IEC-PAS 61499-1

Edition 1.0
2000-09

Function blocks for industrial-process measurement and control systems

Part 1: Architecture

PUBLICLY AVAILABLE SPECIFICATION

IEC
INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

Reference number
IEC/PAS 61499-1

INTERNATIONAL ELECTROTECHNICAL COMMISSION

____

FUNCTION BLOCKS FOR INDUSTRIAL-PROCESS MEASUREMENT AND CONTROL SYSTEMS –

Part 1: Architecture

FOREWORD

A PAS is a technical specification not fulfilling the requirements for a standard, but made available to the public and established in an organization operating under given procedures.

IEC-PAS 61499-1 has been processed by IEC technical committee 65: Industrial-process measurement and control.

The text of this PAS is based on the following document:	This PAS was approved for publication by the P-members of the committee concerned as indicated in the following document:
Draft PAS	**Report on voting**
65/248/PAS	65/252/RVD

Following publication of this PAS, the technical committee or subcommittee concerned will investigate the possibility of transforming the PAS into an International Standard.

1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, the IEC publishes International Standards. Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. The IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.

2) The formal decisions or agreements of the IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested National Committees.

3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical reports or guides and they are accepted by the National Committees in that sense.

4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.

5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.

6) Attention is drawn to the possibility that some of the elements of this PAS may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. GENERAL REQUIREMENTS

1.1. Scope

This Specification defines a generic architecture and presents guidelines for the use of *function blocks* in distributed industrial-process measurement and control systems (IPMCSs). This architecture is presented in terms of reference *models*, textual syntax and graphical representations. These models, representations and syntax **can be used for**:

- the specification and standardization of *function block types*;
- the functional specification and standardization of system elements;
- the implementation independent specification, analysis, and validation of distributed IPMCSs;
- the *configuration*, *implementation*, operation, and maintenance of distributed IPMCSs;
- the exchange of *information* among *software tools* for the performance of the above *functions*.

NOTE - This Specification does not restrict or specify the functional capabilities of IPMCSs or their system elements, except as such capabilities are represented using the elements defined herein. Clause 5 of this Part addresses the extent to which the elements defined in this Specification may be restricted by the functional capabilities of compliant systems, subsystems, and devices.

Part of the purpose of this specification is to provide reference models for the use of function blocks in other standards dealing with the support of the system life cycle, including system planning, design, implementation, validation, operation and maintenance. The models given in this Specification are intended to be generic, domain independent and extensible to the definition and use of function blocks in other standards or for particular applications or application domains. It is intended that specifications written according to the rules given in this Specification be concise, implementable, complete, unambiguous, and consistent.

NOTE 1 The provisions of this Specification alone are not sufficient to ensure interoperability among devices of different vendors. Standards complying with this Specification may specify additional provisions to ensure such interoperability.

NOTE 2 Standards complying with this Specification may specify additional provisions to enable the performance of *system*, *device*, *resource* and *application* management *functions*.

This Specification consists of two Parts:

- Part 1, "Architecture", contains:
  - general requirements, including an introduction, scope, normative references, definitions, and reference models;
  - rules for the declaration of *function block types*, and rules for the behavior of *instances* of the types so declared;
  - rules for the use of function blocks in the *configuration* of distributed IPMCSs;
  - rules for the use of function blocks in meeting the communication requirements of distributed IPMCSs;
  - rules for the use of function blocks in the management of *applications, resources* and *devices* in distributed IPMCSs;
  - requirements to be met by compliant systems and standards.
- Part 2, "Engineering task support", will present guidance for the support of engineering tasks in the design, implementation, operation and maintenance of distributed industrial-process measurement and control systems constructed according to the architecture defined in this Part.

1.2. Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. At the time of publication, the editions indicated were valid. All normative documents are subject to revision, and parties to agreements based on this specification are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. Members of the IEC and ISO maintain registers of currently valid International Standards.

IEC 600050-351(1998?), International Electrotechnical Vocabulary Chapter 351: Automatic Control (2nd.Ed.)

IEC 559 (1989), Binary floating-point arithmetic for microprocessors

IEC 617-12 (1983), Graphical symbols for diagrams, Part 12: Binary logic elements

IEC 65B/373/CD, Committee Draft - IEC 61131-3, Programmable controllers, Part 3: Programming languages, 2nd Ed., 1998-11-27.

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

ISO 8601:1988, Data elements and interchange formats - Information interchange - Reresentations 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, 1994

ISO/IEC 8824: 1990, Information technology - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)

ISO/IEC 8825: 1990, Information technology - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)

ISO TR 8509-1987, Information processing systems - Open Systems Interconnection - Service conventions

ISO/IEC 10040-1992, Information technology - Open Systems Interconnection - Systems management overview

ISO/IEC 10646-1:1993, Information technology - Universal multiple-octet coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane

1.3. Definitions

NOTE 1 - Terms defined in this clause are *italicized* where they appear in the bodies of definitions.

NOTE 2 - The ISO/AFNOR *Dictionary of computer science* and the *International Electrotechnical Vocabulary* should be consulted for terms not defined or referenced in this specification.

1.3.1. Definitions from other standards

NOTE Definitions are written out in this document for convenience. To avoid duplication, the terms alone will be listed in the final International Standard.

For the purposes of this specification, the following terms as defined in IEC 60050-351 apply:

interface: A shared boundary between two *functional units,* defined by functional characteristics, signal characteristics, or other characteristics as appropriate.

system: A set of interrelated elements considered in a defined context as a whole and separated from its environment.

Notes: 1 -Such elements may be both material objects and concepts as well as the results thereof (e.g. forms of organisation, mathematical methods, and programming languages)
2 - The system is considered to be separated from the environment and other external systems by an imaginary surface, which can cut the links between them and the considered system.

For the purposes of this specification, the following terms as defined in the various Parts of ISO 2382 apply:

NOTE - Definition numbers from ISO 2382 are given in parentheses following the definition.

data type: A set of values together with a set of permitted *operations*. (15.04.01)

data: A reinterpretable representation of *information* in a formalized manner suitable for communication, interpretation or processing. (01.01.02)

functional unit: An *entity* of *hardware* or *software,* or both, capable of accomplishing a specified purpose. (01.01.40)

mapping: A set of values having defined correspondence with the quantities or values of another set. (02.04.05)

message: An ordered series of *characters* intended to convey *information*. (16.02.01)

message sink: That part of a communication *system* in which *messages* are considered to be received. (16.02.03)

message source: That part of a communication *system* from which *messages* are considered to originate. (16.02.02)

network: An arrangement of *nodes* and interconnecting *branches*. (01.01.44)

operation: A well-defined action that, when applied to any permissible combination of known *entities*, produces a new *entity*. (02.10.01)

parameter: A *variable* that is given a constant value for a specified *application* and that may denote the application. (02.02.04)

For the purposes of this specification, the following terms as defined in the ISO/AFNOR *Dictionary of Computer Science* apply:

character: A member of a set of elements that is used for the representation, organization, or control of *data*.

connection: An association established between *functional units* for conveying *information*.

hardware: Physical equipment, as opposed to programs, procedures, rules and associated documentation.

information: The meaning that is currently assigned to *data* by means of the conventions applied to that data.

For the purposes of this specification, the following term as defined in the document *IEC DIS 61508-4: Functional safety - Safety-related systems - Part 4: Definitions and Abbreviations of Terms* applies:

fault: abnormal condition that may cause a reduction in, or loss of, the capability of a *functional unit* to perform a required *function*.

1.3.2. Additional definitions

The following terms are defined for the purposes of this specification.

1.3.2.1. **acceptor:** A *function block instance* which provides a *socket adapter* of a defined *adapter interface type*.

1.3.2.2. **access path:** The association of a symbolic name with a *variable* for the purpose of open communication.

1.3.2.3. **adapter connection:** A *connection* from a *plug adapter* to a *socket adapter* of the same *adapter interface type*, which carries the flows of *data* and *events* defined by the adapter interface type.

1.3.2.4. **adapter interface type:** A *type* which consists of the definition of a set of *event inputs, event outputs, data inputs*, and *data outputs*, and whose *instances* are *plug adapters* and *socket adapters*.

1.3.2.5. **algorithm:** A finite set of well-defined rules for the solution of a problem in a finite number of *operations*.

1.3.2.6. **application:** A *software functional unit* that is specific to the solution of a problem in industrial-process measurement and control.

NOTE: An application may be distributed among *resources*, and may communicate with other applications.

1.3.2.7. **attribute:** a property or characteristic of an *entity*, for instance, the version identifier of a *function block type* specification.

1.3.2.8. **basic function block type:** a *function block type* which cannot be decomposed into other function blocks and which utilizes an *execution control chart (ECC)* to control the *execution* of its *algorithms*.

1.3.2.9. **bidirectional transaction:** A *transaction* in which a request and possibly *data* are conveyed from an *requester* to a *responder*, and in which a response and possibly data are conveyed from the responder back to the requester

1.3.2.10. **communication connection:** A *connection* which utilizes the "communication mapping function" of one or more *resources* for the conveyance of *information*.

1.3.2.11. **communication function block:** A *service interface function block* which represents the *interface* between an *application* and the "communication mapping function" of a *resource*.

1.3.2.12. **communication function block type:** A *function block type* whose *instances* are *communication function blocks*.

1.3.2.13. **component function block:** A *function block instance* which is used in the specification of an *algorithm* of a *composite function block type.*

NOTE - A component function block can be of *basic, composite or service interface type.*

1.3.2.14. **component subapplication:** A *subapplication instance* which is used in the specification of a *subapplication type.*

1.3.2.15. **composite function block type:** A *function block type* whose *algorithms* and the control of their *execution* are expressed entirely in terms of interconnected *component function blocks, events,* and *variables.*

1.3.2.16. **concurrent:** Pertaining to *algorithms* that are *executed* during a common period of time during which they may have to alternately share common *resources*.

1.3.2.17. **configuration (of a *system* or *device*) :** A step in system design: selecting *functional units*, assigning their locations and defining their interconnections.

1.3.2.18. **configuration (of a *programmable controller system*) :** A language element corresponding to a *programmable controller system* as defined in IEC 61131-1.

1.3.2.19. **configuration parameter:** A *parameter* related to the *configuration* of a *system, device* or *resource.*

1.3.2.20. **confirm primitive:** A *service primitive* which represents an interaction in which a *resource* indicates completion of some *algorithm* previously *invoked* by an interaction represented by a *request primitive*.

1.3.2.21. **critical region:** An *operation* or a sequence of operations which is *executed* under the exclusive control of a locking object which is associated with the *data* on which the operations are performed.

1.3.2.22. **data connection:** An association between two *function blocks* for the conveyance of *data*.

1.3.2.23. **data input:** An *interface* of a *function block* which receives *data* from a *data connection.*

1.3.2.24. **data output:** An *interface* of a *function block* which supplies *data* to a *data connection.*

1.3.2.25 **declaration:** The mechanism for establishing the definition of an *entity*. A declaration may involve attaching an *identifier* to the entity, and allocating *attributes* such as *data types* and *algorithms* to it.

1.3.2.26. **device:** An independent physical *entity* capable of performing one or more specified *functions* in a particular context and delimited by its *interfaces*.

NOTE - A *programmable controller system* as defined in IEC 61131-1 is a *device* in the terms of this Specification.

1.3.2.27. **device management application:** An *application* whose primary function is the management of a multiple *resources* within a *device.*

1.3.2.28. **entity:** A particular thing, such as a person, place, *process*, object, concept, association, or *event*.

1.3.2.29. **event:** An instantaneous occurrence that is significant to scheduling the *execution* of an *algorithm*.

NOTE - The execution of an algorithm may make use of *variables* associated with an event.

1.3.2.30. **event connection:** An association among *function blocks* for the conveyance of *events*.

1.3.2.31. **event input:** An *interface* of a *function block* which receives *events* from an *event connection.*

1.3.2.32. **event input variable (EI variable):** A Boolean *variable* corresponding to an *event input.*

1.3.2.33. **event output:** An *interface* of a *function block* which issues *events* to an *event connection.*

1.3.2.34. **event output variable (EO variable):** A Boolean *variable* corresponding to an *event output.*

1.3.2.35. **exception:** An *event* that causes suspension of normal *execution*.

1.3.2.36. **execution:** The process of carrying out a sequence of *operations* specified by an *algorithm.*

NOTE - The sequence of operations to be executed may vary from one *invocation* of a *function block instance* to another, depending on the rules specified by the function block's *algorithm* and the current values of *variables* in the function block's data structure.

1.3.2.37. **execution control action (EC action):** An element associated with an *execution control state* which identifies an *algorithm* to be *executed* and an *event* to be issued on completion of execution of the algorithm.

1.3.2.38. **execution control chart (ECC):** A graphical or textual representation of the causal relationships among *events* at the *event inputs* and *event outputs* of a *function block* and the *execution* of the function block's *algorithms*, using *execution control states, execution control transitions,* and *execution control actions*.

1.3.2.39. **execution control initial state (EC initial state):** The *execution control state* which is active upon initialization of an *execution control chart*.

1.3.2.40. **execution control state (EC state):** A situation in which the behavior of a *basic function block* with respect to its *variables* is determined by the *algorithms* associated with the *execution control state* through its *execution control action*.

1.3.2.41. **execution control transition (EC transition):** The condition whereby control passes from a predecessor *execution control state* to a successor *execution control state*.

1.3.2.42. **function:** A specific purpose of an *entity* or its characteristic action.

1.3.2.43. **function block (function block instance):** A *software functional unit* comprising an individual, named copy of a data structure and associated *operations* specified by a corresponding *function block type*.

NOTE 1 - Typical operations of a function block include modification of the values of the data in its associated data structure.

NOTE 2 - The *function block instance* and its corresponding *function block type* defined in IEC 61131-3 are programming language elements with a different set of features.

1.3.2.44. **function block network:** A *network* whose nodes are *function blocks* or *subapplications* and their *parameters* and whose branches are *data connections* and *event connections*.

NOTE - This is a generalization of the *function block diagram* defined in IEC 61131-3.

1.3.2.45. **identifier:** One or more *characters* used to name an *entity*.

1.3.2.46. **implementation:** The development phase in which the *hardware* and *software* of a *system* become operational.

1.3.2.47. **indication primitive:** A *service primitive* which represents an interaction in which a *resource* either: a) indicates that it has, on its own initiative, *invoked* some *algorithm*; or b) indicates that an *algorithm* has been invoked by a peer *application*.

1.3.2.48. **input variable:** A *variable* whose value is supplied by a *data input*, and which may be used in one or more *operations* of a *function block*.

NOTE - An *input parameter* of a *function block*, as defined in IEC 61131-3, is an *input variable.*

1.3.2.49. **instance:** A *functional unit* comprising an individual, named *entity* with the *attributes* of a defined *type.*

1.3.2.50. **instance name:** An *identifier* associated with and designating an *instance.*

1.3.2.51. **instantiation:** The creation of an *instance* of a specified *type.*

1.3.2.52. **internal operations (of a *function block*):** *Operations* associated with an *algorithm* of a function block, with its *execution* control, or with the functional capabilities of the associated *resource*.

1.3.2.53. **internal variable:** A *variable* whose value is used or modified by one or more *operations* of a *function block* but is not supplied by a *data input* or to a *data output*.

1.3.2.54. **invocation:** The process of initiating the *execution* of the sequence of *operations* specified in an *algorithm.*

1.3.2.55. **literal:** A lexical unit that directly represents a value.

1.3.2.56. **management function block:** A *function block* whose primary *function* is the management of *applications* within a *resource.*

1.3.2.57. **management resource:** A *resource* whose primary *function* is the management of other *resources.*

1.3.2.58. **model:** A representation of a real world process, *device*, or concept.

1.3.2.59 **multitasking:** A mode of operation that provides for the *concurrent execution* of two or more *algorithms*.

1.3.2.60. **output variable:** A *variable* whose value is established by one or more *operations* of a *function block,* and is supplied to a *data output*.

NOTE - An *output parameter* of a *function block*, as defined in IEC 61131-3, is an *output variable.*

1.3.2.61. **plug adapter:** An *instance* of an *adapter interface type* which provides a starting point for an *adapter connection* from a *provider* function block.

1.3.2.62. **provider:** A *function block instance* which provides a *plug adapter* of a defined *adapter interface type*.

1.3.2.63. **request primitive:** A *service primitive* which represents an interaction in which an *application* invokes some *algorithm* provided by a *service*.

1.3.2.64. **requester:** A *functional unit* which initiates a *transaction* via a *request primitive.*