration von funktionalen Einheiten und Systemen	58
7.1 Prinzipien der Konfiguration	58
7.2 Funktionale Festlegung von Ressource-, Geräte- und Segmenttypen	58
7.2.1 Funktionale Festlegung von Ressourcentypen	58
7.2.2 Funktionale Festlegung von Gerätetypen	59
7.2.3 Funktionale Festlegung von Segmenttypen	59
7.3 Anforderungen an Konfigurationen	60
7.3.1 Konfigurationen von Systemen	60
7.3.2 Festlegung von Anwendungen	60
7.3.3 Konfiguration von Geräten und Ressourcen	60
7.3.4 Konfiguration von Netzwerksegmenten und Verbindungen	61
Anhang A (normativ) Ereignisfunktionsbausteine	62
Anhang B (normativ) Textuelle Syntax	69
B.1 Technik der Syntaxfestlegung	69
B.2 Festlegung von Funktionsbaustein- und Unteranwendungstypen	70
B.2.1 Festlegung von Funktionsbausteintypen	70
B.2.2 Festlegung von Unteranwendungstypen	73
B.3 Konfigurationselemente	74
B.4 Gemeinsame Elemente	77
B.5 Unterstützende Erzeugungen für Managementbefehle	77
B.6 Datentypen mit Namen	78
B.7 Adapterschnittstellentyp	78
Anhang C (informativ) Objektmodelle	79
C.1 Modell-Notation	79
C.2 Modelle des Entwicklungssystems	79
C.2.1 Bibliothekselemente	80
C.2.2 Deklarationen	81

	Seite
C.2.3 Deklarationen des Funktionsbausteinnetzwerks.....	82
C.2.4 Deklarationen des Funktionsbausteintyps	83
C.3 Modelle für industrielle Leitsysteme	85
Anhang D (informativ) Verhältnis zur IEC 61131-3	87
D.1 Allgemeines	87
D.2 „Einfache“ Funktionsbausteine.....	87
D.3 Ereignisgesteuerte Funktionen und Funktionsbausteine	88
D.4 Normenkonformität mit IEC 61131-3.....	89
D.5 Ausnahmen	89
D.6 Interoperabilität mit speicherprogrammierbaren Steuerungen	89
D.6.1 Einleitung.....	89
D.6.2 Service Connections	89
D.6.3 Funktionsbausteintypen	90
D.6.4 Normerfüllung	95
Anhang E (informativ) Informationsaustausch	97
E.1 Verwendung von Mitteln der Anwendungsschicht	97
E.2 Kommunikationsfunktionsbausteintypen.....	97
E.2.1 Allgemeines	97
E.2.2 Funktionsbausteine für unidirektionale Transaktionen.....	98
E.2.3 Funktionsbausteine für bidirektionale Transaktionen.....	99
E.3 Übertragungssyntaxen	101
E.3.1 Hintergrundinformationen.....	101
E.3.2 Abstrakte Syntaxen IEC 61499-FBDATA	101
E.3.3 Kodierungsregeln	103
E.3.3.1 BASIC-Kodierung.....	103
E.3.3.2 COMPACT-Kodierung.....	103
Anhang F (normativ) Textuelle Festlegungen	105
Anhang G (informativ) Attribute	120
G.1 Allgemeine Prinzipien.....	120
G.2 Definitionen der Attribute.....	120
G.3 Beispiele	121
G.4 Quellen von Attributen.....	122
G.5 Vererbung von Attributen	122
G.6 Deklarationssyntax	122
Bilder	
Bild 1 – Systemmodell	20
Bild 2 – Gerätemodell (Beispiel: Gerät 2 aus Bild 1).....	21
Bild 3 – Ressourcenmodell.....	22
Bild 4 – Anwendungsmodell	23
Bild 5 – Eigenschaften von Funktionsbausteinen.....	24
Bild 6 – Ausführungsmodell.....	26

	Seite
Bild 7 – Zeitverhalten der Ausführung	26
Bild 8 – Verteilung und Managementmodelle	28
Bild 9 – Funktionsbaustein- und Unteranwendungstypen	30
Bild 10 – Deklaration des Basisfunktionsbausteintyps	31
Bild 11 – Beispiel für ein ECC (Plan zur Ausführungssteuerung)	33
Bild 12 – ECC-Zustandsmaschine	34
Bild 13 – Beispiel eines zusammengesetzten Funktionsbausteins PI_REAL	37
Bild 14 – Beispiel eines Basisfunktionsbausteins PID_CALC	38
Bild 15 – Beispiel einer Unteranwendung PI_REAL_APPL	40
Bild 16 – Adapterschnittstellen – Begriffsmodell	41
Bild 17 – Beispiel der Typdeklaration eines Adapters – grafisches Beispiel	42
Bild 18 – Veranschaulichung der Deklarationen von liefernden und annehmenden Funktionsbausteintypen	44
Bild 19 – Veranschaulichung von Adapterverbindungen	45
Bild 20 – Beispiel eines Dienstschnittstellen-Funktionsbausteins	48
Bild 21 – Beispiel eines Zeitablaufdiagramms	49
Bild 22 – Generische Managementfunktionsbausteintypen	53
Bild 23 – Telegrammabläufe für einen nicht erfolgreichen Dienst	53
Bild 24 – Zustandsmaschine eines verwalteten Funktionsbausteins	57
Bild A.1 – Ereignisteilung und -verschmelzung	68
Bild C.1 – Überblick über ein Entwicklungssystem	79
Bild C.2 – Bibliothekselemente	80
Bild C.3 – Deklarationen	81
Bild C.4 – Deklarationen des Funktionsbausteinnetzwerks	83
Bild C.5 – Deklarationen des Funktionsbausteintyps	84
Bild C.6 – Überblick über industrielle Leitsysteme	85
Bild C.7 – Funktionsbausteintypen und Instanzen	86
Bild D.1 – Beispiel eines „einfachen“ Funktionsbausteintyps	87
Bild D.2 – Funktionsbausteintyp READ	91
Bild D.3 – Funktionsbausteintyp UREAD	92
Bild D.4 – Funktionsbausteintyp WRITE	93
Bild D.5 – Funktionsbausteintyp TASK	94
Bild E.1 – Typfestlegung für unidirektionale Transaktionen	98
Bild E.2 – Verbindungseinrichtung für unidirektionale Transaktionen	98
Bild E.3 – Normale unidirektionale Datenübertragung	99
Bild E.4 – Verbindungsauflösung bei unidirektionaler Datenübertragung	99
Bild E.5 – Typfestlegungen für bidirektionale Transaktionen	100
Bild E.6 – Verbindungseinrichtung für bidirektionale Transaktionen	100
Bild E.7 – Bidirektionale Datenübertragung	100
Bild E.8 – Verbindungsauflösung bei bidirektionaler Datenübertragung a) clientinitiiert, b) serverinitiiert, c) systeminitiiert	100

Tabellen

Tabelle 1 – Zustände und Transitionen der ECC-Zustandsmaschine	35
Tabelle 2 – Genormte Eingänge und Ausgänge für Dienstschnittstellen-Funktionsbausteine	46
Tabelle 3 – Bedeutungen der Telegramme.....	50
Tabelle 4 – Bedeutung der Variablen für Kommunikationsfunktionsbausteine.....	51
Tabelle 5 – Semantik der Telegramme für Kommunikationsfunktionsbausteine	52
Tabelle 6 – Eingangswerte und Bedeutungen von CMD	54
Tabelle 7 – Ausgangswerte und Bedeutungen von STATUS	54
Tabelle 8 – Befehlssyntax	55
Tabelle 9 – Bedeutung der Aktionen in Bild 24	58
Tabelle A.1 – Ereignisfunktionsbausteine	62
Tabelle C.1 – Klassenbeschreibungen des Entwicklungssystems	80
Tabelle C.2 – Syntaktische Erzeugungen für Bibliothekselemente.....	81
Tabelle C.3 – Syntaktische Erzeugungen für Deklarationen.....	82
Tabelle C.4 – Klassen in industriellen Leitsystemen	85
Tabelle D.1 – Bedeutung der STATUS-Werte.....	88
Tabelle D.2 – Quellcode des Funktionsbausteutyp READ	91
Tabelle D.3 – Quellcode des Funktionsbausteutyp UREAD	92
Tabelle D.4 – Quellcode des Funktionsbausteutyp WRITE	93
Tabelle D.5 – Quellcode des Funktionsbausteutyp TASK	95
Tabelle D.6 – IEC 61499 Interoperabilitätscharakteristiken	96
Tabelle E.1 – COMPACT-Kodierung von Datentypen mit fester Länge	104
Tabelle G.1 – Elemente der Attributdefinitionen.....	121

1 Anwendungsbereich

Dieser Teil der IEC 61499 definiert eine allgemeine Architektur und gibt Leitlinien für die Anwendung von *Funktionsbausteinen* in verteilten industriellen Leitsystemen. Diese Architektur wird mit Hilfe von Referenzmodellen, textueller Syntax und grafischen Darstellungen dargestellt. Diese Modelle, Darstellungen und Syntax können benutzt werden für:

- die Festlegung und Normierung von *Funktionsbausteintypen*;
- die funktionale Festlegung und Normierung von Systemelementen;
- die implementierungsunabhängige Festlegung, Analyse und Validierung von verteilten industriellen Leitsystemen;
- die *Konfiguration*, die *Implementierung*, den Betrieb und die Instandhaltung von verteilten industriellen Leitsystemen;
- den *Informationsaustausch* zwischen den *Softwarewerkzeugen* für die Ausführung der obigen *Funktionen*.

ANMERKUNG 1 Dieser Teil der IEC 61499 beschränkt weder die funktionalen Eigenschaften von Leitsystemen und ihren Systemelementen oder legt sie nicht fest, ausgenommen sind solche Eigenschaften, die sich durch Verwendung der hier definierten Elemente darstellen. IEC 61499-4 beschreibt den Umfang, indem die in diesem Teil der IEC 61499 definierten Elemente durch die funktionalen Eigenschaften von normgerechten Systemen, Subsystemen und Geräten eingeschränkt werden können.

Diese Spezifikation hat zum Teil den Zweck, Referenzmodelle für die Verwendung von Funktionsbausteinen in anderen Normen zur Verfügung zu stellen, die sich mit der Unterstützung des Systemlebenszyklus beschäftigen, wobei Systemplanung, Entwurf, Implementierung, Validierung, Betrieb und Instandhaltung eingeschlossen sind. Die in diesem Teil der IEC 61499 angegebenen Modelle haben das Ziel, allgemein, anwendungsunabhängig und erweiterbar bezüglich der Definition und der Verwendung von Funktionsbausteinen in anderen Normen, in unterschiedlichen Anwendungen oder Anwendungsgebieten zu sein. Es ist das Ziel, dass Festlegungen, die nach den Regeln geschrieben sind, die in diesem Teil der IEC 61499 vorgegeben sind, prägnant, realisierbar, vollständig, eindeutig und konsistent sind.

ANMERKUNG 2 Die Festlegungen dieses Teils der IEC 61499 allein sind nicht ausreichend, um die Interoperabilität zwischen Geräten verschiedener Lieferanten sicherzustellen. Normen, die diesen Teil der IEC 61499 einhalten, können zusätzliche Bestimmungen festlegen, um eine solche Interoperabilität sicherzustellen.

ANMERKUNG 3 Normen, die diesen Teil der IEC 61499 erfüllen, können zusätzliche Bestimmungen festlegen, um die Leistungsfähigkeit der Managementfunktionen für *System*, *Gerät*, *Ressource* und *Anwendung* zu ermöglichen.

Dieser Standard besteht aus vier Teilen:

- Teil 1 (dieses Dokument) enthält:
 - Allgemeine Anforderungen, die den Anwendungsbereich, die normativen Referenzen, die Definitionen und die Referenzmodelle beinhalten;
 - Regeln für die Deklaration von *Funktionsbausteintypen* und Regeln für das Verhalten von deklarierten *Instanzen* von diesen Typen;
 - Regeln für den Gebrauch von Funktionsbausteinen in der *Konfiguration* verteilter industrieller Leitsysteme;
 - Regeln für den Gebrauch von Funktionsbausteinen zur Erfüllung der Kommunikationsanforderungen verteilter industrieller Leitsysteme;
 - Regeln für den Gebrauch von Funktionsbausteinen beim Management von *Anwendungen*, *Ressourcen* und *Geräten* in verteilten industriellen Leitsystemen.
- Der Teil 2 definiert Anforderungen für Software-Werkzeuge zur Unterstützung der folgenden Systementwicklungsaufgaben:
 - Die Spezifikation von *Funktionsbausteintypen*;

- Die funktionale Spezifikation von *Ressourcentypen* und *Gerätetypen*;
- Die Spezifikation, Analyse und Validierung von verteilten industriellen Leitsystemen;
- Die *Konfiguration*, *Implementierung*, Betrieb und Instandhaltung von verteilten industriellen Leitsystemen;
- Der Austausch von *Informationen* zwischen den *Software-Werkzeugen*.
- Teil 3 (Erläuternde Informationen) wurde aufgrund der zurzeit umfassenden Verfügbarkeit von Übungs- und Lehrmaterial zu IEC 61499 zurückgezogen. Jedoch könnte eine aktualisierte zweite Ausgabe in Zukunft verfasst werden.
- Teil 4 definiert Regeln für die Entwicklung von *Normerfüllungsprofilen*, die jene Funktionen der IEC 61499-1 und IEC 61499-2 festlegen, welche implementiert werden müssen, damit IEC 61499-basierte Systeme, Geräte und Software-Werkzeuge folgende Eigenschaften aufweisen:
 - Interoperabilität von Geräten verschiedener Lieferanten;
 - Portabilität von Software zwischen Software-Werkzeugen von verschiedenen Lieferanten und
 - Konfigurierbarkeit von Geräten verschiedener Hersteller durch Software-Werkzeuge von verschiedenen Lieferanten.

2 Normative Verweisungen

Die folgenden zitierten Dokumente 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 600050-351:1998, *International Electrotechnical Vocabulary (IEV) – Teil 351: Automatic Control (2nd Ed.)*

IEC 559:1989, *Binary floating-point arithmetic for microprocessors*

IEC 61131-3, *Programmable controllers – Part 3: Programming languages (2nd Ed.)*

ISO 2382 (various Parts and dates), *Information processing systems – Vocabulary*

ISO 8601:1988, *Data elements and interchange formats – Information interchange – Representations of dates and times*

ISO/AFNOR, *Dictionary of Computer Science, 1989, ISBN 2-12-4869111-6*

ISO/IEC 7498-1, *Information Technology – Open Systems Interconnection – Basic Reference Model: The Basic Model, 1994*

ISO/IEC 8824-1:2008, *Information technology – Open Systems Interconnection – Specification of Abstract Syntax Notation One (ASN.1)*

ISO/IEC 8825-1:2008, *Information technology – Open Systems Interconnection – Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)*

ISO/IEC 10040:1998, *Information technology – Open Systems Interconnection – Systems management overview*

ISO/IEC 10646-1, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane*

3 Begriffe

Für die Anwendung dieses Schriftstücks gelten sowohl die Begriffe nach *ISO/AFNOR Dictionary of computer science* das *International Electrotechnical Vocabulary* und die folgenden Begriffe:

ANMERKUNG Die in diesem Abschnitt definierten Begriffe sind kursiv gedruckt, wenn sie innerhalb von Definitionen auftreten.

3.1

annehmende Einheit

(en: acceptor)

Funktionsbausteininstanz, die einen *Anschlussadapter* mit einem definierten *Adapterschnittstellentyp* zur Verfügung stellt

3.2

Adapterverbindung

(en: adapter connection)

Verbindung von einem *Steckeradapter* zu einem *Anschlussadapter* desselben *Adapterschnittstellentyps*, die den Fluss von Daten- und Ereignisfluss transportiert, der durch den Adapterschnittstellentyp definiert ist

3.3

Adapterschnittstellentyp

(en: adapter interface type)

Typ, der aus der Definition einer Menge von *Ereigniseingängen*, *Ereignisausgängen*, *Dateneingängen* und *Datenausgängen* besteht, deren *Instanzen* *Steckeradapter* und *Anschlussadapter* sind

3.4

Algorithmus

(en: algorithm)

endliche Menge von wohl definierten Regeln zur Lösung eines Problems mit einer endlichen Anzahl von *Operationen*

3.5

Anwendung

(en: application)

funktionale Software-Einheit, die spezifisch für die Lösung eines Problems in der industriellen Prozessleittechnik ist

ANMERKUNG Eine Anwendung kann auf *Ressourcen* verteilt sein und kann mit anderen Anwendungen kommunizieren.

3.6

Attribut

(en: attribute)

Eigenschaft oder Charakteristik einer *Einheit*, z. B. der Versionsbezeichner der Spezifikation einer *Funktionsbausteininstanz*

3.7

Basisfunktionsbausteutyp

(en: basic function block type)

Funktionsbausteutyp, der nicht in weitere Funktionsbausteine zerlegt werden kann und der einen *Plan der Ausführungssteuerung (ECC)* verwendet, um die *Ausführung* seiner *Algorithmen* zu steuern

3.8

bidirektionale Transaktion

(en: bidirectional transaction)

Transaktion, bei der eine Anforderung und mögliche *Daten* von einer *anfragenden Einheit* zu einer *antwortenden Einheit* übermittelt werden, und bei der eine Antwort und mögliche Daten von der antwortenden zurück zur anfragenden Einheit übermittelt werden

3.9

Zeichen

(en: character)

Element der Menge von Elementen, das zur Darstellung, Organisation oder Steuerung von *Daten* angewendet wird

[ISO/AFNOR:1989]

3.10

Kommunikationsverbindung

(en: communication connection)

Verbindung, die die „Kommunikationsabbildungsfunktion“ von einer oder mehreren *Ressourcen* zur Übermittlung von *Informationen* verwendet

3.11

Kommunikationsfunktionsbaustein

(en: communication function block)

Dienstschnittstellen-Funktionsbaustein, der die *Schnittstelle* zwischen einer *Anwendung* und der „Kommunikationsabbildungsfunktion“ einer *Ressource* darstellt

3.12

Kommunikationsfunktionsbausteintyp

(en: communication function block type)

Funktionsbausteintyp, dessen *Instanzen* *Kommunikationsfunktionsbausteine* sind

3.13

Teilfunktionsbaustein

(en: component function block)

Funktionsbausteininstanz, die zur Spezifikation eines *Algorithmus* eines *zusammengesetzten Funktionsbausteintyps* verwendet wird

ANMERKUNG Ein Teilfunktionsbaustein kann ein *Basistyp*, ein *zusammengesetzter* oder ein *Dienstschnittstellentyp* sein.

3.14

Teilunteranwendung

(en: component subapplication)

Unteranwendungsinstanz, die zur Spezifikation eines *Unteranwendungstyps* verwendet wird

3.15

zusammengesetzter Funktionsbausteintyp

(en: composite function block type)

Funktionsbausteintyp, dessen *Algorithmen* und ihre Bearbeitungssteuerung vollständig mit verknüpften *Teilfunktionsbausteinen*, *Ereignissen* und *Variablen* ausgedrückt werden

3.16

Gleichzeitig

(en: concurrent)

bezogen auf *Algorithmen*, die während einer gemeinsamen Zeitspanne *ausgeführt* werden, in der sie sich abwechselnd gemeinsame *Ressourcen* teilen können

3.17

Konfiguration (eines Systems oder eines Geräts)

(en: configuration (of a system or device))

Auswählen von *Funktionseinheiten*, das Zuweisen ihrer Gerätezuordnung und das Definieren ihrer Verbindungen untereinander

3.18

Konfiguration (eines speicherprogrammierbaren Steuerungssystems)

(en: configuration (of a programmable controller system))

Sprachelement, das sich auf ein *speicherprogrammierbares Steuerungssystem* bezieht, wie es IEC 61131-1 definiert ist

3.19

Konfigurationsparameter

(en: configuration parameter)

Parameter, der sich auf die *Konfiguration* eines *Systems*, eines *Geräts* oder einer *Ressource* bezieht

3.20

Bestätigungstelegramm

(en: confirm primitive)

Telegramm, das eine Interaktion darstellt, bei der eine *Ressource* die Fertigstellung von *Algorithmen* anzeigt, die vorher durch eine Interaktion *aufgerufen* wurden, die durch ein *Anforderungstelegramm* dargestellt wurde

3.21

Verbindung

(en: connection)

Zuordnung, eingerichtet zwischen *funktionalen Einheiten* zur Übermittlung von *Information*

[ISO/AFNOR: 1989]

3.22

kritische Zone

(en: critical region)

Operation oder eine Folge von *Operationen*, die unter der ausschließlichen Kontrolle eines absperrenden Objekts *ausgeführt* werden, das mit den Daten verknüpft ist, mit denen die *Operationen* durchgeführt werden

3.23

Daten

(en: data)

wiederinterpretierbare Darstellung von *Informationen* in einer formalisierten Art, die verwendbar für Kommunikation, Interpretation oder Verarbeitung ist

[ISO 2382-01.01.02]

3.24

Datenverbindung

(en: data connection)

Verbindung zwischen zwei *Funktionsbausteinen* für die Übermittlung von *Daten*

3.25

Dateneingang

(en: data input)

Schnittstelle eines *Funktionsbausteins*, die *Daten* von einer *Datenverbindung* empfängt

3.26

Datenausgang

(en: data output)

Schnittstelle eines *Funktionsbausteins*, die *Daten* an eine *Datenverbindung* liefert

3.27

Datentyp

(en: data type)

Menge von Werten zusammen mit einer Menge von zulässigen *Operationen*

[ISO 2382-15.04.01]

3.28

Deklaration

(en: declaration)

Mechanismus zur Festlegung der Definition einer *Einheit*

ANMERKUNG Eine Deklaration kann die Zuordnung eines Bezeichners zu der *Einheit* einschließen sowie das Zuordnen von *Attributen* wie *Datentypen* und *Algorithmen*.

3.29

Gerät

(en: device)

unabhängige physische *Einheit*, die fähig ist, eine oder mehrere spezifische *Funktionen* in einem speziellen Kontext auszuführen und die durch ihre *Schnittstellen* abgegrenzt ist

ANMERKUNG Ein *speicherprogrammierbares Steuerungssystem*, wie es in IEC 61131-1 definiert ist, ist ein Gerät.

3.30

Gerätemanagementanwendung

(en: device management application)

Anwendung, deren Hauptfunktion das Management von mehreren *Ressourcen* in einem *Gerät* ist

3.31

Einheit

(en: entity)

spezifisches Objekt (Ding), wie Person, Ort, *Prozess*, Objekt, Konzept, Zuordnung oder *Ereignis*

3.32

Ereignis

(en: event)

momentanes Vorkommnis, das ausschlaggebend für den Anstoß der *Ausführung* eines *Algorithmus* ist

ANMERKUNG Bei der Ausführung eines Algorithmus darf eine mit dem Ereignis verbundene *Variable* benutzt werden.

3.33

Ereignisverbindung

(en: event connection)

Verknüpfung zwischen *Funktionsbausteinen* zur Übermittlung von *Ereignissen*

3.34

Ereigniseingang (en: event input)

Schnittstelle eines *Funktionsbausteins*, die *Ereignisse* von einer *Ereignisverbindung* empfängt

3.35

Ereignis Ausgang

(en: event output)

eine *Schnittstelle* eines *Funktionsbausteins*, die *Ereignisse* an eine *Ereignisverbindung* ausgibt

3.36

Ausnahme

(en: exception)

Ereignis, das eine Unterbrechung der normalen *Ausführung* bewirkt

3.37

Ausführung

(en: execution)

Prozess, eine Folge von *Operationen* auszuführen, die durch einen *Algorithmus* festgelegt sind

ANMERKUNG Abhängig von den Regeln, die durch den *Algorithmus* des Funktionsbausteins und den aktuellen Werten der *Variablen* in der Datenstruktur des Funktionsbausteins festgelegt sind, kann sich der Ablauf von auszuführenden *Operationen* von einem *Aufruf* einer *Funktionsbausteininstanz* zum nächsten Aufruf unterscheiden.

3.38

Aktion der Ausführungssteuerung

(en: execution control action – EC action)

Element, das einem *Zustand der Ausführungssteuerung* zugeordnet ist, der festlegt, welcher *Algorithmus* auszuführen ist, welches *Ereignis* auszugeben ist, oder beides.

ANMERKUNG Das Zeitverhalten der Algorithmusausführung und das Ausgeben des Ereignisses sind in 5.2.2. behandelt

3.39

Plan der Ausführungssteuerung ECC

(en: execution control chart – ECC)

grafische oder textuelle Darstellung der kausalen Beziehungen zwischen *Ereignissen* an den *Ereigniseingängen* und *Ereignisausgängen* eines *Funktionsbausteins* und der *Ausführung* der *Algorithmen* des Funktionsbausteins unter Verwendung der *Zustände der Ausführungssteuerung*, der *Transitionen der Ausführungssteuerung* und der *Aktionen der Ausführungssteuerung*

3.40

Anfangszustand der Ausführungssteuerung

(en: execution control initial state EC initial state)

Zustand der Ausführungssteuerung, der bei der Initialisierung eines *Plans der Ausführungssteuerung* aktiv ist

3.41

Zustand der Ausführungssteuerung

(en: execution control state – EC state)

Situation, in der das Verhalten eines *Basisfunktionsbausteins* bezüglich seiner *Variablen* durch die *Algorithmen* definiert ist, die mit dem *Zustand der Ausführungssteuerung* verknüpft sind

3.42

Transition der Ausführungssteuerung

(en: execution control transition – EC transition)

Mittel, mit der die Steuerung von einem vorherigen *Zustand der Ausführungssteuerung* zu einem nachfolgenden übergeht

3.43

Fehler

(en: fault)

nicht normale Bedingung, der eine Verminderung oder den Verlust der Fähigkeit einer *funktionalen Einheit* verursachen kann, eine geforderte *Funktion* auszuführen

3.44

Funktion

(en: function)

spezifischer Zweck einer *Einheit* oder seine charakteristische Aktion

3.45

Funktionsbaustein

(en: function block)

Funktionsbausteininstanz

(en: function block instance)

funktionale *Software-Einheit*, die eine mit einem Namen versehene Kopie einer Datenstruktur und zugeordnete *Operationen* umfasst, die durch einen entsprechenden *Funktionsbausteintyp* festgelegt sind

ANMERKUNG 1 Typische Operationen eines Funktionsbausteins schließen eine Veränderung der Datenwerte in seiner Datenstruktur ein.

ANMERKUNG 2 Die *Funktionsbausteininstanz* und ihr zugehöriger *Funktionsbausteintyp*, die in IEC 61131-3 definiert sind, sind Elemente einer Programmiersprache mit einer anderen Menge von Eigenschaften.

3.46

Netzwerk von Funktionsbausteinen

(en: function block network)

Netzwerk, dessen Knoten *Funktionsbausteine* oder *Unteranwendungen* und ihre *Parameter* sind und dessen Verbindungslinien *Datenverbindungen* und *Ereignisverbindungen* sind

ANMERKUNG Dies ist eine Verallgemeinerung der *Funktionsbausteinsprache*, die in IEC 61131-3 definiert ist.

3.47

Funktionsbausteintyp

(en: function block type)

Typ, dessen *Instanzen* *Funktionsbausteine* sind

ANMERKUNG Zu den Funktionsbausteintypen gehören Basisfunktionsbausteintypen, zusammengesetzte Funktionsbausteintypen und Dienstschnittstellen-Funktionsbausteintypen.

3.48

funktionale Einheit

(en: functional unit)

Einheit aus *Hardware* oder *Software* oder aus beidem, die in der Lage ist, einen vorgegebenen Zweck zu erfüllen

[ISO 2382-01.01.40]

3.49

Hardware

(en: hardware)

physische Einrichtung im Gegensatz zu Programmen, Prozeduren, Regeln und der zugehörigen Dokumentation

[ISO/AFNOR: 1989]

3.50

Bezeichner

(en: identifier)

ein oder mehrere *Zeichen*, die zur Benennung einer *Einheit* verwendet werden

3.51

Implementierung

(en: implementation)

Entwicklungsphase, in der die *Hardware* und *Software* eines *Systems* funktionsfähig wird

3.52

Anzeigetelegramm

(en: indication primitive)

Telegramm, das eine Interaktion darstellt, bei der eine *Ressource* entweder a) anzeigt, dass sie auf eigene Initiative einen *Algorithmus aufgerufen* hat, oder b) anzeigt, dass ein *Algorithmus* von einer gleichartigen *Anwendung* aufgerufen wurde

3.53

Information

(en: information)

Bedeutung, die *Daten* mit Hilfe von Regeln, die für diese Daten gelten, momentan zugewiesen wird

[ISO/AFNOR:1989]

3.54

Eingangsvariable

(en: input variable)

Variable, deren Wert von einem *Dateneingang* geliefert wird und die in einer oder mehreren *Operationen* eines *Funktionsbausteins* verwendet werden kann

ANMERKUNG Ein *Eingangsparameter* eines *Funktionsbausteins*, wie er in IEC 61131-3 definiert ist, ist eine *Eingangsvariable*.

3.55

Instanz

(en: instance)

funktionale Einheit, die eine individuelle, mit einem Namen versehene *Einheit* umfasst, die die *Attribute* eines definierten *Typs* besitzt

3.56

Instanzname

(en: instance name)

Bezeichner, der einer *Instanz* zugeordnet ist und der sie kennzeichnet

3.57

Instanziierung

(en: instantiation)

Bildung einer *Instanz* eines spezifischen *Typs*

3.58

Schnittstelle

(en: interface)

gemeinsame Grenze zwischen zwei *Funktionseinheiten*, die durch funktionelle Merkmale, Signalkenngrößen oder andere geeignete Merkmale festgelegt ist

[IEC 60050-351]

3.59

interne Operationen eines Funktionsbausteins

(en: internal operations of a function block)

Operation, die mit einem *Algorithmus* eines Funktionsbausteins verbunden ist oder mit seiner *Ausführungssteuerung* oder mit den funktionalen Fähigkeiten der zugehörigen *Ressource*

3.60

interne Variable

(en: internal variable)

Variable, deren Wert von einer oder mehreren *Operationen* eines *Funktionsbausteins* benutzt oder verändert wird, die aber nicht von einem *Dateneingang* oder zu einem *Datenausgang* übermittelt wird

3.61

Aufruf

(en: invocation)

Prozess, mit dem die *Ausführung* einer Folge von *Operationen*, die in einem *Algorithmus* festgelegt sind, eingeleitet wird

3.62

Verbindung

(en: link)

Entwurfselement, das eine *Verbindung* zwischen einem *Gerät* und einem *Netzwerksegment* beschreibt

3.63

Literal

(en: literal)

lexikalische Einheit, die direkt einen Wert darstellt

3.64

Managementfunktionsbaustein

(en: management function block)

Funktionsbaustein, dessen primäre *Funktion* die Verwaltung von *Anwendungen* innerhalb einer *Ressource* ist

3.65

Managementressource

(en: management resource)

Ressource, deren primäre *Funktion* die Verwaltung von anderen *Ressourcen* ist

3.66

Abbildung

(en: mapping)

Menge von Eigenschaften oder *Attributen*, die einen definierten Zusammenhang mit den Elementen einer anderen Menge haben

3.67

Nachricht

(en: message)

geordnete Folge von Zeichen zum Zweck der Übermittlung von *Information*

[ISO 2382-16.02.01]

3.68

Nachrichtensenke

(en: message sink)

Teil eines *Kommunikationssystems*, von dem angenommen wird, dass es *Nachrichten* empfängt

[ISO 2382-16.02.03]

3.69

Nachrichtenquelle

(en: message source)

Teil eines *Kommunikationssystems*, von dem angenommen wird, dass die *Nachrichten* dort herkommen

[ISO 2383-16.02.02]

3.70

Modell

(en: model)

Mathematische oder physisch Darstellung eines Systems oder eines Prozesses

3.71

Multitasking

(en: multitasking)

Verarbeitungsart, die die *gleichzeitige Ausführung* von zwei oder mehr *Algorithmen* ermöglicht

3.72

Netzwerk

(en: network)

Gebilde von Knoten und Verbindungszweigen

[ISO 2382-01.01.44]

3.73

Operation

(en: operation)

wohldefinierte Aktion, die eine neue *Einheit* erzeugt, wenn sie auf eine zulässige Kombination von bekannten *Einheiten* angewendet wird

[ISO 2382-02.10.01]

3.74

AusgangsvARIABLE

(en: output variable)

Variable, deren Wert durch eine oder mehrere *Operationen* eines *Funktionsbausteins* festgelegt ist und die an einen *Datenausgang* übermittelt wird

ANMERKUNG Ein *Ausgangsparameter* eines *Funktionsbausteins*, wie er in IEC 61131-3 definiert ist, ist eine *AusgangsvARIABLE*.

3.75

Parameter

(en: parameter)

Variable, die einen konstanten Wert für eine bestimmte *Anwendung* hat und die diese anzeigen kann

[ISO 2383-02.02.04]

3.76

Stecker

(en: plug)

Steckeradapter

(en: plug adapter)

Instanz eines *Adapterschnittstellentyps*, die einen Anfangspunkt für eine *Adapterverbindung* von einem *liefernden Funktionsbaustein* bietet

3.77

liefernde Einheit

(en: provider)

Funktionsbausteininstanz, die einen *Steckeradapter* eines definierten *Adapterschnittstellentyps* bietet

3.78

Anforderungstelegramm

(en: request primitive)

Telegramm, das eine Interaktion darstellt, bei der eine *Anwendung* einen *Algorithmus* aufruft, der von einem *Dienst* zur Verfügung gestellt wird

3.79

Anfordernder

(en: requester)

funktionale Einheit, die eine *Transaktion* mittels eines *Anforderungstelegramms* einleitet

3.80

Ressource

(en: resource)

funktionale Einheit, die eine unabhängige Betriebssteuerung besitzt und die ihren *Anwendungen* verschiedene *Dienste* anbietet, die das Aufrufen und *Ausführen* von *Algorithmen* einschließen

ANMERKUNG 1 Die RESSOURCE, die in IEC 61131-3 definiert ist, ist ein Element einer Programmiersprache, das der oben definierten *Ressource* entspricht.

ANMERKUNG 2 Ein *Gerät* umfasst ein oder mehrere Ressourcen.

3.81

Ressourcenmanagementanwendung

(en: resource management application)

Anwendung, deren primäre *Funktion* die Verwaltung einer einzelnen *Ressource* ist

3.82

Antwortender

(en: responder)

funktionale Einheit, die eine Transaktion mittels eines *Antworttelegramms* zu Ende führt

3.83

Antworttelegramm

(en: response primitive)

Telegramm, das eine Interaktion darstellt, bei der eine *Anwendung* anzeigt, dass sie einen *Algorithmus* vollendet hat, der durch eine Interaktion vorher *aufgerufen* wurde, die von einem *Anzeigetelegramm* dargestellt wird

3.84

abtasten

(en: sample)

abfragen und speichern des momentanen Werts einer *Variablen* für eine spätere Verwendung

3.85

Ausführungsplanfunktion

(en: scheduling function)

Funktion, die *Algorithmen* oder *Operationen* für eine *Ausführung* auswählt und diese Ausführung einleitet und beendet

3.86

Segment

(en: segment)

physikalischer Teil eines *Kommunikationsnetzwerkes*

3.87

Dienst

(en: service)

funktionale Fähigkeit einer *Ressource*, die durch eine Folge von *Telegrammen* modelliert werden kann

3.88

Dienstschnittstellen-Funktionsbaustein

(en: service interface function block)

Funktionsbaustein, der einer *Anwendung* einen oder mehrere *Dienste* anbietet, die auf einer Abbildung von *Telegrammen* auf die *Ereigniseingänge*, *Ereignisausgänge*, *Dateneingänge* und *Datenausgänge* des Funktionsbausteins beruhen

3.89

Telegramm

(en: service primitive)

abstrakte, implementierungsunabhängige Darstellung einer Interaktion zwischen einer *Anwendung* und einer *Ressource*

3.90

Dienstsequenzdiagramm

(en: service sequence diagram)

Diagramm welches eine Sequenz von *Telegrammen* darstellt

3.91

Anschlussadapter

(en: socket adapter)

Instanz eines *Adapterschnittstellentyps*, die einen Endpunkt für eine *Adapterverbindung* zu einem *annehmen-*
den Funktionsbaustein bietet

3.92

Software

(en: software)

geistige Schöpfung, die die Programme, Prozeduren, Regeln, *Konfigurationen* und alle zugehörigen Dokumentationen umfasst, die zum Betrieb eines *Systems* gehören

3.93

Software-Werkzeuge

(en: software tool)

Software, die zur Produktion, Prüfung und Analyse anderer Software verwendet wird

3.94

Unteranwendungsinstanz

(en: subapplication instance)

Instanziierung eines *Unteranwendungstyps* innerhalb einer *Anwendung* oder innerhalb eines Unteranwendungstyps

ANMERKUNG Eine Unteranwendungsinstanz kann auf mehrere *Ressourcen* verteilt sein, d. h. ihre Teilfunktionsbausteine oder der Inhalt ihrer Teilunteranwendungen kann verschiedenen Ressourcen zugeordnet sein.

3.95

Unteranwendungstyp

(en: subapplication type)

funktionale Einheit, deren Rumpf aus miteinander verbundenen *Teilfunktionsbausteinen* und *Teilunteranwendungen* besteht

ANMERKUNG Ein Unteranwendungstyp ermöglicht die Erzeugung von Unterstrukturen von *Anwendungen* in der Form einer selbst-ähnlichen Hierarchie.

3.96

System

(en: system)

Menge miteinander zusammenhängender Elemente, die in einem bestimmten Zusammenhang als Ganzes gesehen und als von ihrer Umgebung abgegrenzt betrachtet werden

[IEC 60050-351]

ANMERKUNG 1 Solche Elemente können sowohl materielle Gegenstände oder Begriffe als auch deren Ergebnisse (z. B. Formen einer Organisation, mathematische Methoden und Programmiersprachen) sein.

ANMERKUNG 2 Das System wird als von der Umgebung und von anderen äußeren Systemen durch eine imaginäre Hüllfläche abgegrenzt gedacht, welche die Verbindungen zwischen diesen Systemen und dem betrachteten System durchschneidet.

3.97

Transaktion

(en: transaction)

Einheit eines Dienstes, bei dem eine Anforderung und eventuell *Daten* von einem *Anfragenden* zu einem *Antwortenden* übermittelt werden und bei dem ebenfalls eine Antwort und eventuell Daten von dem *Antwortenden* zurück zu dem *Anfragenden* übermittelt werden können

3.98

Typ

(en: type)

Software-Element, das die gemeinsamen *Attribute* festlegt, die von allen *Instanzen* des Typs verwendet werden

3.99

Typname

(en: type name)

Bezeichner, der einem *Typ* zugeordnet ist und der ihn kennzeichnet

3.100

unidirektionale Transaktion

(en: unidirectional transaction)

Transaktion, bei der eine Anforderung und eventuell *Daten* von einem *Anfragenden* zu einem *Antwortenden* übermittelt werden und bei der keine Antwort vom *Antwortenden* zurück zum *Anfordernden* übermittelt wird

3.101

Variable

(en: variable)

Software-Einheit, die verschiedene Werte annehmen kann, jeweils ein Wert zu einem Zeitpunkt

ANMERKUNG 1 Die Werte einer Variable sind gewöhnlich auf einen *Datentyp* beschränkt.

ANMERKUNG 2 Variable lassen sich in *Eingangsvariable*, *Ausgangsvariable* und *interne Variable* klassifizieren.

4 Referenzmodelle

4.1 Systemmodell

Für die Anwendungen dieser Spezifikation gestaltet sich ein industrielles Prozessleitsystem, wie in Bild 1 gezeigt, als eine Zusammenstellung von *Geräten*, die miteinander verbundenen sind und die untereinander über ein *Kommunikationsnetzwerk* kommunizieren, das aus *Segmenten* und *Verbindungen* besteht. Geräte sind über Verbindungen mit den Netzwerksegmenten verbunden.

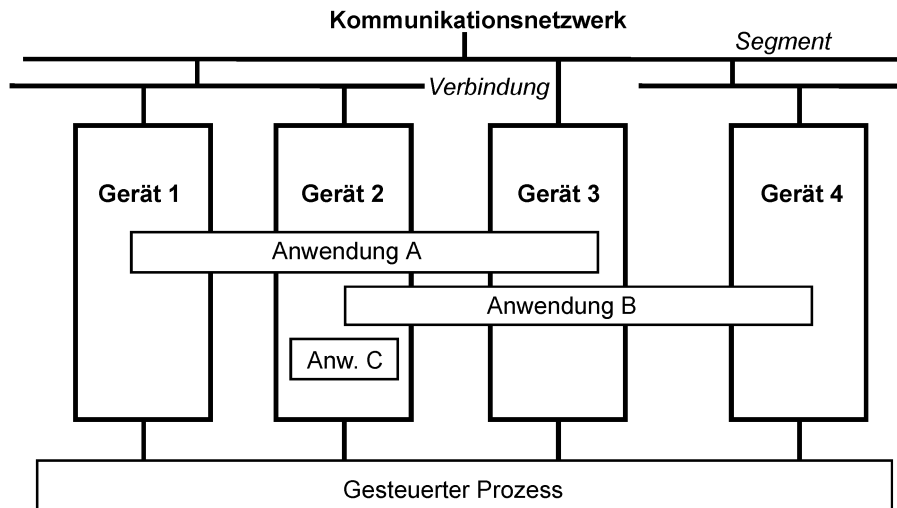


Bild 1 – Systemmodell

ANMERKUNG Der gesteuerte Prozess ist nicht Teil des Leitsystems.

Eine *Funktion*, die vom industriellen Prozessleitsystem ausgeführt wird, ist als eine *Anwendung* modelliert, die in einem einzelnen Gerät liegen kann, wie es für die Anwendung C in Bild 1 gezeigt ist, oder sie kann über mehrere Geräte verteilt sein, wie es für die Anwendungen A und B in Bild 1 gezeigt ist. Zum Beispiel kann eine Anwendung aus einem oder mehreren Regelkreisen bestehen, bei denen das Einlesen der Eingänge von einem Gerät ausgeführt wird, die Regelung von einem anderen und die Ausgangsberechnung von einem dritten.

4.2 Gerätemodell

Wie in Bild 2 veranschaulicht, muss ein *Gerät* mindestens eine *Schnittstelle*, d. h. eine Prozessschnittstelle oder Kommunikationsschnittstelle, enthalten und es kann null oder mehr *Ressourcen* und *Funktionsbaustein-netzwerke* enthalten.

ANMERKUNG 1 Ein Gerät wird als eine *Instanz* eines entsprechenden Gerätetyps betrachtet, wie er in 7.2.2 festgelegt ist.

ANMERKUNG 2 Ein Gerät, das keine Ressourcen enthält, wird als funktional gleichbedeutend mit einer Ressource betrachtet, wie sie in 4.3 definiert ist.

Eine „Prozessschnittstelle“ liefert eine *Abbildung* zwischen dem physikalischen *Prozess* (analoge Messungen, binäre E/A usw.) und den Ressourcen. Die Information, die mit dem physikalischen Prozess ausgetauscht wird, wird zur Ressource in Form von *Daten* oder *Ereignissen* oder beidem dargestellt.

Kommunikationsschnittstellen liefern eine Abbildung zwischen Ressourcen und der Information, die über ein *Kommunikationsnetzwerk* ausgetauscht wird. Die von Kommunikationsschnittstellen gelieferten Dienste können umfassen:

- Darstellung von kommunizierter Information an die Ressource als *Daten* oder *Ereignisse* oder beidem;
- Zusätzliche Dienste zur Unterstützung von Programmierung, *Konfiguration*, Diagnose usw.

ANMERKUNG 3 Kommunikationsverbindungen können entweder direkt einem Gerät oder einer Instanz eines spezifischen Ressourcentyps („Kommunikationsressource“) zugeordnet sein, worauf – abhängig vom Ressourcentyp – dieser Teil der verteilten Anwendung abgebildet sein darf oder nicht.

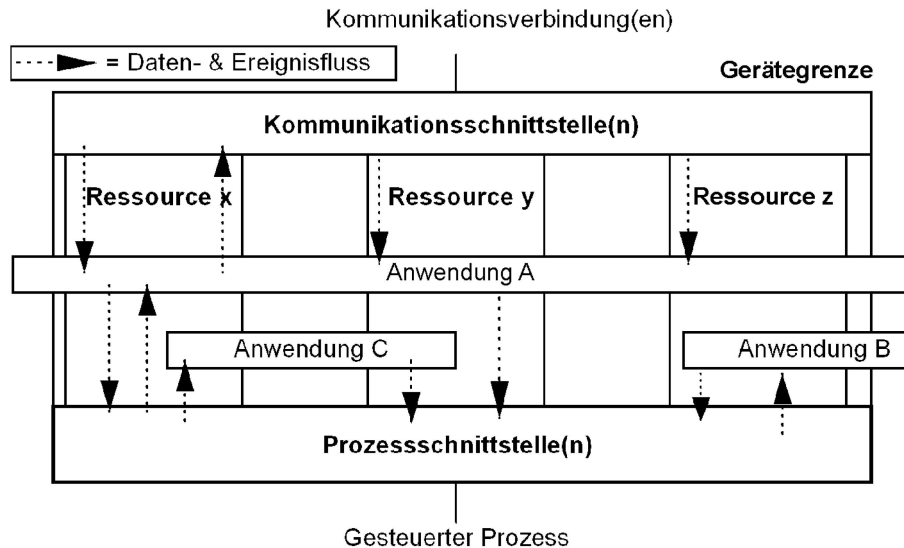


Bild 2 – Gerätemodell
(Beispiel: Gerät 2 aus Bild 1)

4.3 Ressourcenmodell

Für die Anwendung in dieser Spezifikation wird eine *Ressource* als eine *funktionale Einheit* betrachtet, welche eine eigenständige Kontrolle über ihre Funktion besitzt, die in einem *Gerät* enthalten ist, das eine unabhängige Steuerung besitzt. Sie darf neu eingerichtet, konfiguriert, parametrisiert, gestartet, gelöscht usw. werden, ohne dass andere Ressourcen beeinflusst werden.

ANMERKUNG 1 Eine Ressource wird als eine *Instanz* eines entsprechenden Ressourcentyps betrachtet, wie er in 7.2.1 definiert ist.

ANMERKUNG 2 Obwohl eine Ressource eine unabhängige Steuerung besitzt, kann es sein, dass ihre Betriebszustände mit denen anderer Ressourcen zum Zweck der Installation, des Tests usw. koordiniert werden müssen.

Zu den *Funktionen* einer Ressource gehört es, *Daten* und/oder *Ereignisse* vom Prozess und/oder von den *Kommunikationsschnittstellen* entgegenzunehmen, die Daten und/oder Ereignisse zu verarbeiten und die Daten und/oder Ereignisse an den Prozess und/oder die *Kommunikationsschnittstellen* zurückzugeben, so wie es durch die *Anwendungen* festgelegt ist, die die Ressource nutzen.

ANMERKUNG 3 Neben dem Unterstützen der oben aufgezählten Funktionen können spezifische Typen die Fähigkeit darstellen, Schnittstellenfunktionen zu implementieren, wie Prozessschnittstellen oder Kommunikationsdienste der unteren Ebenen über Kommunikationsverbindungen. Abhängig vom Typ dieser Ressourcen können diese Dienste die einzigen sein oder auch nicht die einzigen, die dazu in der Lage sind.

ANMERKUNG 4 Die Betrachtung von weiteren möglichen Aspekten der Ressourcen liegt außerhalb des Anwendungsbereichs dieses Teil der IEC 61499.

Wie in Bild 3 veranschaulicht, ist eine Ressource mit folgenden Teilen modelliert:

- Eine oder mehrere „lokale Anwendungen“ (oder lokale Teile von verteilten Anwendungen). Die Variablen und Ereignisse, die in diesem Teil behandelt werden, sind Eingangs- und Ausgangsvariable und Ereignisse an den Ereigniseingängen und Ereignisaustritten von Funktionsbausteinen, die die Operationen ausführen, die die Anwendung benötigt.
- Einen Teil „Prozessabbildung“, der die Funktion besitzt, eine Abbildung von Daten und Ereignissen zwischen Anwendungen und Prozessschnittstelle(n) durchzuführen. Wie in Bild 3 gezeigt, kann dieses Abbilden durch einen Dienstschnittstellen-Funktionsbaustein modelliert werden, der für diesen Zweck spezialisiert ist.

- Eine „Kommunikationsabbildung“, die die Funktion besitzt, ein Abbilden von Daten und Ereignissen zwischen Anwendungen und Kommunikationsschnittstellen durchzuführen. Wie in Bild 3 gezeigt, kann dieses Abbilden durch Dienstschnittstellen-Funktionsbausteine modelliert werden, die für diesen Zweck spezialisiert sind.
- Eine Funktion Ausführungsplan, die die Ausführung der Funktionsbausteine in den Anwendungen und die Datenübertragung zwischen ihnen bewerkstelligt, entsprechend den zeitlichen und ablaufmäßigen Anforderungen, die bestimmt sind durch: 1) das Auftreten von Ereignissen, 2) die Verbindungen der Funktionsbausteine untereinander und 3) die Information des Ausführungsplans, wie Perioden und Prioritäten.

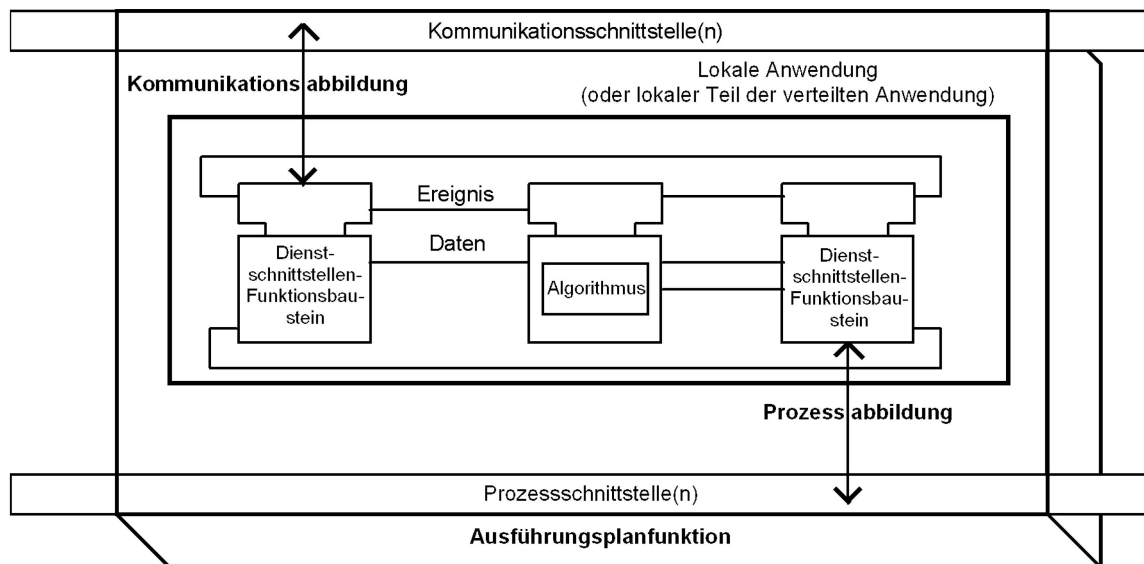


Bild 3 – Ressourcenmodell

ANMERKUNG 1 Dieses Bild dient nur zur Veranschaulichung. Weder die grafische Darstellung noch die Anordnung der Funktionsbausteine ist normativ.

ANMERKUNG 2 Kommunikations- und Prozessschnittstellen können von mehreren Ressourcen gemeinsam genutzt werden.

4.4 Anwendungsmodell

Für die Anwendung dieses Teils der IEC 61499 besteht eine *Anwendung* aus einem *Funktionsbausteinnetzwerk*, dessen Knoten *Funktionsbausteine* oder *Unteranwendungen* und deren *Parameter* sind und dessen Kanten *Datenverbindungen* und *Ereignisverbindungen* sind.

Unteranwendungen sind *Instanzen* von Typen der Unteranwendungen, die wie Anwendungen aus *Funktionsbausteinnetzwerken* bestehen. Namen der Anwendungen, *Namen* der Unteranwendungen und Funktionsbausteininstanzen können somit verwendet werden, um eine Hierarchie von *Bezeichnern* zu erzeugen, die alle *Funktionsbausteininstanzen* in einem System eindeutig identifizieren kann.

Eine Anwendung kann auf verschiedene *Ressourcen* im selben oder in verschiedenen *Geräten* verteilt sein. Eine *Ressource* benutzt die kausalen Beziehungen, die durch die Anwendung festgelegt sind, um die geeigneten Antworten auf *Ereignisse* zu bestimmen, die von den Kommunikations- und Prozessschnittstellen oder anderen Funktionen der Ressource stammen können. Diese Antworten können umfassen:

- Aufruf und Ausführung von Algorithmen;
- Änderung von Variablen;
- Erzeugung von zusätzlichen Ereignissen;
- Interaktionen zwischen Kommunikations- und Prozessschnittstellen.

In Zusammenhang mit diesem Teil der IEC 61499 sind Anwendungen durch *Netzwerke* definiert, die den Ereignis- und Datenfluss zwischen *Instanzen* von *Funktionsbausteinen* oder *Unteranwendungen* festlegen, so wie es in Bild 4 veranschaulicht ist. Der Ereignisfluss bestimmt den Aufruf und die *Ausführung* der *Operationen* durch die zugehörige Ressource. Die Operationen sind durch die Algorithmen der jeweiligen Funktionsbausteine nach den Regeln in 5.2.2 festgelegt.

Normen, Komponenten und Systeme, die diesen Teil der IEC 61499 erfüllen, dürfen alternative Mittel zum Aufrufen der Ausführung festlegen. Solche alternativen Mittel müssen genau festgelegt sein und dabei die Elemente benutzen, die in diesem Teil der IEC 61499 definiert sind.

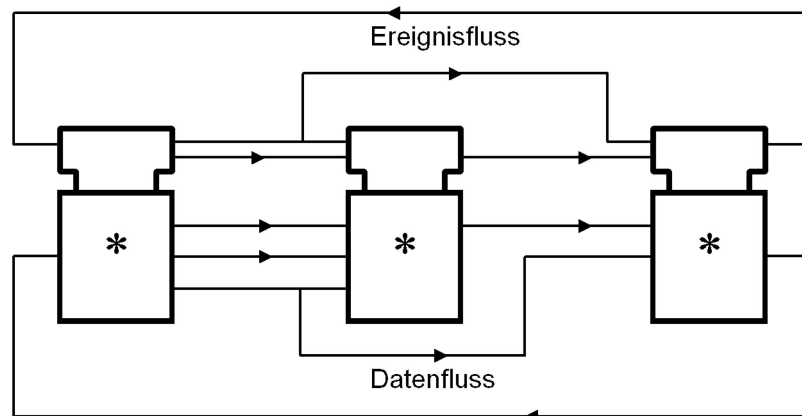


Bild 4 – Anwendungsmodell

ANMERKUNG 1 „*“ stellt die Instanz eines Funktionsbausteins oder einer Unteranwendung dar.

ANMERKUNG 2 Dieses Bild dient nur zur Veranschaulichung. Die grafische Darstellung ist nicht normativ.

4.5 Funktionsbausteinmodell

4.5.1 Eigenschaften von Funktionsbausteininstanzen

Ein *Funktionsbaustein* (*Funktionsbausteininstanz*) ist eine *funktionale Software-Einheit*, die eine individuelle, mit Namen versehene Kopie der Datenstruktur umfasst, die durch einen *Funktionsbausteintyp* festgelegt ist, der von einem Funktionsbausteinaufruf bis zum nächsten bestehen bleibt. Die Eigenschaften der Funktionsbausteininstanzen sind in diesem Unterabschnitt beschrieben und die Festlegungen des Funktionsbausteintyps in 4.5.2.

Eine *Funktionsbausteininstanz* zeigt die folgenden Eigenschaften, die in Bild 5 veranschaulicht sind:

- sein *Typname* und *Instanzname*;
- eine Menge von *Ereigniseingängen*, von denen jeder *Ereignisse* von den *Ereignisverbindungen* empfangen kann, wodurch die Ausführung eines oder mehrerer *Algorithmen* ausgelöst werden kann;
- eine Menge von *Ereignisausgängen*, von denen jeder *Ereignisse* zu den *Ereignisverbindungen* aussenden kann, was abhängig von der Ausführung der *Algorithmen* oder von anderen funktionalen Fähigkeiten der *Ressource* ist, in der sich der Funktionsbaustein befindet;
- eine Menge von *Dateneingängen*, die auf die entsprechenden *Eingangsvariablen* abgebildet sein können;
- eine Menge von *Datenausgängen*, die auf die entsprechenden *Ausgangsvariablen* abgebildet sein können;
- interne Daten, die auf eine Menge von internen Variablen abgebildet werden können;
- funktionale Eigenschaften, die durch die Kombination von internen Daten oder Statusinformation oder beiden mit einer Menge von Algorithmen, funktionalen Fähigkeiten der zugehörigen Ressource oder beiden festgelegt sind. Diese funktionalen Eigenschaften sind durch die Festlegung des Funktionsbausteintyps definiert.

ANMERKUNG 1 Die Information über den internen Zustand darf durch *interne Variablen* oder durch eine interne Darstellung einer Zustandsmaschine für die Ausführungssteuerung dargestellt werden.

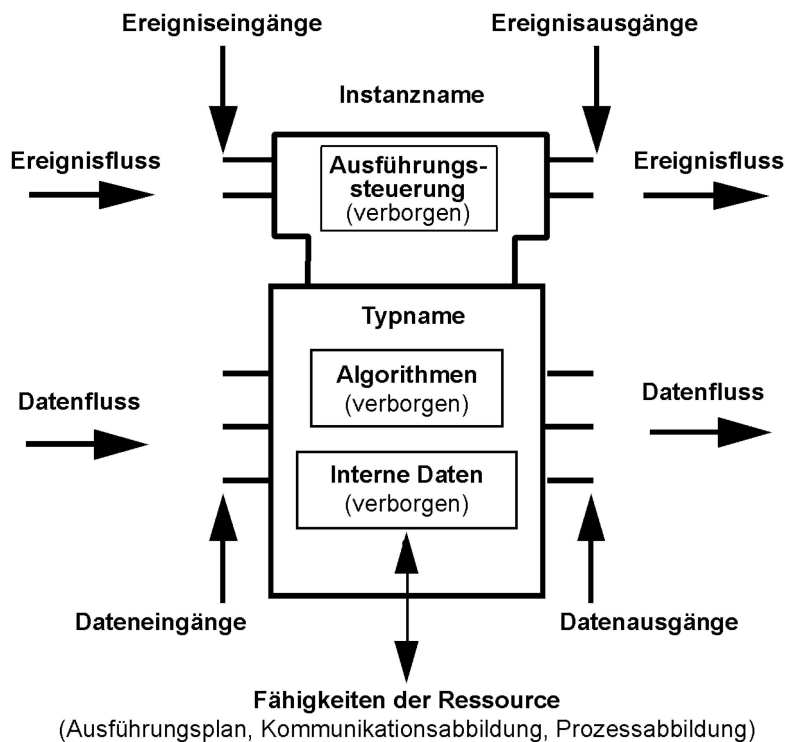


Bild 5 – Eigenschaften von Funktionsbausteinen

ANMERKUNG Dieses Bild dient nur zur Veranschaulichung. Die grafische Darstellung ist nicht normativ.

Die Algorithmen, die in einem Funktionsbaustein enthalten sind, sind im Prinzip von außerhalb des Funktionsbausteins unsichtbar, bis auf das, was der Lieferant des Funktionsbausteins formal oder formlos beschrieben hat. Zusätzlich kann der Funktionsbaustein interne *Variablen* oder Zustandsinformation oder beides enthalten, die zwischen den Aufrufen der Algorithmen des Funktionsbausteins erhalten bleiben, aber nicht durch Datenflussverbindungen von außerhalb des Funktionsbausteins zugänglich sind.

ANMERKUNG 2 Ein Zugriff auf interne Variablen und Zustandsinformation von Funktionsbausteinen darf durch zusätzliche funktionale Fähigkeiten der zugehörigen Ressource zur Verfügung gestellt werden.

In den Abschnitten 5 und 6 sind Mittel zum Festlegen der kausalen Beziehungen zwischen Ereigniseingängen, Ereignisausgängen und der Ausführung von Algorithmen definiert.

4.5.2 Festlegung von Funktionsbausteintypen

Ein *Funktionsbausteintyp* ist ein *Software-Element*, das die Eigenschaften aller *Instanzen* dieses Typs festlegt, dazu gehören:

- sein *Typname*;
- die Anzahl, die Namen, die Typnamen und die Anordnung der *Ereigniseingänge* und *Ereignisausgänge*;
- die Anzahl, die Namen, der *Datentyp* und die Anordnung der Eingangs-, Ausgangs- und internen *Variablen*.

Der Mechanismus für die *Deklaration* dieser Eigenschaften ist in 5.2.1 definiert.

Zusätzlich definiert die Spezifikation des Funktionsbausteins die Funktionalität der *Instanzen* des Typs. Diese Funktionalität kann folgendermaßen ausgedrückt werden:

- Für *Basisfunktionsbausteintypen* stehen in 5.2.1.3 Deklarationsmechanismen für die Festlegung der *Algorithmen* zur Verfügung, die mit den Werten der *Eingangsvariablen*, *Ausgangsvariablen* und *internen Variablen* rechnen, um neue Werte der *Ausgangsvariablen* und *internen Variablen* zu erzeugen. Die Zuordnungen zwischen dem *Aufruf* der Algorithmen und dem Auftreten der *Ereignisse* an den Ereigniseingängen und -ausgängen werden mit Hilfe des *Plans der Ausführungssteuerung* ausgedrückt, wobei die Deklarationsmechanismen benutzt werden, die in 5.2.1.4 definiert sind.
- Die Funktionalität einer *Instanz* eines *zusammengesetzten Funktionsbausteintyps* oder eines *Unteranwendungstyps* wird definiert, indem die Mechanismen nach 5.3.1 bzw. 5.4.1 in der Form von *Datenverbindungen* und *Ereignisverbindungen* zwischen seinen *Teilfunktionsbausteinen* oder *Unteranwendungen* und den Ereignis- und Dateneingängen und -ausgängen des zusammengesetzten Funktionsbausteins oder der Unteranwendung verwendet werden.
- Die Funktionalität einer Instanz eines *Dienstschnittstellen-Funktionsbausteintyps* wird durch ein *Abbilden* der *Telegramme* auf die *Ereigniseingänge*, *Ereignisausgänge*, *Dateneingänge* und *Datenausgänge* beschrieben, wobei die Mechanismen verwendet werden, die in 6.1 definiert sind.
- Es können weitere Mittel wie Text in natürlicher Sprache zur Beschreibung der Funktionalität eines Funktionsbausteintyps verwendet werden; doch die Festlegung solcher Mittel liegt außerhalb des Anwendungsbereichs dieses Standards.

4.5.3 Ausführungsmodel für Basisfunktionsbausteine

Wie in Bild 6 gezeigt ist, wird die *Ausführung* der *Algorithmen* für *Basisfunktionsbausteine* durch den Teil **Ausführungssteuerung** einer *Funktionsbausteininstanz* als Antwort auf Ereignisse an den Ereigniseingängen aufgerufen. Dieser *Aufruf* hat die Form einer Anforderung an die **Ausführungsplanfunktion** der zugehörigen *Ressource*, um die Ausführung der *Operationen* des Algorithmus aufzurufen. Nach Abschluss der Ausführung eines Algorithmus erzeugt die Ausführungssteuerung jeweils null oder mehr Ereignisse an den *Ereignisausgängen*.

ANMERKUNG 1 *Ereignisse* an *Ereigniseingängen* werden durch Verbindung mit den *Ereignisausgängen* anderer Funktionsbausteininstanzen oder derselben Funktionsbausteininstanz zur Verfügung gestellt. Ereignisse an diesen Ereignisausgängen können erzeugt werden durch die Ausführungssteuerung, wie oben beschrieben, die „Kommunikationsabbildung“, die „Prozessabbildung“, das „Aufrufen“ oder andere funktionale Fähigkeiten der *Ressource*.

ANMERKUNG 2 Die Ausführungssteuerung in *zusammengesetzten Funktionsbausteinen* wird mittels des Ereignisses innerhalb des Funktionsbausteinrumpfes erreicht.

Bild 6 stellt den Ablauf von Ereignissen und Algorithmusausführung für den Fall dar, in dem ein einzelner Ereigniseingang, ein einzelner Algorithmus und ein einzelner Ereignisausgang miteinander verknüpft sind. Die relevanten Zeiten sind in diesem Diagramm folgendermaßen definiert:

- t₁: Die relevanten Werte der Eingangsvariablen (d. h. von denen, die mit dem Ereigniseingang durch das Bestimmungszeichen WITH verknüpft sind, das in 5.2.1.2 definiert ist) stehen zur Verfügung.
- t₂: Das Ereignis tritt am Ereigniseingang auf.
- t₃: Die Funktion Ausführungssteuerung fordert die Funktion Ressourcenaufplan auf, einen Algorithmus zur Ausführung aufzurufen.
- t₄: Die Ausführung der Algorithmen beginnt.
- t₅: Der Algorithmus vollendet die Bereitstellung der Werte für die Ausgangsvariablen, die mit dem Ereignisausgang mit Hilfe des Bestimmungszeichen WITH verknüpft sind, das in 5.2.1.2 definiert ist.
- t₆: Der Funktion Ressourcenaufplan wird mitgeteilt, dass die Ausführung des Algorithmus beendet ist.
- t₇: Die Funktion Aufrufplan ruft die Funktion Ausführungssteuerung auf.
- t₈: Die Funktion Ausführungssteuerung signalisiert ein Ereignis am Ereignisausgang.

Wie in Bild 7 gezeigt ist, gibt es in diesem Fall die folgenden signifikanten Zeitverzögerungen, die von Interesse beim Anwendungsentwurf sind:

$$T_{\text{Einstell}} = t_2 - t_1$$

$$T_{\text{Start}} = t_4 - t_2 \text{ (Zeit vom Ereignis am Ereigniseingang bis zum Beginn der Ausführung des Algorithmus)}$$

$$T_{\text{Alg}} = t_6 - t_4 \text{ (Ausführungszeit des Algorithmus)}$$

$$T_{\text{Ende}} = t_8 - t_6 \text{ (Zeit vom Ende der Ausführung des Algorithmus bis zum Ereignis am Ereignisausgang)}$$

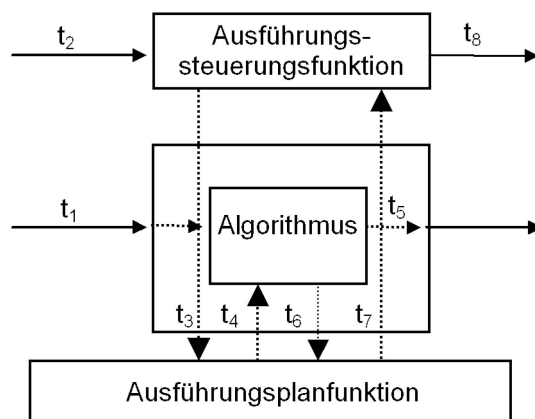


Bild 6 – Ausführungsmodell

ANMERKUNG Dieses Bild dient nur zur Veranschaulichung. Die grafische Darstellung ist nicht normativ.

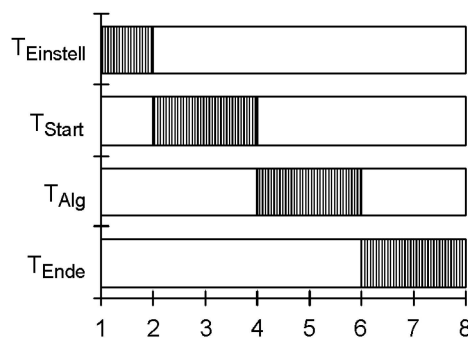


Bild 7 – Zeitverhalten der Ausführung

ANMERKUNG Die Beschriftung der Achse mit 1, 2, ... im Bild oben entspricht den Zeiten $t_1, t_2, \dots$ im Bild 6.

Normative Anforderungen für die Festlegung der Ausführungssteuerung von Funktionsbausteinen im allgemeinen Fall (der den obigen Fall einschließt) sind im Abschnitt 5 definiert.

ANMERKUNG 3 Abhängig vom zu lösenden Problem können verschiedene Anforderungen für die Synchronisierung der Werte der *Eingangsvariablen* mit der *Ausführung von Algorithmen* um die Vorhersagbarkeit der Algorithmusergebnisse zu gewährleisten bestehen. Diese Anforderungen können zum Beispiel enthalten:

- Die Zusicherung, dass die Werte der Variablen, die von einem Algorithmus verwendet werden, während der Ausführung des Algorithmus unverändert bleiben.
- Die Zusicherung, dass die Werte der Variablen, die von einem Algorithmus benutzt werden, den Daten entsprechen, die beim Auftreten des Ereignisses an dem Ereigniseingang, das den Aufruf zur Ausführung des Algorithmus verursacht hat, vorhanden waren.

- Die Zusicherung, dass die Werte der Variablen, die von allen für die Ausführung in einem Funktionsbaustein aufgerufenen Algorithmen benutzt werden, den Daten entsprechen, die beim Auftreten des Ereignisses an dem Ereigniseingang, das den Aufruf zur Ausführung des Algorithmus verursacht hat, vorhanden waren.

ANMERKUNG 4 *Ressourcen* benötigen eventuell ein *Multitasking* zum Aufrufen der *Ausführung* von *Algorithmen*. Die Spezifikation von Attributen, die ein derartiges Aufrufen ermöglichen, ist in Anhang G beschrieben.

4.6 Verteilungsmodell

Wie in Bild 8a) veranschaulicht, kann eine *Anwendung* oder *Unteranwendung* verteilt werden, indem man ihre *Funktionsbausteininstanzen* verschiedenen *Ressourcen* in einem oder mehreren *Geräten* zuteilt. Da die internen Details eines Funktionsbausteins vor einer Anwendung oder Unteranwendung, die ihn benutzt, verborgen sind, muss ein Funktionsbaustein eine atomare Einheit bei der Verteilung darstellen. Das heißt, alle Elemente, die in einem gegebenen Funktionsbaustein enthalten sind, müssen in derselben Ressource enthalten sein.

Die funktionalen Beziehungen zwischen den Funktionsbausteinen einer Anwendung oder Unteranwendung dürfen durch ihre Verteilung nicht beeinflusst werden. Doch wird im Gegensatz zu einer Anwendung oder Unteranwendung, die auf eine einzelne Ressource beschränkt ist, das Zeitverhalten und die Zuverlässigkeit einer Kommunikationsfunktion das Zeitverhalten und die Zuverlässigkeit einer verteilten Anwendung oder Unteranwendung beeinträchtigen.

Es sind die folgenden Abschnitte dieses Teils anzuwenden, wenn Anwendungen oder Unteranwendungen auf mehrere Ressourcen verteilt sind:

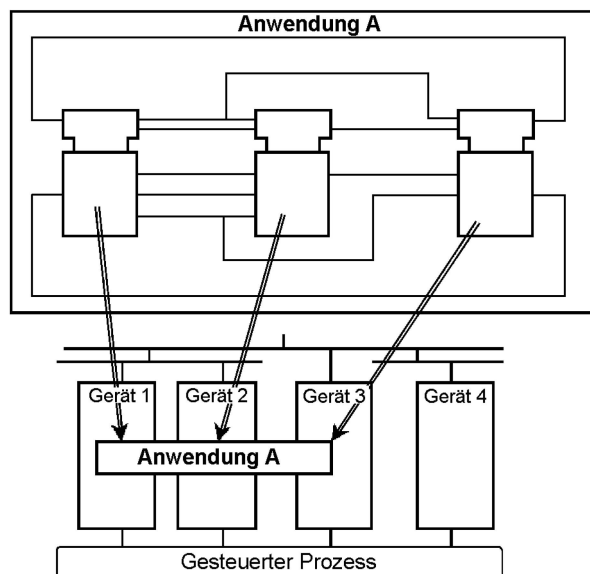
- Abschnitt 6 definiert die Anforderungen an Kommunikationsdienste, um eine Verteilung von Anwendungen oder Unteranwendungen auf mehrere Geräte zu unterstützen.
- Abschnitt 7 definiert die Anforderungen für den Fall, dass mehrere Anwendungen oder Unteranwendungen auf mehrere Ressourcen und Geräte verteilt sind.

4.7 Managementmodell

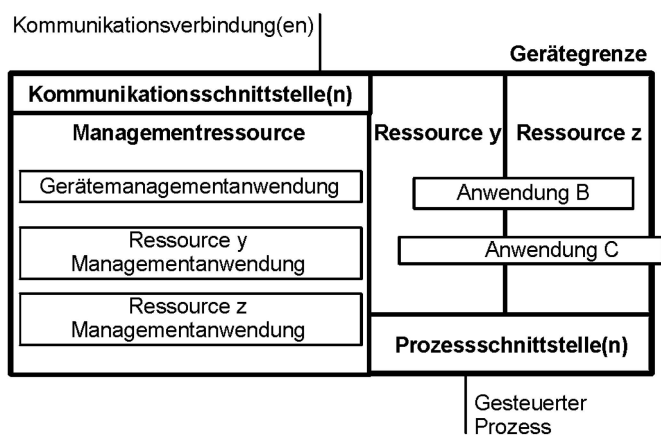
Bild 8b) und 8c) geben eine schematische Darstellung des Managements der *Ressourcen* und *Geräte*. Bild 8b) veranschaulicht einen Fall, bei dem eine *Managementressource* gemeinsame Einrichtungen zum Management von anderen *Ressourcen* innerhalb eines Geräts bietet, während Bild 8c) die Verteilung der Managementdienste auf Ressourcen innerhalb eines Geräts zeigt. *Managementanwendungen* können gebildet werden, indem man implementierungsabhängige *Dienstschnittstellen-Funktionsbausteine* und *Kommunikationsbausteine* verwendet.

ANMERKUNG 1 6.3 definiert *Funktionsbausteintypen für Dienstschnittstellen* zum Management von *Anwendungen* und IEC 61499-2 gibt Beispiele für ihre Verwendung.

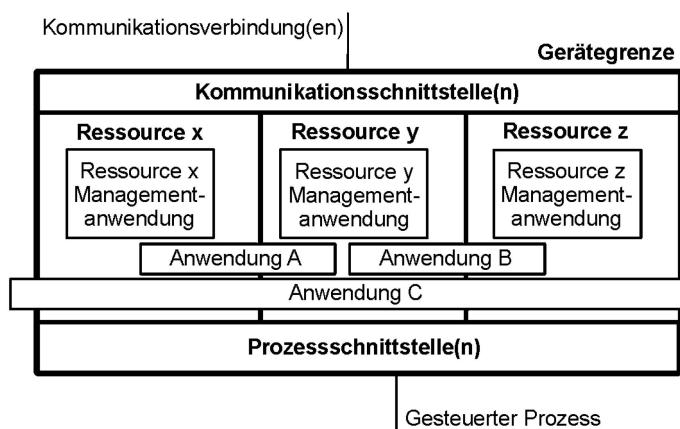
ANMERKUNG 2 *Managementanwendungen* dürfen *Funktionsbausteininstanzen* für Dienstschnittstellen enthalten, die *Instanzen* von *Geräten* oder *Ressourcen* darstellen, um Geräte- oder Ressourcenparameter abzufragen oder zu ändern.



a) Verteilungsmodell



b) Gemeinsames Managementmodell



c) Verteiltes Managementmodell

Bild 8 – Verteilung und Managementmodelle

4.8 Betriebszustandsmodelle

Jedes *System* muss entworfen, in Betrieb genommen, betrieben und instand gehalten werden. Dies wird mit der Vorstellung des System-„Lebenszyklus“ dargestellt. Ein System besteht seinerseits aus mehreren *funktionalen Einheiten*, wie *Geräten*, *Ressourcen* und *Anwendungen*, die jeweils einen eigenen Lebenszyklus haben.

Unterschiedliche Aktionen können durchgeführt werden, um die *funktionalen Einheiten* bei den jeweiligen Schritten im Lebenszyklus zu unterstützen. Um darzustellen, welche Aktion erfolgen kann und um die Integrität der funktionalen Einheiten zu erhalten, müssen Betriebszustände definiert werden, wie z. B. IN BETRIEB, KONFIGURIERBAR, GELADEN, ANGEHALTEN usw.

Jeder Betriebszustand einer funktionalen Einheit legt fest, welche Aktionen zugelassen sind und welches Verhalten dabei erwartet wird.

Ein System darf so organisiert sein, dass gewisse funktionale Einheiten das Recht besitzen oder erlangen können, die Betriebszustände anderer funktionalen Einheiten zu verändern.

Beispiele für die Anwendung von Betriebszuständen sind:

- Eine funktionale Einheit im Zustand LÄUFT, d. h. in Ausführung, kann eventuell außerstande sein, eine Aktion Herunterladen (down load) anzunehmen.
- Eine verteilte funktionale Einheit muss eventuell einen konsistenten Zustand bei allen Komponenten aufrecht erhalten und eine Strategie zur Mitteilung einer Änderung des Betriebszustands an alle Komponenten entwickeln.

Spezielle Betriebszustände für verwaltete *Funktionsbausteininstanzen* sind in 6.3.3 definiert.

5 Festlegung von Funktionsbaustein- und Unteranwendungstypen

5.1 Überblick

Wie in Bild 9 veranschaulicht ist, definiert dieser Abschnitt die Mittel zur Typfestlegung für drei Arten von Bausteinen:

- 5.2 definiert die Mittel zur Festlegung und Bestimmung des Verhaltens von Instanzen der *Basisfunktionsbausteintypen*, wie es in Bild 9a) veranschaulicht ist. Bei diesem Typ von Funktionsbaustein wird die Ausführungssteuerung mit Hilfe eines *Plans zur Ausführungssteuerung* festgelegt und die auszuführenden *Algorithmen* werden so deklariert, wie es in Normen nach IEC 61499-4 festgelegt ist.
- 5.3 definiert die Mittel zur Festlegung von *zusammengesetzten Funktionsbausteintypen*, wie es in Bild 9b) veranschaulicht ist. Bei diesem Typ von Funktionsbaustein werden die Algorithmen und ihre Ausführungssteuerung durch Ereignis- und Datenverbindungen in einem oder mehreren *Funktionsbausteinnetzwerken* festgelegt.
- 5.4 definiert die Mittel zur Festlegung von *Unteranwendungstypen*, wie es in Bild 9c) veranschaulicht wird. Bei diesem Typ von Baustein werden die Algorithmen und ihre Ausführungssteuerung wie bei den zusammengesetzten Funktionsbausteinen festgelegt, jedoch mit der speziellen Eigenschaft, dass die *zusammengesetzten Funktionsbausteine* der Unteranwendungen auf verschiedene *Ressourcen* verteilt sein dürfen. Unteranwendungen können geschachtelt sein, so dass der Rumpf einer Unteranwendung wiederum *Teilunteranwendungen* enthalten kann.

Es dürfen auch andere Mittel zur Beschreibung des Verhaltens von Instanzen eines Funktionsbausteintyps verwendet werden. Die Festlegung solcher Mittel ist nicht im Anwendungsbereich dieses Standards; daher ist es erforderlich, wenn solche Mittel verwendet werden, dass eine eindeutige *Abbildung* zwischen ihren Begriffen und Konzepten und den entsprechenden Begriffen und Konzepten dieses Standards angegeben werden muss.

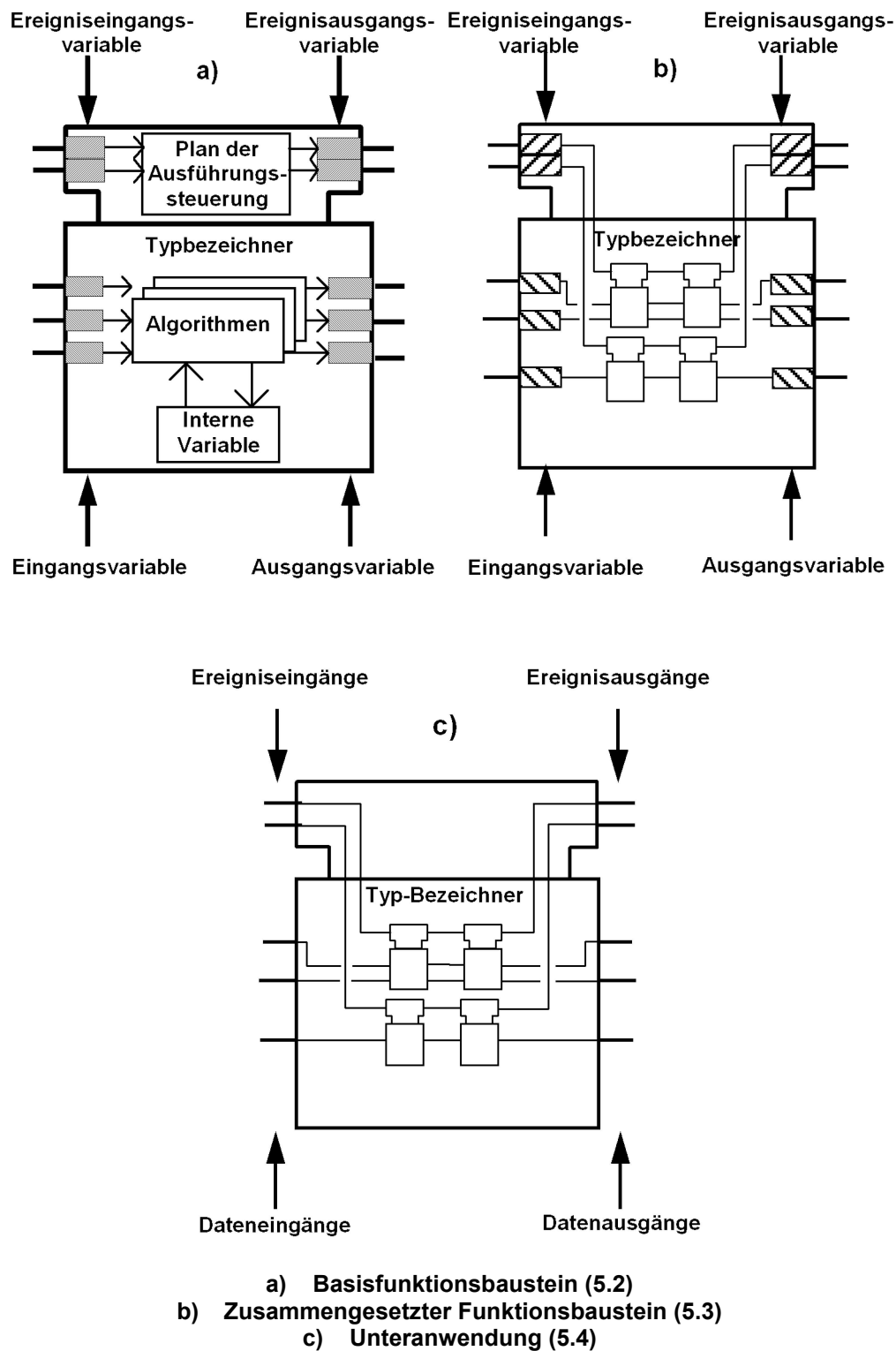


Bild 9 – Funktionsbaustein- und Unteranwendungstypen

ANMERKUNG Dieses Bild dient nur zur Veranschaulichung. Die grafische Darstellung ist nicht normativ.

5.2 Basisfunktionsbausteine

5.2.1 Typdeklaration

5.2.1.1 Allgemeines

Ein *Basisfunktionsbaustein* verwendet einen Plan zur *Ausführungssteuerung*, um die Ausführung seiner *Algorithmen* zu steuern.

Wie in Bild 10 veranschaulicht, kann ein *Basisfunktionsbaustein* entweder textuell entsprechend der Syntax deklariert werden, die in Anhang B.2 festgelegt ist, oder grafisch entsprechend den nachfolgenden Regeln:

- Der Typname des Funktionsbausteins muss oben in der Mitte des rechteckigen Hauptblocks gezeigt werden.
- Die Namen- und *Typdeklarationen* von *Eingangsvariablen* und *Anschlussadapter* werden in der rechten Ecke des unteren Teils des Bausteins angezeigt.
- Die Namen- und *Typdeklarationen* von *Ausgangsvariablen* und *Steckeradapter* werden in der rechten Ecke des unteren Teils des Bausteins angezeigt.
- Die *Schnittstelle* des Funktionsbausteintyps zu den *Ereignissen* wird im oberen Teil des Bausteins deklariert, so wie es in 5.2.1.2 festgelegt ist.
- Die *Algorithmen*, die zum Funktionsbausteintyp gehören, werden deklariert, so wie es in 5.2.1.3 festgelegt ist.
- Die Steuerung der *Ausführung* der zugehörigen Algorithmen wird deklariert, so wie es 5.2.1.4 festgelegt ist.

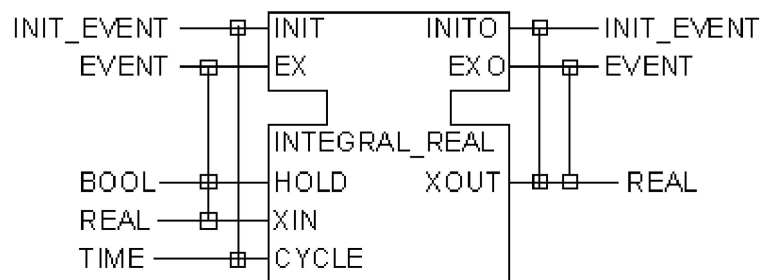


Bild 10 – Deklaration des Basisfunktionsbausteintyps

ANMERKUNG 1 Siehe Anhang F wegen einer textuellen Deklaration dieses Beispiels.

ANMERKUNG 2 Dieses Bild dient nur zur Veranschaulichung. Die grafische Darstellung ist nicht normativ.

5.2.1.2 Deklaration der Ereignisschnittstelle

Wie in Bild 10 gezeigt ist, kann die *Schnittstelle eines Basisfunktionsbausteintyps* zu den *Ereignissen* textuell nach der Syntax deklariert werden, die in Anhang B.2 angegeben ist, oder grafisch nach den folgenden Regeln:

- Die *Ereignisschnittstellen* werden in einem bestimmten Bereich am Bausteinkopf platziert.
- Die *Ereigniseingangsamen* werden an der linken Seite des oberen Teils des Bausteins gezeigt.
- Die *Ereignisausgangsamen* werden an der rechten Seite des oberen Teils des Bausteins gezeigt.
- Die *Ereignistypen* werden außerhalb des Blocks neben ihren zugehörigen Ereigniseingängen oder -ausgängen gezeigt.

ANMERKUNG 1 Falls kein Ereignistyp für einen Ereigniseingang oder -ausgang angegeben ist, wird er als voreingestellter Typ EVENT betrachtet.

ANMERKUNG 2 Ein Ereignisausgang vom Typ EVENT kann mit einem Ereigniseingang beliebigen Typs verbunden werden, und ein Ereigniseingang vom Typ EVENT kann ein Ereignis beliebigen Typs empfangen.

ANMERKUNG 3 Ein Ereignisausgang mit einem anderen Typ als EVENT kann nur mit einem Ereigniseingang desselben Typs oder des Typs EVENT verbunden werden.

ANMERKUNG 4 Ein Ereignistyp wird implizit durch seine Verwendung in einer Ereignisdeklaration deklariert.

Wie in Bild 10 und Anhang F veranschaulicht, muss das Bestimmungszeichen WITH oder eine gleichwertige grafische Darstellung verwendet werden, um festzulegen, welche *Eingangsvariablen* oder *Ausgangsvariablen* beim Auftreten eines *Ereignisses* am zugehörigen *Ereigniseingang* beziehungsweise *Ereignisausgang* abgetastet werden müssen.

Jede *Eingangsvariable* und *Ausgangsvariable* erscheint in null oder mehr WITH-Abschnitten oder seiner äquivalenten grafischen Darstellungen.

ANMERKUNG 5 Diese Information darf verwendet werden, um die erforderlichen Kommunikationsdienste zu bestimmen, wenn eine verteilte *Anwendung nach Abschnitt 7 konfiguriert* wird.

ANMERKUNG 6 Eine Eingangsvariable, die nicht in einem WITH-Abschnitt erscheint, kann nicht mit einer Ausgangsvariablen eines anderen Funktionsbausteins verbunden werden. Die Werte solcher Variablen bleiben entweder bei ihren voreingestellten Anfangswerten oder sie werden durch Managementbefehle wie WRITE eingerichtet, wie es in 6.3.2 beschrieben ist.

ANMERKUNG 7 Eine Ausgangsvariable, die nicht in einem WITH-Abschnitt erscheint, kann mit einer Eingangsvariablen eines anderen Funktionsbausteins verbunden oder durch einen Managementbefehl wie READ nach 6.3.2 „gelesen“ werden.

ANMERKUNG 8 Siehe 4.5.3 für eine Anwendung des Bestimmungszeichens WITH im Ausführungsmodell eines Basisfunktionsbausteins.

5.2.1.3 Deklaration des Algorithmus

Wie in Anhang F gezeigt, dürfen *Algorithmen*, die einem *Basisfunktionsbausteintyp* zugeordnet sind, in die Deklaration des Funktionsbausteintyps eingeschlossen werden; dies gilt nach den Regeln für die Deklaration der Festlegung von Funktionsbausteintypen, die in Anhang B angegeben sind. Es können auch andere Mittel zur Festlegung der Bezeichner und Rümpfe der Algorithmen verwendet werden; die Festlegung solcher Mittel gehört jedoch nicht zum Anwendungsbereich dieses Teils der IEC 61499.

Die Deklaration eines Algorithmus kann auch die Deklaration von temporären Variablen beinhalten, welche:

- nur innerhalb des Algorithmus sichtbar sind;
- bei jedem Aufruf des Algorithmus initialisiert werden;
- während der Ausführung des Algorithmus benutzt und verändert werden dürfen; und
- keine Werte haben welche zwischen den Ausführungen des Algorithmus erhalten bleiben.

5.2.1.4 Deklaration der Ausführungssteuerung der Algorithmen

Die Abfolge der Aufrufe der Algorithmen für die *Basisfunktionsbausteintypen* kann in der Festlegung des Funktionsbausteintyps deklariert werden. Wenn die Algorithmen eines Basisfunktionsbausteintyps so definiert sind, wie es in 5.2.1.3 festgelegt ist (oder in anderer Art angegeben ist), dann kann die Abfolge des Algorithmusaufrufs für einen solchen Funktionsbaustein in der Form eines *Plans zur Ausführungssteuerung* (en: ECC) erfolgen, der aus *EC-Zuständen*, *EC-Transitionen* und *EC-Aktionen* besteht (EC: en: Execution Control = Ausführungssteuerung). Diese Elemente müssen folgendermaßen dargestellt und interpretiert werden:

- a) Der ECC muss in einem Abschnitt *Ausführungssteuerung* der Deklaration des Funktionsbausteintyps enthalten sein, der als im oberen Teil des Bausteins liegend betrachtet wird.

- b) Der ECC muss genau einen *EC-Anfangszustand* enthalten, der grafisch mit Doppellinie dargestellt ist und einen zugehörigen *Bezeichner* besitzt. Der EC-Anfangszustand darf keine zugehörigen EC-Aktionen besitzen.
- c) Der ECC muss einen oder mehrere *EC-Zustände* enthalten, die grafisch als mit einer einfachen Linie dargestellt sind und denen jeweils ein *Bezeichner* zugeordnet ist.
- d) Der ECC kann Variablen verwenden, aber nicht verändern, die in einer Funktionsbausteindefiniert sind.
- e) Ein *EC-Zustand* kann null oder mehr zugeordnete *EC-Aktionen* besitzen. Die Zuordnung der EC-Aktionen zum EC-Zustand kann in grafischer oder textueller Form ausgedrückt werden.
- f) Der *Algorithmus* (sofern vorhanden), der einer EC-Aktion zugeordnet ist, und das *Ereignis* (falls vorhanden), das nach Vollendung des Algorithmus ausgegeben wird, müssen in grafischer oder textueller Form ausgedrückt werden.
- g) Eine *EC-Transition* wird grafisch oder textuell als gerichtete Verbindung von einem EC-Zustand zum anderen (oder denselben Zustand) dargestellt.
- h) Jede EC-Transition muss eine zugehörige Transitionsbedingung haben, welche in der Syntax für die nicht-terminale *ec_transition_condition* in Anhang B.2.1 ausgedrückt wird.

Bild 11 veranschaulicht die Elemente eines ECC. Eine äquivalente textuelle Deklaration, die die Syntax von Anhang B.2 verwendet, ist in Anhang F angegeben.

ANMERKUNG 1 Die Notation 1 (eins), wie in Bild 11 dargestellt, wird als Äquivalent zu [TRUE], das eine Transitionsbedingung ohne zugehöriges Ereignis und eine Schutzbedingung, welche immer TRUE ist, betrachtet.

ANMERKUNG 2 In diesem beschränkten Fall kann dasselbe Symbol (z. B. INIT) verwendet werden, um einen EC-Zustand und einen Algorithmusnamen darzustellen, da die Bedeutung des Symbols leicht aus seiner Verwendung abgeleitet werden kann.

ANMERKUNG 3 Der kursive Text ist nicht Teil des ECC.

ANMERKUNG 4 Eine Eins-zu-eins-Zugehörigkeit von Ereignissen mit Algorithmen, wie es in diesem Bild veranschaulicht ist, tritt oft auf, ist aber nicht die einzige mögliche Verwendung. Siehe Tabelle A.1 mit Beispielen von anderen Verwendungen, z. B. zeigt der E_SPLIT-Baustein eine Zuordnung von zwei Ereignisausgängen mit einem Zustand, aber keinen Algorithmen. E_MERGE zeigt eine Zuordnung von einem Ausgangsereignis, aber keinen Algorithmen, mit zwei Ereigniseingängen. E_DEMUX zeigt einen von mehreren Algorithmen, zugeordnet einem einzelnen Eingangereignis usw.

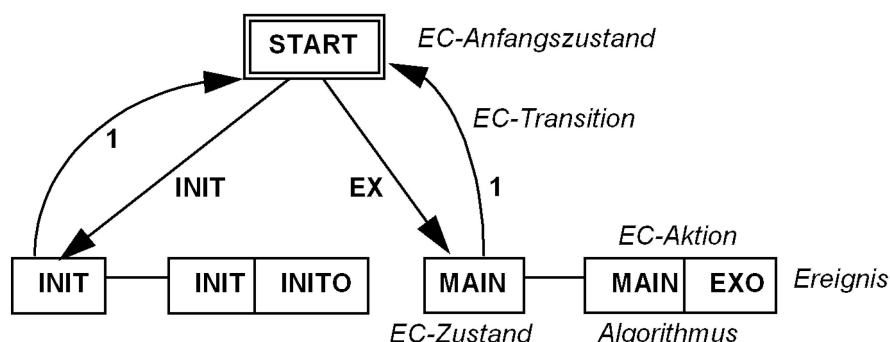


Bild 11 – Beispiel für ein ECC (Plan zur Ausführungssteuerung)

5.2.2 Verhalten von Instanzen

5.2.2.1 Initialisierung

Die Initialisierung einer *Instanz* eines Basisfunktionsbausteins durch eine *Ressource* muss funktional äquivalent mit der folgenden Prozedur sein:

- a) Der Wert jeder *Eingangs*-, *Ausgangs*- und *internen Variablen* muss mit dem entsprechenden *Anfangswert* initialisiert werden, der in der Spezifikation des Funktionsbausteintyps angegeben ist. Falls kein derartiger Anfangswert definiert ist, muss der Wert der Variablen auf den voreingestellten Anfangswert initialisiert werden, der für den Datentyp der Variablen definiert ist.
- b) Alle zusätzlichen algorithmusspezifischen Initialisierungen müssen durchgeführt werden; zum Beispiel müssen alle *Anfangsschritte* nach IEC 61131-3-*Ablaufsprache (AS)* aktiviert werden und alle anderen *Schritte* deaktiviert werden.
- c) Der *EC-Anfangszustand* muss aktiviert und alle anderen *EC-Zustände* müssen deaktiviert werden, und die ECC-Zustandsmaschine nach 5.2.2.2 muss in ihren Anfangszustand (s_0) gesetzt werden.

ANMERKUNG Die Bedingungen, unter denen eine Ressource eine derartige Initialisierung durchführen muss, sind implementierungsabhängig.

Der Funktionsbausteintyp kann auch einen *Algorithmus* für die Initialisierung festlegen, der beim Auftreten des entsprechenden Ereignisses durchgeführt wird, wie zum Beispiel der Algorithmus `INIT`, der in Bild 11 gezeigt ist. Eine *Anwendung* kann dann die Bedingungen festlegen, unter denen dieser Algorithmus auszuführen ist, indem man zum Beispiel einen Ausgang einer Instanz vom Typ `E_RESTART`, der in Anhang A definiert ist, mit einem passenden Eingang `INIT` verbindet, der in Bild 10 gezeigt ist.

5.2.2.2 Algorithmusaufruf

Die *Ausführung* eines *Algorithmus*, der zu einer *Funktionsbausteininstanz* gehört, wird durch eine Anforderung an die **Ausführungsplanfunktion** der *Ressource aufgerufen*, die Ausführung der *Operationen* des Algorithmus aufzurufen.

ANMERKUNG 1 Die Operationen, die durch einen Algorithmus durchgeführt werden, können von einer Ausführung zur nächsten wegen der veränderten Zustände des Funktionsbausteins variieren, auch wenn der Funktionsbaustein nur einen einzigen Algorithmus besitzt und ein einziger Ereigniseingang seine Ausführung auslöst.

Der Aufruf des Algorithmus einer Instanz eines *Basisfunktionsbausteintyps* muss durch die funktional äquivalente Wirkungsweise *seines Plans zur Ausführungssteuerung (ECC)* erreicht werden. Die Wirkungsweise des ECC muss nach den folgenden Regeln erfolgen, die durch die Zustandsmaschine im Bild 12 und Tabelle 1 definiert ist.

ANMERKUNG 2 Ein Auftreten eines Ereignisses an einem Ereigniseingang ruft nach diesem Modell kein Schalten der Transition, die dieses Ereignis enthält, hervor, falls diese nicht dem aktuellen aktiven Zustand entspricht, d. h. wenn das Ereignis im gegebenen Zustand nicht relevant ist. Das Abtasten der Eingangsvariablen, die dem Ereignis mit einem `WITH`-Konstrukt zugeordnet sind, wird jedoch in jedem Fall erfolgen.

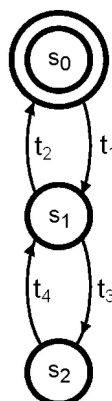


Bild 12 – ECC-Zustandsmaschine

Tabelle 1 – Zustände und Transitionen der ECC-Zustandsmaschine

Zustand		Operationen
s0		–
s1		Transitionen auswerten ^{c, e}
s2		Aktionen ausführen ^{d, e}
Transition	Bedingung	Operationen
t1	ein Eingangsereignis tritt auf ^a	Eingänge abfragen ^{b, e}
t2	keine Transition schaltet	
t3	eine Transition schaltet	
t4	Algorithmen beendet	

a Die Ressource soll sicher stellen, dass nicht mehr als ein Eingangsereignis zu jedem beliebigen Zeitpunkt auftritt.

b Diese Transition besteht aus dem *Abtasten* (oder einer Funktional gleichwertigen Aktion) der Eingangsvariablen, die dem aktuellen Ereignisseingang durch ein *WITH* Deklarationen zugeordnet sind, wie es in 5.2.1.2 beschrieben ist.

c Diese Operation besteht aus dem Auswerten der Transitionsbedingungen bei allen EC-Transitionen, die auf den aktiven EC-Zustand folgen, und dem Schalten der ersten EC-Transition (sofern vorhanden), für die eine *TRUE* Schutzbedingung wie in B.2.1 definiert gefunden wurde. Hierbei gelten folgende Regeln:

- 1) Das „Schalten der EC-Transition“ besteht aus dem Deaktivieren ihres vorhergehenden EC-Zustands und dem Aktivieren ihres nachfolgenden.
- 2) Die Reihenfolge, in der die Transitionsbedingungen ausgewertet werden, entspricht der Reihenfolge wie die Transitionen nach Anhang B.2.1 deklariert sind, oder äquivalent in XML-Syntax nach IEC 61499-2.
- 3) Die Schutzbedingung einer Transitionsbedingung welche nur aus einem *even_input_name* besteht hat den Standardwert *TRUE*.
- 4) Wenn der Eintritt in den Zustand s1 über t1 erfolgt ist, werden nur Transitionsbedingungen welche das aktuelle Eingangsereignis über ihr *event_input_name* nach B.2.1 zugeordnet haben, oder Transitionsbedingungen ohne zugeordnete Ereignisse ausgewertet.
- 5) Wenn der Eintritt in den Zustand s1 über t4 erfolgt ist, werden nur Transitionsbedingungen ohne zugeordnete Ereignisse ausgewertet.

d Diese Operation besteht aus dem Ausführen des – falls vorhanden – zugeordneten Algorithmus für jede EC-Aktion, die dem aktiven EC-Schritt zugeordnet ist, und dem Ausgeben eines Ereignisses am zugeordneten Ereignisausgang, falls er vorhanden ist. Die Reihenfolge, in der die Aktionen ausgeführt werden, entspricht der Reihenfolge, in der sie grafisch von oben nach unten erscheinen oder in der sie in der textuellen Syntax nach Anhang B.2.1 oder der XML-Syntax nach IEC 61499-2 deklariert ist.

e Alle Operationen, die von einem Auftreten der Transition t1 bis zu einem Auftreten von t2 ausgeführt werden, müssen als *kritische Region* mit einer Sperre auf der Funktionsbausteininstanz implementiert sein.

5.2.2.3 Algorithmusausführung

Die *Ausführung* der Algorithmen in einem Basisfunktionsbaustein muss aus der Ausführung einer endlichen Folge von *Operationen* bestehen, die durch **implementierungsabhängige** Regeln festgelegt sind. Diese müssen für die Sprache geeignet sein, in der die Algorithmen geschrieben sind, für die *Ressource*, in der sie ausgeführt werden, und für das Gebiet, für das sie angewendet werden. Die Ausführung der Algorithmen endet nach der Ausführung der letzten Operation in dieser Folge.

Wenn ein Algorithmus eine Zustandsmaschine implementiert, sind wiederholte Ausführungen des Algorithmus notwendig, um die Zustandsänderungen zu erkennen oder zu bearbeiten. Üblicherweise gibt es keine Zuordnung zwischen diesen Zustandsänderungen und der Beendigung des Algorithmus. Solche Zuordnungen müssen durch die Mittel für die Erzeugung der Ereignisausgänge gebildet werden, die in 5.2.2.2 beschrieben sind.

5.3 Zusammengesetzte Funktionsbausteine

5.3.1 Festlegung von Typen

Die Deklaration von *zusammengesetzten Funktionsbausteintypen* muss den Regeln folgen, die in 5.2.1 angegeben sind, mit der Ausnahme, dass *Ereigniseingänge* und *Ereignisausgänge* der Teilfunktionsbausteine mit den Ereigniseingängen und Ereignisausgängen der zusammengesetzten Funktionsbausteine verbunden werden können, um die Abfolge und die Zusammenhänge der Funktionsbausteinaufrufe darzustellen. Die folgenden Regeln müssen bei dieser Verwendung gelten:

- a) Jeder Ereigniseingang eines zusammengesetzten Funktionsbausteins ist mit genau einem Ereigniseingang von genau einem Teilfunktionsbaustein oder mit genau einem Ereignisausgang des zusammengesetzten Funktionsbausteins verbunden, mit der Ausnahme, dass die grafische Kurzdarstellung zur Ereignisaufteilung, die in Bild A.1 von Anhang A gezeigt ist, verwendet werden darf.
- b) Jeder Ereigniseingang eines Teilfunktionsbausteins ist mit nicht mehr als einem Ereignisausgang genau eines anderen Teilfunktionsbausteins oder mit nicht mehr als einem Ereigniseingang des zusammengesetzten Funktionsbausteins verbunden, mit der Ausnahme, dass die grafische Kurzdarstellung zur Ereignisverschmelzung, die in Bild A.1 von Anhang A gezeigt ist, verwendet werden darf.
- c) Jeder Ereignisausgang eines Teilfunktionsbausteins ist mit nicht mehr als einem Ereigniseingang genau eines anderen Teilfunktionsbausteins oder mit nicht mehr als einem Ereignisausgang des zusammengesetzten Funktionsbausteins verbunden, mit der Ausnahme, dass die grafische Kurzdarstellung zur Ereignisaufteilung, die in Bild A.1 von Anhang A gezeigt ist, verwendet werden darf.
- d) Jeder Ereignisausgang eines zusammengesetzten Funktionsbausteins ist von genau einem Ereignisausgang genau eines Teilfunktionsbausteins ausgehend oder von genau einem Ereigniseingang des zusammengesetzten Funktionsbausteins ausgehend verbunden, mit der Ausnahme, dass die grafische Kurzdarstellung zur Ereignisverschmelzung, die in Bild A.1 von Anhang A gezeigt ist, verwendet werden darf.
- e) Der Gebrauch des Bestimmungszeichens `WITH` in der Deklaration von Ereigniseingängen und Ereignisausgängen von zusammengesetzten Funktionsbausteintypen ist erforderlich. Der Gebrauch des Bestimmungszeichens `WITH` bewirkt das *Abtasten* der zugeordneten Dateneingänge, wie im Fall der Basis- oder Dienstschnittstellen-Funktionsbausteine.

ANMERKUNG 1 Software-Werkzeuge können in der Implementierungsphase Mittel zur Beseitigung von redundantem Abtasten zur Verfügung stellen.

- f) *Instanzen von Unteranwendungstypen*, wie in 5.4 definiert, dürfen nicht in der Spezifikation eines zusammengesetzten Funktionsbausteintyps verwendet werden.

Dateneingänge und *Datenausgänge* von *Teilfunktionsbausteinen* können mit Dateneingängen und Datenausgängen des zusammengesetzten Funktionsbausteins verbunden werden, um den Datenfluss innerhalb des zusammengesetzten Funktionsbausteins darzustellen. Die folgenden Regeln müssen bei dieser Verwendung gelten:

- Jeder Dateneingang des zusammengesetzten Funktionsbausteins kann mit null oder mehr Dateneingängen von null oder mehr Teilfunktionsbausteinen verbunden sein oder er kann mit null oder mehr Datenausgängen der zusammengesetzten Funktionsbausteine oder beiden verbunden sein.
- Jeder Dateneingang eines Teilfunktionsbausteins kann mit nicht mehr als einem Datenausgang genau eines anderen Teilfunktionsbausteins oder mit nicht mehr als einem Dateneingang des zusammengesetzten Funktionsbausteins verbunden sein.
- Jeder Datenausgang eines Teilfunktionsbausteins kann mit null oder mehr Dateneingängen von null oder mehr Teilfunktionsbausteinen verbunden sein oder mit null oder mehr Datenausgängen der zusammengesetzten Funktionsbausteine oder beiden verbunden sein.
- Jeder Datenausgang eines zusammengesetzten Funktionsbausteins muss von exakt einem Datenausgang genau eines Teilfunktionsbausteins ausgehend oder von genau einem Dateneingang des zusammengesetzten Funktionsbausteins ausgehend verbunden sein.