The IEC 61499 Function Block Standard: Overview of the Second Edition

JAMES H. CHRISTENSEN
Holobloc Inc, Cleveland Heights, OH USA

THOMAS STRASSER
AIT Austrian Institute of Technology, Vienna AT

ANTONIO VALENTINI
O[3]neida Europe, Padova IT

VALERIY VYATKIN
University of Auckland, NZ

ALOIS ZOITL
Technical University of Vienna, AT

KEYWORDS: function blocks, distributed systems, embedded systems, IEC, standards, control, automation

ABSTRACT

The IEC 61499 Standard for the development, reuse and deployment of Function Blocks in distributed and embedded industrial control and automation systems was first published in 2000-2002 by the International Electrotechnical Commission (IEC) as a series of Publicly Available Specifications (PAS) for trial use and a Technical Report (TR) containing tutorial information. Since then, it has undergone continuous improvement and development as a result of extensive testing in both academic and industrial laboratories and applications. As a result of these developments, IEC 61499 was published in 2005 as an IEC Standard in three Parts: (1) Architecture, (2) Software tool requirements, and (4) Rules for compliance profiles (IEC/TR 61499-3, containing tutorial information, was withdrawn as obsolete in 2007). In the meantime, work has continued on the updating and improvement of the other 3 parts of the Standard, all of which have now been approved for publication as Second Editions this year. This paper presents an overview of the Standard and the improvements made in the Second Edition, with a brief description of potential future work. Finally, a description is presented of potential business and financial benefits to adopters of the Standard.

I. INTRODUCTION

When development of IEC 61499 was begun in 1992, it was originally envisioned to provide a common reference architecture for the use of software objects identified as *Function Blocks* (FBs).[1] These were used in centralized, scanned controller architectures as exemplified in the IEC 61131-3[2] standard for programming

[1] V.Vyatkin, *IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design*, 2nd Edition, ISA, 2012.

[2] IEC 61131-3, *Programmable controllers - Part 3: Programming languages,* 2nd Edition, International Electrotechnical

Presented at ISA Automation Week 2012; http://www.isa.org

of *Programmable Logic Controllers* (PLCs), as well as in the configuration of decentralized, scheduled execution in *Distributed Control Systems* (DCS) as exemplified in the IEC 61804 series of standards[3]. This resulted in an architecture that could be mapped to both application domains, as well as being in itself implementable as a purely event-driven, distributed architecture.

In the period 1996-2000, the Systems Architecture and Engineering Work Package of the *Holonic Manufacturing Systems* (HMS) Consortium[4] undertook a project to demonstrate the feasibility of implementing the IEC 61499 architecture. Since this project was widely distributed geographically, including researchers from the USA, Canada and Europe, it was necessary to maintain some degree of project control via a *Compliance Profile* (CP)[5]. Based on the experience obtained in this and other projects, the PAS documents were improved and updated to IEC Standard status in 2005[6]. IEC/TR 61499-3, containing tutorial information based on the obsolete PAS documents, was withdrawn in 2007, since during the transition from PAS to Standard status, a substantial amount of literature and documents about the usage of this standard had become publicly available.

Based on accumulated experience in multiple implementations of IEC 61499, Working Group 15 of IEC Technical Subcommittee 65B (SC65B/WG15) began maintenance work on Parts 1 and 2 of the Standard in 2009. This WG consists of 19 international automation and control experts from eight countries (i.e., Austria, Germany, Italy, Japan, Netherlands, New Zealand, Switzerland, USA) coming from industry, academia and institutes for research and technology transfer. Following the normal sequence for maintenance of International Standards, the work of SC65B/WG15 has resulted in final approval of Second Editions of the IEC 61499 series as International Standards in late 2012.

II. KEY CONCEPTS OF IEC 61499[7]

IEC 61499-1 defines the *function block type* as the basic unit for encapsulating and reusing Intellectual Property (IP="know-how"). In object-oriented terms, this is a *class* defining the behavior of (possibly) multiple instances. As shown in Figure 1, it includes *event* inputs and outputs as well as the more traditional *data* inputs and outputs, to provide for synchronization between data transfer and program execution in distributed systems.

Commission, Geneva, 2003.

[3] See, for example, IEC 61804-2, *Function blocks (FB) for process control - Part 2: Specification of FB concept*, International Electrotechnical Commission, Geneva, 2006.

[4] See http://www.ims.org/2011/11/hms -- phase-i-and-ii-holonic-manufacturing-systems/

[5] See http://www.holobloc.com/doc/ita/index.htm.

[6] IEC 61499-1, *Function Blocks: Part 1 - Architecture;* IEC 61499-2, *Function Blocks: Part 2 - Software tool requirements*; IEC 61499-4, *Function Blocks: Part 4 - Rules for compliance profiles* (all published by International Electrotechnical Commission, Geneva, 2005).

[7] This section is adapted from http://www.holobloc.com/papers/iec61499/overview.htm, under the terms of the Creative Commons Attribution 3.0 License.

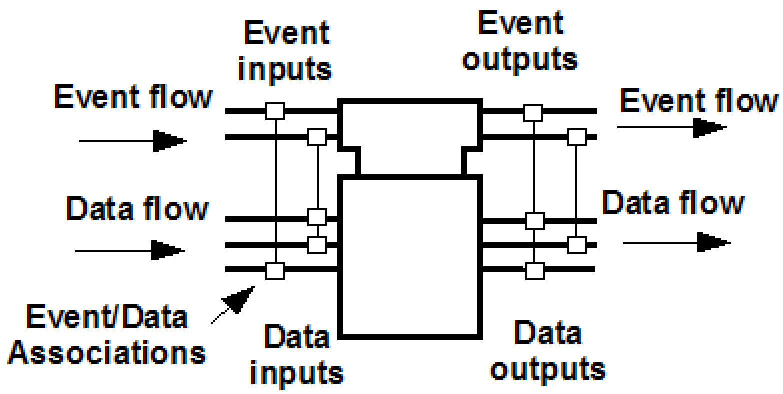

Figure 1 - A Function Block type

As its name implies, the **basic FB type** is the "atom" out of which higher-level "molecules" are constructed. With IEC 61499-2 compliant software tools, software developers can encapsulate IP in the form of *algorithms* written in one of the IEC 61131-3 programming languages or other languages such as Java or C++. As shown in Figure 2, execution of these algorithms is triggered by **Execution Control Charts (ECCs)**, which are event-driven state machines similar to the well-known Harel Statecharts[8].

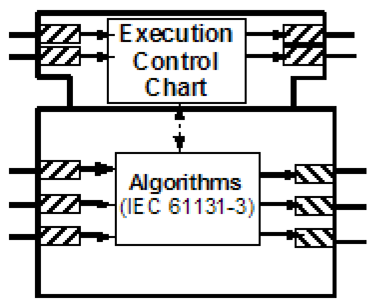

Figure 2 - A Basic Function Block

Another "atomic" function block type is the **Service Interface Function Block** (SIFB) type. This represents the interface to low-level services provided by the operating system or hardware of the embedded device, such as:

- **Graphical User Interface** (GUI) elements such as a slider (illustrated in Figure 3), knob or pilot light
- **Communication services** (the CLIENT_2 SIFB illustrated in Figure 3 is a communication "client" for a remote "server")
- **Interfaces to hardware** such as a temperature sensor, a motor speed controller, a control valve or a room light intensity controller

IEC 61499-2 compliant software tools and and their associated runtime packages can provide a large selection of GUI and communications SIFBs. Providers of **hardware SIFBs** (typically the manufacturers of embedded devices) can use IEC 61499-2 compliant software tools to document how they work in the form

[8] See http://en.wikipedia.org/wiki/Statechart#Harel_statechart

of *service sequence diagrams*.

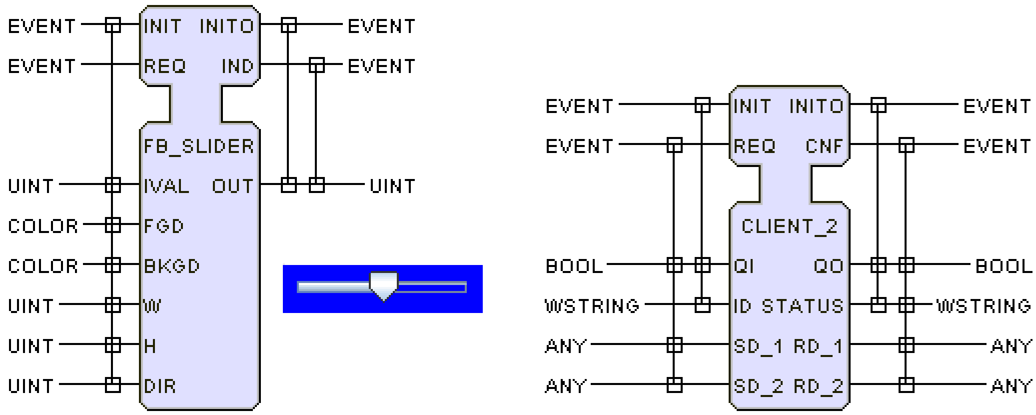

Figure 3 - Service Interface Function Block types

Software developers can use IEC 61499-2 compliant software tools to build higher-level FB "molecules" called **composite FB types** out of lower-level function block "atoms" (component function blocks). This is done by specifying the event and data interfaces of the composite type, then filling it with a diagram showing how its internal component function blocks are connected. In this kind of function block, execution of the algorithms in the component function blocks is controlled by the flow of events from one component to another.

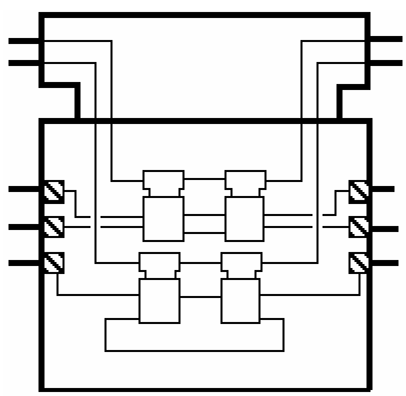

Figure 4 - A Composite FB Type

In the IEC 61499-1 architectural model, distributable **applications** are built by interconnecting *instances* of reusable FB types with appropriate *event and data connections*,in the same manner as designing a circuit board with integrated circuits. Using IEC 61499-2 compliant software tools, these FBs can then be distributed to physical **devices** across a network as shown in Figure 5, as long as these devices comply with the applicable *compliance profile*.

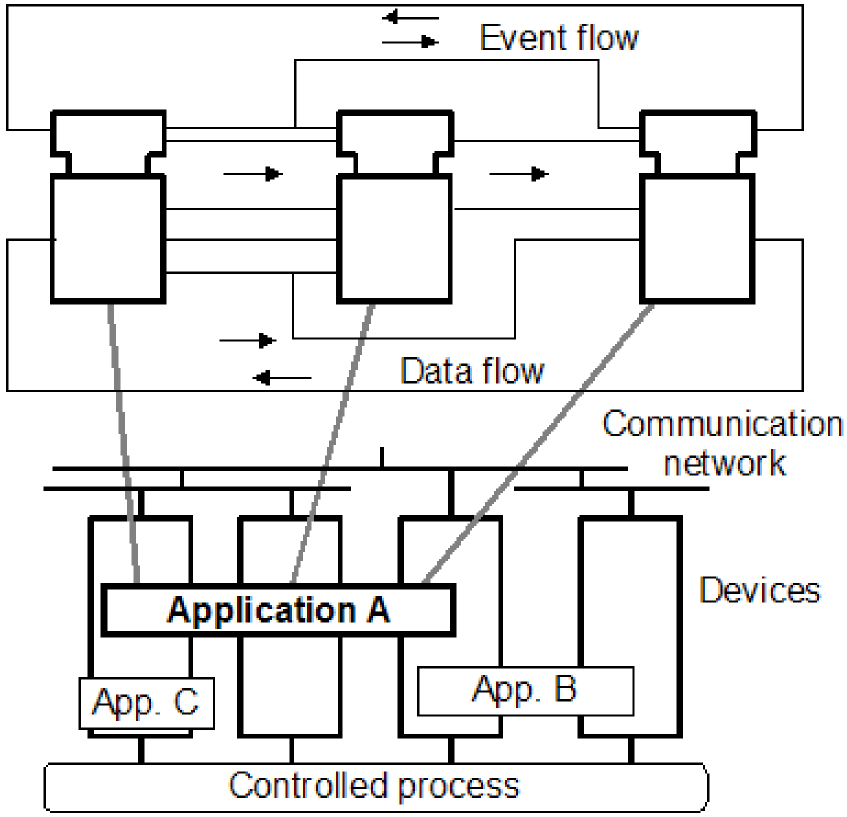

Figure 5 - Distribution of an Application

It is also possible to distribute an application across multiple **resources** within a device. Resources might be multiple processors plugged into a backplane, or multiple tasks within a single processor with a multitasking operating system.

In the IEC 61499 architecture, *resources* are the workhorses that provide the services needed to integrate all the pieces of applications into a working distributed system. IEC 61499-2 compliant software tools can be used to:

- Map the messages that are passed back and forth between devices into the input and output events and data of *communication SIFBs*
- Use event and data inputs and outputs to trigger the performance of the *algorithms* of *basic* and *composite* FBs, and synchronize their operation with other FBs
- Map the data and event inputs and outputs of I/O *SIFBs* to the inputs and outputs of the system, where it can sense what is going on in the physical world and take appropriate physical actions in response.

III. IMPROVEMENTS IN THE SECOND EDITION

The Second Edition of IEC 61499 Parts 1, 2 and 4 will contain changes made in response to approximately 120 editorial and 40 technical comments received from National Committees, with additional editorial changes to conform with IEC requirements. Significant technical changes from the First Edition are listed below.

EXECUTION CONTROL

Interpreting the definitions for executing IEC 61499 FBs, especially how to interpret the definitions for the ECC of basic FBs has been one of the most discussed topics in research and industry in recent years[9, 10]. For this reason, a large part of the refinement work for the second edition has been spent on removing ambiguities in the description of the ECC's behavior in IEC 61499-1. The goal has been to clarify and simplify for device vendors the requirements for implementation of ECC behavior, while at the same time making it more understandable for application developers, so that all can rely on a common understanding of a basic FB's execution behavior.

The first major change in execution control requirements is to resolve *concurrency* issues by requiring that a resource shall ensure that no more than one input event is delivered at any instant in time to the FB. This provision aims at making execution of FB applications more deterministic by excluding possibilities of simultaneous activation of multiple algorithms. If such simultaneous activity is not blocked, it could result in different execution results, even with the same FB application on the same platform.

A second major change resolves *data consistency* issues by requiring that, in conjunction with the event input delivery, *sampling* of input data (or its functional equivalent) shall be performed on those input variables associated with the input event using a graphical or textual WITH construct in the declaration of the FB type. Ambiguities in sampling rules could result in different execution results of the same function block application on different devices.

A third major change resolves a confusion that resulted from a change in the *semantics of events* between the First Edition of IEC Standard 61499-1 and the earlier PAS (Publicly Available Specification) version: in the PAS, an event could be used as a Boolean variable, while in the Standard it was defined as an "instantaneous occurrence." In ECCs the transition conditions of the currently active state are evaluated sequentially on activation by an input event. That means that as long a valid transition condition is found a new state will be entered and its actions (i.e., executing algorithms, sending output events) performed. In the Second Edition of IEC 61499-1, it is clarified that an input event is only valid the first time the transition conditions are evaluated. In any further evaluations (i.e., after the new state has been reached) only conditions without an event in their condition may be evaluated. However, this does not imply that the transition taken at first after activation by the input event needs to have that event in its condition; also in this case a condition without event in its

---

[9] T.Strasser, A.Zoitl, J.H.Christensen, C.Sünder, "Design and Execution Issues in IEC 61499 Distributed Automation and Control Systems," *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and reviews*, Jan 2011, pp.41-51.
[10] V. Vyatkin, "IEC 61499 as Enabler of Distributed and Intelligent Automation: State-of-the-Art Review ," *IEEE Trans. on Industrial Informatics*, vol. 7, no. 4 (Nov 2011), pp. 768-781.

condition may be taken. For instance, as shown in Figure 6, these provisions prevent an infinite iterative or recursive loop between states START and TRIGGERED upon the single occurrence of an EI event with K>3. The importance of this provision for the application developer is explained by the increasing role of ECCs used as a language of application development. During the initial development of the Standard, the ECC was regarded as a simple mechanism for activating different operational modes of the FB (AUTO, MANUAL, etc). However, practical experience has shown that using ECC as a state machine language, in which various combinations of eventful and eventless transitions may occur, simplifies the representation of application logic and reduces the complexity of algorithms. This results in significant improvements in the readability and maintainability of basic FB types. As in other architectural specification frameworks such as UML[11], this usage of the ECC requires more rigorous semantic definition.

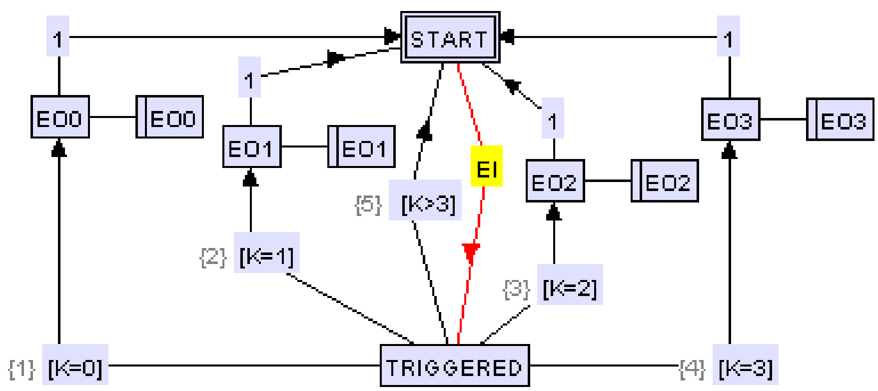

Figure 6 - Execution Control Chart (ECC) example

In order to better show the difference between events and data, a new transition condition syntax in the form of `event_input_name[guard_condition]` is specified in the Second Edition of IEC 61499-1. This is easier to read and reflects the syntax of the Unified Modeling Language (UML), which should make ECCs easier to learn and understand. Additionally, the terminology "clearing a transition" has been changed to "passing a transition" in order to avoid the connotation that a transition could be represented by a user-accessible Boolean variable that could be "set" or "cleared." This in fact was an artifact of a long-standing mistranslation of the French term *franchissement* in the original GRAFCET standard[12].

A final improvement that is given as an example in the Second Edition of IEC 61499-2 is the possibility to represent graphically the priority ordering (i.e., order of evaluation) of multiple transitions from a single EC state, as illustrated in Figure 6. This ordering is significant when such transitions are not mutually exclusive, i.e., when multiple transitions have the same associated event or no associated event.

TEMPORARY VARIABLES

[11] ISO/IEC 19501, *Information technology -- Open Distributed Processing -- Unified Modeling Language (UML) Version 1.4.2*, International Electrotechnical Commission, Geneva, 2005.

[12] IEC 60848, *GRAFCET specification language for sequential function charts*, 2nd Edition, International Electrotechnical Commission, Geneva, 2002.

The definition of *algorithms* in basic FBs has been extended such that the declaration of algorithms may also include the declaration of *temporary variables* using a `VAR_TEMP` declaration as in IEC 61131-3. The behavior of these temporary variables is such that they are only visible within the algorithm in which they are specified. On each algorithm invocation such variables are created and initialized. They may be used and modified during execution of the algorithm but their values do not persist between executions of the algorithm. This greatly improves the readability of basic FBs as variables only needed within an algorithm now need not be declared as internal variables of the FB. A typical application example would be a variable used as loop counter.

NETWORK AND SEGMENT TYPES

A syntax for **segment types** has been introduced in the Second Edition of IEC 61499. This enables the specification of the properties of a specific network type or protocol. IEC 61499-2 compliant software tools can now provide libraries of different kinds of segment types for the system configuration, allowing a clear documentation of the overall system structure. In a further step device types can specify which kinds of ports to different segment types they provide, for example as *resource types* or *SIFB types*. As part of an IEC 61499 system configuration, **links** can be specified between such ports and appropriate **segments** (instances of segment types). This can then be used for ensuring a valid system configuration, and could also provide a basis for future software tool capabilities for automatic network configuration and communication performance evaluation.

INTERACTION WITH PROGRAMMABLE CONTROLLERS (PLCs)

In addition to the facilities defined in the First Edition of IEC 61499-1 for the mapping of functionalities defined in IEC 61131-3 into the IEC 61499 architecture, the Second Edition of IEC 61499-1 defines the use of SIFBs that can act as *clients* of the PLC communication services defined in IEC 61131-5[13]. These include a READ block for synchronous reading of PLC data, a UREAD block for asynchronous reading, and a WRITE block for synchronous writing of PLC data, as well as a TASK block for remote triggering of PLC *tasks* as defined in IEC 61131-3.

SIMPLIFIED ‘READ’ AND ‘WRITE’

The First Edition of IEC 61499-1 adopted the *access path* concept from IEC 61131-3 for use with READ and WRITE management actions to provide access to externally visible interfaces of devices, resources and FBs, and additionally to provide access to internal variables of basic FBs or internal FB elements of composite FBs. However, this concept violates the principles of *encapsulation* and *component orientation* of the IEC 61499 FB model. In order to enforce good software design and enhance system performance, reliability, maintainability and safety, the use of access paths has been removed from IEC 61499. With the removal of access paths, the READ and WRITE management commands are now allowed to access only the interface of FBs, devices and resources, and the internals are hidden from them.

[13] IEC 61131-5, *Programmable controllers - Part 5: Communications*, International Electrotechnical Commission, Geneva, 2001.

ADDITIONAL CHANGES AND CORRECTIONS

In addition to the changes described above, the Second Edition of IEC 61499 contains a number of smaller changes and corrections. These are mainly corrections to add anticipated but forgotten elements, for example a RESET command for management of FB operational states.

Two additional changes provide a unification of language elements across the different FB types. The first change is that *service sequences*, which allow describing the externally-visible dynamic behaviors of a FB, are now allowed for all types of FBs. The second is that definitions for the usage of *adapters* have been extended from *composite* FB types to *basic* FB types as well.

Finally, IEC 61499-2 has been updated to contain informative examples of software tool capabilities and updated Document Type Definitions (DTDs) to conform with the changes in IEC 61499-1.

IV. FUTURE DEVELOPMENTS

In the process of preparing the Second Editions of IEC 61499 Parts 1, 2 and 4, IEC SC65B/WG15 considered several proposals for extensions to the Standard which could enhance its applicability for the development of distributed automation and control systems; however, these proposals were sufficiently developed in detail or sufficiently tested for immediate standardization. Therefore, the WG plans to develop a New Work Item Proposal (NWIP) to prepare a new Part 5 of the IEC 61499 as a Technical Specification (IEC/TS 61499-5) for provisional application to determine the suitability for standardization of some or all of these extensions following a period of testing in practice.

As mentioned earlier, IEC/TR 61499-3 (Tutorial Information) was withdrawn as obsolete in 2007, since it referred to the pre-standard (PAS) version of IEC 61499. Nevertheless, the original content of IEC 61499-3 contains several important points which provide valuable information about the development and application of IEC 61499. Therefore, the SC65B/WG15 is currently discussing the possibility to revise Part 3 of the standard and to include additional tutorial information which can be very useful during the design and implementation of an IEC 61499-compliant implementation of a distributed automation and control system.

V. BUSINESS AND FINANCIAL BENEFITS

The business and financial benefits of widespread adoption of the IEC 61499 Standard are directly impacted by the extent to which to the qualities of **open systems** targeted by IEC 61499-4 and illustrated in Figure 7 are attained.

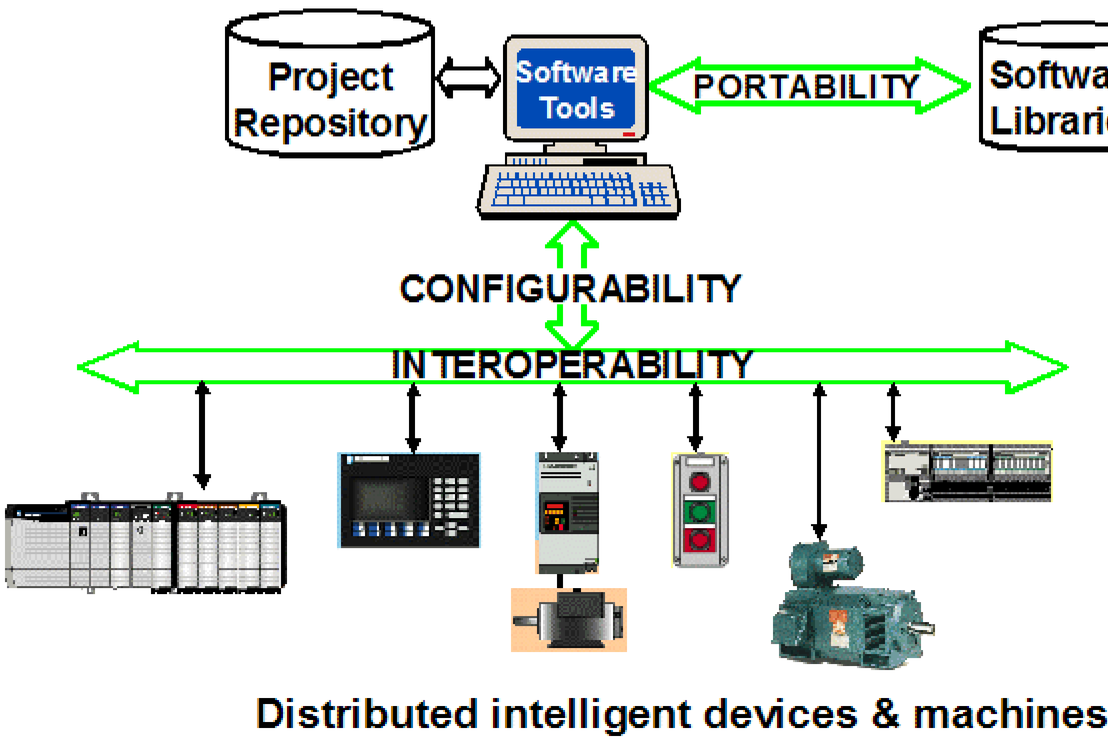

Figure 7 - Qualities of Open Automation and Control Systems
(Source: http://www.holobloc.com/papers/iec61499/overview.htm)

These qualities are defined as:

- **portability:** the extent to which software elements (FB types, data types, resource types, device types, and system configurations) can be accepted and correctly interpreted by multiple software tools
- **configurability:** the extent to which a system can be *configured* via selection of *functional units* (FBs, resources, and devices), assigning their locations and parameters and establishing their data and event interconnections
- **interoperability:** the extent to which functional units in a *system* are able to operate together to perform the required set of automation, control, and data processing functions.

The benefits of adoption of a new architecture such as that defined in the IEC 61499 Standard are also strongly affected by *network externalities,* where the value of the technology to one user depends on how many other users there are:

> Technologies subject to strong network effects tend to exhibit long lead times followed by explosive growth. The pattern results from *positive feedback:* as the installed base of users grows, more and more users find adoption worthwhile.[14]

The adoption pattern of these technologies follows the well-known S-shaped "logistic curve" shown in Figure

[14] C. Shapiro and H.R. Varian, *Information Rules,* Harvard Business School Press, 1999, p. 13.

8, where the vertical axis can be taken to represent the portion of available application "sockets" occupied by the technology (the number of available sockets may actually increase during the lifetime of the technology). Shapiro and Varian characterize this curve as occurring in three phases: **Launch**, **Takeoff**, and **Saturation**[15]. The optimum timing for the various players in the control and automation marketplace can be identified as follows, corresponding to the indicated ranges on the time-scaled horizontal axis:

1. **Launch {-6..-2}:** This is the optimum time for providers of **software tools** and **runtime platforms** to enter the market, in order to establish the dominant software architecture via broadly accepted **compliance profiles**[16].
2. **Takeoff {-2..0}:** This is the optimum time for providers of **hardware platforms** to enter the market, in order to establish a presence among **early adopters** of the technology in the **system integrator** and end-user communities. In the case of IEC 61499, such early adopters may be found where the systems to be controlled are inherently distributed and modular, for instance in material handling and sortation, building automation, pipelines and the "Smart Grid."
3. **Saturation {0..6}:** This or late Takeoff is the optimum time for the large majority of automation system users to be using the technology, as economies of scale drive initial system costs down.

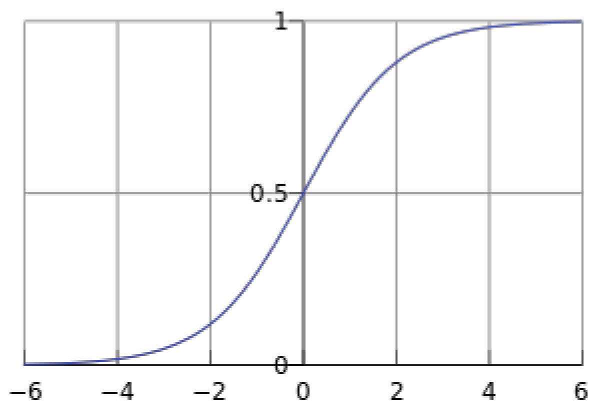

Figure 8 - The Logistic Curve

In order to be ready to adopt the technology at the optimum time, potential adopters should engage in training and maintain close monitoring of the evolution of the technology during the late portion of the preceding phase. In the case of IEC 61499, ample evidence exists[17] to show that the technology is currently in the **late Launch** phase, so the time for most users and system integrators to **begin training, feasibility studies and market monitoring** is **within the next two years**.

---

[15] Shapiro and Varian, *op.cit.,* p.178.

[16] J. Christensen *et al,* "The IEC 61499 Function Block Standard: Software Tools and Runtime Platforms", ISA Automation Week 2012.

[17] J. Christensen *et al,* "The IEC 61499 Function Block Standard: Innovation and Early Adoption", ISA Automation Week 2012.

VI. CONCLUSIONS

The maintenance of IEC 61499 Parts 1, 2, and 4 is completed now, and the Second Edition to be published in late 2012 will be a clear, unambiguous, and industrially useful specification for the use of FBs in distributed and embedded devices and systems. Providers of software tools, runtime platforms and controls hardware should seriously consider entering the market within the next two years; system integrators and end users should begin engagement in training, feasibility studies, and technology monitoring to determine the optimum time for adoption of the IEC 61499 technology.

IEC SC65B/WG15 is expected to focus its future work on an improved Second Edition of the Tutorial Information IEC/TR 61499-3 and on the proposal for a new Technical Specification IEC/TS 61499-5 with significant extensions to provide better support for the development of distributed automation and control systems.