

DIN EN IEC 62541-10

ICS 35.200; 35.240.50

Ersatz für
DIN EN 62541-10:2015-11
Siehe Anwendungsbeginn

**OPC Unified Architecture –
Teil 10: Programme
(IEC 62541-10:2020);
Englische Fassung EN IEC 62541-10:2020**

OPC Unified Architecture –
Part 10: Programs
(IEC 62541-10:2020);
English version EN IEC 62541-10:2020

Architecture unifiée OPC –
Partie 10: Programmes
(IEC 62541-10:2020);
Version anglaise EN IEC 62541-10:2020

Gesamtumfang 52 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE

Anwendungsbeginn

Anwendungsbeginn für die von CENELEC am 2020-08-11 angenommene Europäische Norm als DIN-Norm ist 2021-08-01.

Für DIN EN 62541-10:2015-11 besteht eine Übergangsfrist bis 2023-08-11.

Nationales Vorwort

Vorausgegangener Norm-Entwurf: E DIN EN IEC 62541-10:2019-09.

Für dieses Dokument ist das nationale Arbeitsgremium DKE/K 931 „Systemaspekte der Automatisierung“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE (www.dke.de) zuständig.

Aktuelle Informationen zu diesem Dokument können über die Internetseiten von DKE (www.dke.de) und DIN (www.din.de) durch eine Suche nach der Dokumentennummer aufgerufen werden.

Die enthaltene IEC-Publikation wurde vom IEC/SC 65E „Devices and integration in enterprise systems“ erarbeitet und unterliegt dem Copyright der IEC .

Das IEC-Komitee hat entschieden, dass der Inhalt dieses Dokuments bis zu dem Datum (stability date) unverändert bleiben soll, das auf der IEC-Website unter „<http://webstore.iec.ch>“ zu diesem Dokument angegeben ist. Zu diesem Zeitpunkt wird entsprechend der Entscheidung des Komitees das Dokument

- bestätigt,
- zurückgezogen,
- durch eine Folgeausgabe ersetzt oder
- geändert.

In diesem Dokument und in den anderen Teilen der IEC 62541-Reihe werden bestimmte Dokumentenkonventionen verwendet:

Kursivschrift wird verwendet, um einen definierten Begriff oder eine Definition zu kennzeichnen, der/die in dem Abschnitt „Begriffe und Abkürzungen“ in einem der Teile der IEC 62541-Reihe erscheint.

Kursivschrift wird auch verwendet, um den Namen eines Service-Eingangs- oder -Ausgangsparameters oder den Namen einer Struktur oder eines Elements einer Struktur zu kennzeichnen, die normalerweise in Tabellen definiert sind.

Die *kursiv* gedruckten Begriffe und Namen werden mit wenigen Ausnahmen in Camel-Case-Schreibweise geschrieben (die Schreibweise von zusammengesetzten Wörtern oder Phrasen, in denen die Elemente ohne Leerzeichen verbunden sind, wobei der Anfangsbuchstabe jedes Elements innerhalb der Verbindung groß geschrieben wird). Zum Beispiel lautet der definierte Begriff „AddressSpace“ statt „Address Space“. Dies macht es einfacher zu verstehen, dass es eine einzige Definition für AddressSpace gibt und nicht separate Definitionen für Address und Space.

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

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

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

Das Präsidium des DIN hat mit Präsidialbeschluss 1/2004 festgelegt, dass DIN-Normen, deren Inhalt sich auf internationale Arbeitsergebnisse der Informationsverarbeitung gründet, unter bestimmten Bedingungen allein in englischer Sprache veröffentlicht werden dürfen. Diese Bedingungen sind für die vorliegende Norm erfüllt.

Da sich die Benutzer der vorliegenden Norm der englischen Sprache als Fachsprache bedienen, wird die Englische Fassung der EN IEC 62541-10 veröffentlicht. Zu deren Abschnitt 1, der den Anwendungsbereich festlegt und Abschnitt 3, der die Begriffe und Abkürzungen festlegt, wurde eine Übersetzung angefertigt und als informativer Nationaler Anhang NA der vorliegenden Norm hinzugefügt. Für die meisten der verwendeten Begriffe existieren keine gebräuchlichen deutschen Benennungen, da sich die deutschen Anwender in der Regel ebenfalls der englischen Benennungen bedienen. Diese Norm steht nicht in unmittelbarem Zusammenhang mit Rechtsvorschriften und ist nicht als Sicherheitsnorm anzusehen.

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

Änderungen

Gegenüber DIN EN 62541-10:2015-11 wurden folgende Änderungen vorgenommen:

- a) ProgramType zu ProgramStateMachineType geändert. Diese entspricht KnotenEinrichtung (und somit den Implementierungen). In ProgrammDiagnoseDatenTyp wurde die Definition von LetzteEingabeArgumente und LetzteAusgabeArgumente geändert, und es wurden zwei zusätzliche Felder für Argumentwerte hinzugefügt. Außerdem wurde StatusErgebnis zu StatusCode geändert. Neue Version des Typs für ProgrammDiagnose2DatenTyp erstellt.
- b) Optionale Modellierungsregel für Programmsteuerungsmethoden zu OptionalerPlatzhalter geändert. Nach der Klarstellung in IEC 62541-3 erlaubt dies Untertypen (oder Instanzen), so dass Argumente hinzugefügt werden können.

Frühere Ausgaben

DIN EN 62541-10: 2013-06, 2015-11

Nationaler Anhang NA (informativ)

Auszugsweise Übersetzung

Reihenfolge und Inhalt der folgenden Abschnitte sind identisch mit denen in den jeweiligen Abschnitten der Englischen Fassung.

1 Anwendungsbereich

Dieser Teil der Normenreihe IEC 62541 legt das *Informationsmodell* (en: *information model*) fest, welches im Zusammenhang mit *Programmen* in der OPC Unified Architecture (vereinheitlichte Architektur der plattformunabhängigen Kommunikation) (OPC-UA) steht. Dies schließt die Festlegung der *KnotenKlassen* (en: *NodeClasses*), genormter *Merkmale* (en: *Properties*), *Methoden* (en: *Methods*) und *Ereignisse* (en: *Events*) und zugehöriges Verhalten und Informationen für *Programme* ein.

Das vollständige Adressraummodell, einschließlich aller *KnotenKlassen* und *Attribute* (en: *Attributes*) ist in IEC 62541-3 festgelegt. Die *Dienste* (en: *Services*), wie beispielsweise jene zum Aufrufen der *Methoden*, welche zur Verwaltung von *Programmen* verwendet werden, sind in IEC 62541-4 festgelegt.

3 Begriffe und Abkürzungen

3.1 Begriffe

Für die Anwendung dieses Dokuments gelten die Begriffe nach IEC TR 62541-1, IEC 62541-3 und die folgenden Begriffe.

ISO und IEC stellen terminologische Datenbanken für die Anwendung in der Normung unter den folgenden Adressen bereit:

- IEC Electropedia: verfügbar unter <http://www.electropedia.org/>
- ISO Online Browsing Platform: verfügbar unter <http://www.iso.org/obp>

3.1.1

Funktion

(en: function)

programmtechnische Aufgabenstellung, ausgeführt durch einen *Server* oder ein Gerät, üblicherweise durch Ausführung eines Maschinencodes

3.1.2

Endlicher Zustands Automat

(en: finite state machine)

Sequenz von Zuständen und gültigen Zustandsübergängen zusammen mit den Auslösern und Wirkungen dieser Zustandsübergänge, welche die Handlungen eines *Programms* im Sinne von diskreten Stufen definieren

3.1.3

ProgrammZustandsAutomatTyp

(en: ProgramStateMachineType)

Typdefinition eines *Programms* und ein Subtyp des *EndlicherZustandsAutomatTyp*

3.1.4

Programmsteuerungsmethode

(en: program control method)

Methode mit spezifischer Semantik und angelegt zur Steuerung eines *Programms* durch Auslösen eines Zustandsübergangs

3.1.5

Programmaufruf

(en: program invocation)

eindeutige *Objekt*-Instanz eines *Programms*, die auf einem *Server* existiert

Anmerkung 1 zum Begriff: Ein *Programmaufruf* unterscheidet sich von anderen *Objekt*-Instanzen desselben *ProgrammStatusMaschinenTyps* durch den eindeutigen Suchpfad des ObjektKnotens.

3.2 Abkürzungen

DA	Datenzugriff (en: data access)
FSM	Endlicher Zustands Automat (en: finite state machine)
HMI	Mensch Maschine Schnittstelle (en: human-machine interface)
UA	Vereinheitlichte Architektur (en: Unified Architecture)

– Leerseite –

English Version

**OPC Unified Architecture - Part 10: Programs
(IEC 62541-10:2020)**

Architecture unifiée OPC - Partie 10: Programmes
(IEC 62541-10:2020)

OPC Unified Architecture - Teil 10: Programme
(IEC 62541-10:2020)

This European Standard was approved by CENELEC on 2020-08-11. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the CEN-CENELEC Management Centre or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the CEN-CENELEC Management Centre has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Croatia, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and the United Kingdom.



European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

European foreword

The text of document 65E/719/FDIS, future edition 3 of IEC 62541-10, prepared by SC 65E "Devices and integration in enterprise systems" of IEC/TC 65 "Industrial-process measurement, control and automation" was submitted to the IEC-CENELEC parallel vote and approved by CENELEC as EN IEC 62541-10:2020.

The following dates are fixed:

- latest date by which the document has to be implemented at national level by (dop) 2021-05-11
publication of an identical national standard or by endorsement
- latest date by which the national standards conflicting with the (dow) 2023-08-11
document have to be withdrawn

This document supersedes EN 62541-10:2015 and all of its amendments and corrigenda (if any).

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

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

Endorsement notice

The text of the International Standard IEC 62541-10:2020 was approved by CENELEC as a European Standard without any modification.

Annex ZA
(normative)

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

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

NOTE 1 Where an International Publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

NOTE 2 Up-to-date information on the latest versions of the European Standards listed in this annex is available here: www.cenelec.eu.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC/TR 62541-1	–	OPC unified architecture Overview and concepts	– Part 1:CLC/TR 62541-1	–
IEC 62541-3	–	OPC Unified Architecture Address Space Model	– Part 3:–	–
IEC 62541-4	–	OPC Unified Architecture Services	– Part 4:–	–
IEC 62541-5	–	OPC Unified Architecture Information Model	– Part 5:–	–
IEC 62541-7	–	OPC unified architecture Profiles	– Part 7:EN IEC 62541-7	–

CONTENTS

European foreword	8
Annex ZA (normative) Normative references to international publications with their corresponding European publications	9
1 Scope.....	13
2 Normative references	13
3 Terms, definitions and abbreviated terms	13
3.1 Terms and definitions	13
3.2 Abbreviated terms	14
4 Concepts	14
4.1 General.....	14
4.2 Programs	15
4.2.1 Overview.....	15
4.2.2 Security considerations	16
4.2.3 Program Finite State Machine.....	16
4.2.4 Program states	17
4.2.5 State transitions.....	18
4.2.6 Program state transition stimuli	18
4.2.7 Program Control Methods	18
4.2.8 Program state transition effects.....	19
4.2.9 Program result data	19
4.2.10 Program lifetime	20
5 Model	20
5.1 General.....	20
5.2 ProgramStateMachineType.....	21
5.2.1 Overview.....	21
5.2.2 ProgramStateMachineType Properties	23
5.2.3 ProgramStateMachineType components.....	23
5.2.4 ProgramStateMachineType causes (Methods).....	28
5.2.5 ProgramStateMachineType effects (Events).....	30
5.2.6 AuditProgramTransitionEventType	32
5.2.7 FinalResultData	33
5.2.8 ProgramDiagnostic2 DataType	33
5.2.9 ProgramDiagnostic2Type VariableType.....	34
Annex A (informative) Program example.....	36
A.1 Overview.....	36
A.2 DomainDownload Program	36
A.2.1 General.....	36
A.2.2 DomainDownload states.....	37

A.2.3 DomainDownload transitions	37
A.2.4 DomainDownload Methods	38
A.2.4.1 General.....	38
A.2.4.2 Method Arguments	38
A.2.5 DomainDownload Events.....	39
A.2.5.1 General.....	39
A.2.5.2 Event information	39
A.2.5.3 Final result data.....	39
A.2.6 DomainDownload model	39
A.2.6.1 Overview	39
A.2.6.2 DomainDownloadType.....	39
A.2.6.3 Sequence of operations	52
Figure 1 – Automation facility control	15
Figure 2 – Program illustration	16
Figure 3 – Program states and transitions	17
Figure 4 – Program Type.....	21
Figure 5 – Program FSM References	23
Figure 6 – ProgramStateMachineType causes and effects	28
Figure A.1 – Program example.....	36
Figure A.2 – DomainDownload state diagram.....	37
Figure A.3 – DomainDownloadType partial state model.....	43
Figure A.4 – Ready To Running model	46
Figure A.5 – Opening To Sending To Closing model.....	47
Figure A.6 – Running To Suspended model	49
Figure A.7 – Suspended To Running model	49
Figure A.8 – Running To Halted – Aborted model	50
Figure A.9 – Suspended To Aborted model	50
Figure A.10 – Running To Completed model	51
Figure A.11 – Sequence of operations.....	52
Table 1 – Program Finite State Machine.....	16
Table 2 – Program states	17
Table 3 – Program state transitions	18
Table 4 – Program Control Methods	18
Table 5 – ProgramStateMachineType.....	22
Table 6 – Program states	24
Table 7 – Program transitions	25
Table 8 – ProgramStateMachineType causes	29

Table 9 – ProgramTransitionEventType	30
Table 10 – ProgramTransitionEvents	31
Table 11 – AuditProgramTransitionEventType.....	32
Table 12 – ProgramDiagnostic2DataType structure	33
Table 13 – ProgramDiagnostic2DataType definition	34
Table 14 – ProgramDiagnostic2Type VariableType.....	34
Table A.1 – DomainDownload states	38
Table A.2 – DomainDownload Type	40
Table A.3 – Transfer State Machine Type	40
Table A.4 – Transfer State Machine – states	41
Table A.5 – Finish State Machine Type.....	42
Table A.6 – Finish State Machine – states	42
Table A.7 – DomainDownload Type Property Attributes variable values.....	42
Table A.8 – Additional DomainDownload transition types.....	44
Table A.9 – Start Method additions	46
Table A.10 – StartArguments	46
Table A.11 – IntermediateResults Object.....	48
Table A.12 – Intermediate result data Variables	48
Table A.13 – FinalResultData.....	51

OPC UNIFIED ARCHITECTURE –

Part 10: Programs

1 Scope

This part of IEC 62451 defines the *information model* associated with *Programs* in the OPC Unified Architecture. This includes the description of the *NodeClasses*, standard *Properties*, *Methods* and *Events* and associated behaviour and information for *Programs*.

The complete Address Space model including all *NodeClasses* and *Attributes* is specified in IEC 62541-3. The *Services* such as those used to invoke the *Methods* used to manage *Programs* are specified in IEC 62541-4.

2 Normative references

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

IEC TR 62541-1, *OPC Unified Architecture – Part 1: Overview and Concepts*

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

IEC 62541-4, *OPC Unified Architecture – Part 4: Services*

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

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

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC TR 62541-1, IEC 62541-3 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

3.1.1

function

programmatic task performed by a *Server* or device, usually accomplished by computer code execution

3.1.2

finite state machine

sequence of states and valid state transitions along with the causes and effects of those state transitions that define the actions of a *Program* in terms of discrete stages

3.1.3

ProgramStateMachineType

type definition of a *Program* and is a subtype of the *FiniteStateMachineType*

3.1.4

program control method

Method having specific semantics designed for the control of a *Program* by causing a state transition

3.1.5

program invocation

unique *Object* instance of a *Program* existing on a *Server*

Note 1 to entry: A *Program Invocation* is distinguished from other *Object* instances of the same *ProgramStateMachineType* by the object node's unique browse path.

3.2 Abbreviated terms

DA	data access
FSM	finite state machine
HMI	human-machine interface
UA	Unified Architecture

4 Concepts

4.1 General

Integrated automation facilities manage their operations through the exchange of data and the coordinated invocation of system *Functions* as illustrated in Figure 1. *Services* are required to perform the data exchanges and to invoke the *Functions* that constitute system operation. These *Functions* may be invoked through Human Machine Interfaces, cell controllers, or other supervisory control and data acquisition type systems. OPC UA defines *Methods* and *Programs* as an interoperable way to advertise, discover, and request these *Functions*. They provide a normalizing mechanism for the semantic description, invocation, and result reporting of these *Functions*. Together *Methods* and *Programs* complement the other OPC UA *Services* and *ObjectTypes* to facilitate the operation of an automation environment using a client-server hierarchy.

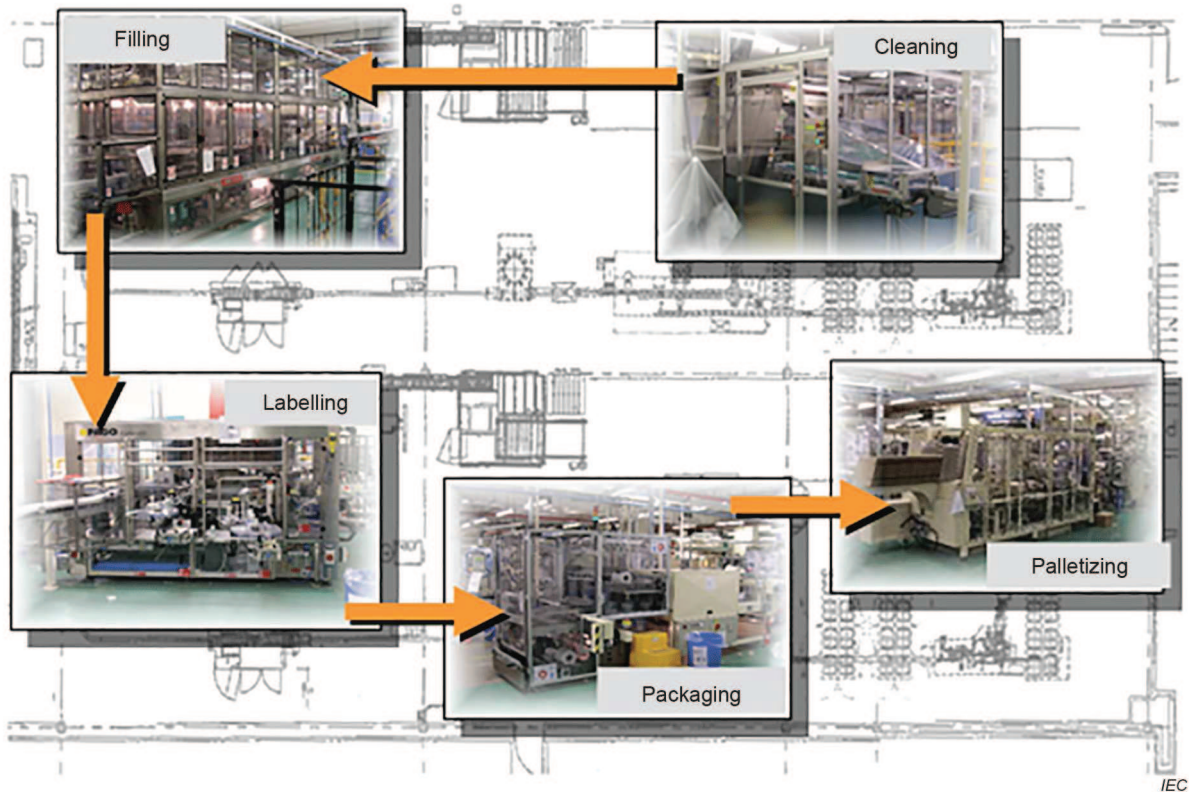


Figure 1 – Automation facility control

Methods and *Programs* model *Functions* typically have different scopes, behaviours, lifetimes, and complexities in *OPC Servers* and the underlying systems. These *Functions* are not normally characterized by the reading or writing of data which is accomplished with the *OPC UA Attribute* service set.

Methods represent basic *Functions* in the *Server* that can be invoked by a *Client*. *Programs*, by contrast, model more complex and stateful functionality in the system. For example, a method call may be used to perform a calculation or reset a counter. A *Program* is used to run and control a batch process, execute a machine tool part program, or manage a domain download. *Methods* and their invocation mechanism are described in IEC 62541-3 and IEC 62541-4.

This document describes the extensions to, or specific use of, the core capabilities defined in IEC 62541-5 as required for *Programs*.

4.2 Programs

4.2.1 Overview

Programs are complex *Functions* in a *Server* or underlying system that can be invoked and managed by a *Client*. *Programs* can represent any level of functionality within a system or process in which *Client* control or intervention is required and progress monitoring is desired. Figure 2 illustrates the model.

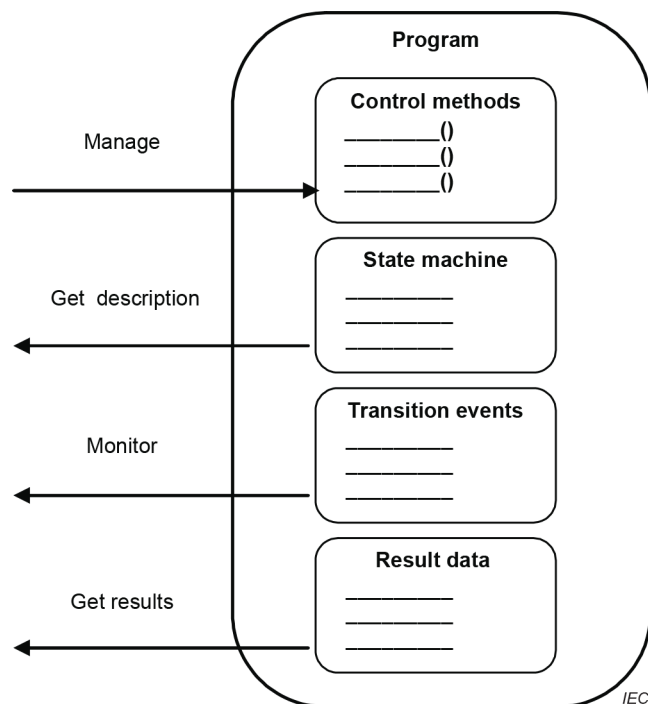


Figure 2 – Program illustration

Programs are stateful and transition through a prescribed sequence of states as they execute. Their behaviour is defined by a *Program Finite State Machine (PFSM)*. The elements of the PFSM describe the phases of a *Program's* execution in terms of valid transitions between a set of states, the stimuli or causes of those transitions, and the resultant effects of the transitions.

4.2.2 Security considerations

Since *Programs* can be used to perform advanced control algorithms or other actions, their use should be restricted to personnel with appropriate access rights. It is recommended that *AuditUpdateMethodEvents* are generated to allow monitoring the number of running *Programs* in addition to their execution frequency.

4.2.3 Program Finite State Machine

The states, transitions, causes and effects that compose the *Program Finite State Machine* are listed in Table 1 and illustrated in Figure 3.

Table 1 – Program Finite State Machine

No.	Transition name	Cause	From state	To state	Effect
1	HaltedToReady	Reset Method	Halted	Ready	Report Transition 1 Event/Result
2	ReadyToRunning	Start Method	Ready	Running	Report Transition 2 Event/Result
3	RunningToHalted	Halt Method or Internal (Error)	Running	Halted	Report Transition 3 Event/Result
4	RunningToReady	Internal	Running	Ready	Report Transition 4 Event/Result
5	RunningToSuspended	Suspend Method	Running	Suspended	Report Transition 5 Event/Result
6	SuspendedToRunning	Resume Method	Suspended	Running	Report Transition 6 Event/Result
7	SuspendedToHalted	Halt Method	Suspended	Halted	Report Transition 7 Event/Result
8	SuspendedToReady	Internal	Suspended	Ready	Report Transition 8 Event/Result