O-PAS conformant communication model for IEC 61499 and IEC 61131 interoperability

Vinicius S. de Oliveira
Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte
Natal, Brazil
vinifoia@ufrn.edu.br

André L. Maitelli
Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte
Natal, Brazil
maitelli@dca.ufrn.com

Andrés O. Salazar
Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte
Natal, Brazil
andres@dca.ufrn.br

Diego Antônio de M. Fonsêca
Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte
Natal, Brazil
diegomoura@dca.ufrn.br

***Abstract*—In this work is proposed a model for communication of different industrial automation standards with focus on the study of IEC 61499, including analysis of specifications and tools for development. Its predecessor (IEC 61131) was created aiming standardization of the languages for Programmable Logic Controllers, but new challenges in the development of industrial automation systems induced the production of new specifications. After a literature review, it was observed an increasingly interest of academy in the new standard, nevertheless, the industrial sphere seems reticent of its adoption compared to the widely accepted 61131. Having as motivation verify the degree of maturity of IEC 61499 as well as EcoStruxure™ Automation Expert software, the main goal of this work is to stablish interoperability of systems from different standards, following O-PAS specifications. Preliminary results show that the communication layer based on OPC UA on a 61499 environment successfully enables integration with other standards, but only for elementary data types, precluding the usage of structured variables, such as O-PAS Signals. Thereby, it is expected to contribute to industrial automation studies with the discussion of comparing both standards and integrating them with the presented methodology.**

***Index Terms*—IEC 61499, industrial automation, PLC, O-PAS, The Open Group**

I. Introduction

Industry, defined as the “ability to perform manual work” [15], is normally related to the production of consumer goods. Therefore, an industrial process is the transformation of raw materials into products for commercialization purposes. To reach the final result, several operations can be carried out (mechanical, physical or chemical), using human force, machines or energy in these tasks.

In the form we know it today, the industry originated in England at the end of the 18th century, faced with changes in production methods, with the replacement of manufacturing (manual production) by the use of machines. The emergence of this new economic activity was so remarkable that this period came to be known as the Industrial Revolution. The use of steel, oil and electricity, in addition to inventions such as the combustion engine, were responsible for an increase in manufacturing performance, culminating in the second phase of the Industrial Revolution.

In a context of globalization, the number of consumers increases massively and to meet this growing demand it was necessary to evolve production and industry management processes. It was then that electronic systems were incorporated into factories, with the development of industrial automation. Computerized and automated systems represent the main milestone of the Third Industrial Revolution, as they made it possible to produce on scales never before imagined.

The equipment that made industry automation possible was the Programmable Logic Controller (PLC), which is responsible for commanding and monitoring machines or industrial processes. PLC is “a digital electronic device that uses a programmable memory to internally store instructions and to implement specific functions, such as logic, sequencing, timing, counting and arithmetic” [11]. They work by controlling actuators, according to data processed from sensor readings. In this way, these devices represent a flexible, robust and adaptable solution, and can be used in practically any application, be it a specific process, machine function or even an entire production line.

Given the demand from the international industrial community for standardization of languages, as well as software structure and program execution in PLCs, the IEC 61131 standard was created in 1992 by the International Electrotechnical Commission (IEC). This organization has the participation of all national electrotechnical committees and aims to promote international cooperation through the standardization of all items related to the fields of electrical and electronics. Therefore, it carries out activities such as publishing International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides [1].

Regarding the 61131 standard, it is a series of, initially, 5 documents that describe each part related to the standard, including general information, equipment requirements, programming languages, user guidelines and communications, respectively. Part 3 [2], in addition to defining the languages Ladder Diagram (LD), Function Block Diagram (FBD), Structured Text (ST), Instruction List (IL) or Sequential Function Chart (SFC), also establishes the controller abstraction levels. Since each machine had its own controller, in IEC 61131 the abstraction is based on its configuration, with the delimitation of resources, tasks and Program Organization Unit (POU).

Since its release, IEC 61131 has been widely accepted in the industrial automation domain. However, the standard has been criticized for not meeting the requirements of the most current complex automated industrial systems and not being compatible with state-of-the-art software engineering practices [14]. Furthermore, there are also uncertainties in the global market and economy, which force manufacturers to quickly adjust to changes in demand, raw materials and energy costs.

Faced with modern challenges characterized by immediacy, the agility to adapt to market changes depends on the practicality of an automated, intelligent and autonomous system. To provide the necessary infrastructure for Industry 4.0, the dynamic reconfiguration of processes is an essential requirement. With this in mind, large systems with central intelligence were giving way to distributed systems, in which each individual part has intelligence and can communicate fluidly with the others, so that the system acts as a whole. In this way, centrally controlled systems will be replaced by intelligent field devices [12], emphasizing their flexibility.

This industry transition to a more distributed paradigm was also motivated by the need to move away from closed proprietary systems and towards an open, secure and interoperable process automation architecture [4]. Faced with this context, IEC developed a new standard focused on industrial automation that was published in 2005: IEC 61499. This represents an extension of IEC 61131, with the pretext of providing the user with greater independence in choosing suppliers.

Aiming to meet portability, interoperability and configurability constraints for complex industrial systems, the IEC 61499 standard makes it possible to model and distribute automated applications independently of the underlying automation hardware. In this way, it is possible to develop devices by different manufacturers that are capable of working together to fulfill a series of automation, control and data processing functions. Equipment in accordance with 61499 also has the characteristic of being able to accept and interpret different software tools, thus, they can be dynamically modified, ensuring that they work in the same way for the same code, even if from different brands.

Speaking of technical specifications, IEC 61499 bases the construction of its systems on Function Blocks (FBs), defined in IEC 61131. They are easy to implement and support access to different networks, so they are easily distributed to controllers and field devices [16]. In view of this, the standard defines the abstraction levels of 61499 solutions, which starts from a view of the entire system, passing through devices and resources, until reaching the FBs.

The first big difference between the standards can be given by the addition of a Execution Control Chart (ECC) in the blocks, which will control the flow of application execution through events. Another distinction is the fact that it does not define the language used to configure the controller, leaving this point is for the programmer to choose, but suggesting the five languages of IEC 61131-3 [2]. Algorithms or programs can be encapsulated in blocks, maintaining their functionality and preserving the developers' intellectual property.

In summary, IEC 61499 brought a new solution for distributed systems, which aim to satisfy new industrial production requirements. Using open source standards, the standard reduces the complexity of implementing robust systems, allowing code reuse and making device reprogramming more flexible. With the 61499 concepts, it is expected to reduce engineering costs and make systems more flexible and sustainable [16].

II. Background

A. *IEC 61131*

As the third industrial revolution, worldwide's production chain was completely changed with the introduction of automation systems. The first solutions of Programmable Logical Controller (PLC) were developed by the companies in their proper way, but as these equipments became more and more widespread among industries, a demand for standardization arised. Thinking about that, the creation of a specification to normatize the launch of PLCs was held by the International Electrotechnical Commission (IEC), which is a organization for standardization in the electrical and electronic fields by promoting international co-operation.

Focusing on the standardization of programming languages for PLCs and conventional industrial controllers the IEC 61131 was then published in 1992. Since then it has been widely accepted in the industrial automation domain also having modifications and updates over the years. At first the standard only included general information about the rules for PLC programming, as well as equipment requirements and user guidelines.

In a few words, the IEC 61131 is a set of documents divided in parts, that define rules for PLCs. Actually there are 10 parts that composes the standard, but Part 4 that contained User Guidelines was withdrawn. Table I shows all the parts of the IEC 61131 standard as well as its actual edition and year of release. With this, it can be noted that the community are constantly working on updates, including new related topics and editions for the standard, such as concepts of object orientation and interoperability features.

Among the rules contained in the parts of the standard, it can be highlighted the part 3, which defines Instruction List, Structured Text, Ladder Diagram, Sequential Function Chart and Function Block Diagram as the 5 possible languages that should be used to program the PLCs. Besides the programming languages, the IEC 61131 has some common elements about

TABLE I
IEC 61131 PUBLICATIONS

Part	Year	Edition	Content
1	2003	2.0	General information
2	2017	4.0	Equipment requirements and tests
3	2013	3.0	Programming languages
5	2000	1.0	Communications
6	2012	1.0	Functional safety
7	2000	1.0	Fuzzy control programming
9	2022	2.0	SDCI for small sensors and actuators
10	2019	1.0	PLC open XML exchange format

structure and configuration to regulate programmable control systems.

In order to avoid operations between different data types variables, in this standard it is indicated elementary types to be used for definition of parameters. They could be of boolean, integer, real, byte and word, but also derivative types with the combination of these, such as date, time and string. The variables have the possibility to be global, meaning they are visible and accessible from all the scope, but each one has to be associated to an explicit hardware address (such as inputs or outputs).

Thinking about the abstraction of systems based on IEC 61131, the upper level is given by the software configuration, in which can be alocated the resources for the realization of tasks. These tasks are responsible for the management of the Program Organization Unit (POU), controlling the execution of functions, programs or function blocks. In this way, all the programming and logic is contained inside the CLP, that will describe the behavior of the system.

B. IEC 61499

As the human population continues to grow dramatically, on the same proportion increases the emmergence of new customers from any part of globe. To be able to meet this demand it is necessary for industries to speed up their production to a level never seen before. Next to that, the efficient adaptation to new products also requires a mass customization with the flexibility of production.

The points just cited characterize the new challenges of industry 4.0, which commonly use technologies like data science and internet of things. Systems designed to meet these requirements are commonly distributed, having its structure based on devices with intelligence and the capability to communicate with others. In this way, smaller parts have specific functions, but work together so that the system can act as a whole.

It is important to mention that automation systems has never lost its relevance in industry since its adoption. With this new scenario, however, comes up a need to provide a generic model for distributed systems that includes processes and communication networks as an environment for embedded devices. Besides the interoperability of equipments, portability and configurability of software tools are also essential for the construction of complex industrial process measurement and control systems.

To achieve all those requirements, the IEC published in 2005 a new standard, called IEC 61499, which was based on the IEC 61131. Briefly, it defines an open architecture, in which the function block type is the basic unit for encapsulating and reusing Intellectual Property. The Function Block interface (Figure 1) is not the same as in the predecessor standard, though. Now it includes an Execution Control Chart, that represents a state machine controlling the execution of the application through states and transitioning by events.

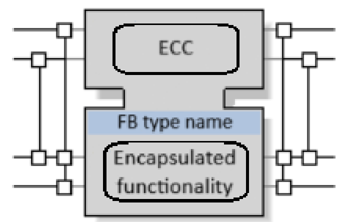

Fig. 1. IEC 61499 Function Block interface.

This standard is also composed by a series of documents, each part related to an aspect of it. Table II shows all the parts of the IEC 61499 standard as well as its actual edition and year of release. With that, it can be noted that this standard does not have as much parts as its predecessor, neither it has many revisions or new parts. This could indicate that IEC 61499 scope is well defined in the three parts it has been originally published (Part 3 contained Tutorial information, but it was withdrawn). Anyway, this does not mean it does not have constant support by the community, seeing that it had a series of revisions in all parts starting in 2012.

TABLE II
IEC 61499 PUBLICATIONS

Part	Year	Edition	Content
1	2012	2.0	Architecture
2	2012	2.0	Software tool requirements
4	2013	2.0	Rules for compliance profiles

When the IEC 61499 was being designed it was intended to be as similar as possible to 61131. Talking about the data types, the same types of its predecessor are used, adding the event type, which can only be linked to event variables. The internal flow is defined by the ECC and is started whenever an input event arrives. After this, the input datas related to that event are updated, the encapsulated functionality is executed and output events are triggered according to the related outputs changes.

In this new standard is that global variables are not used inasmuch as each equipment is independent and autonomous. Another difference between both standards is given by the abstraction, which on IEC 61499 it is possible to visualize the system as a whole. Under the system, it is possible to configure each device individually and manage the resources for the execution of applications. Applications are then built by networks of Function Blocks.

Focusing on the function blocks, they can be Basic (BFB), Composite (CFB) or Service Interface (SIFB), depending on the encapsulated functionality of them. The Basic type has a defined state machine and an algorithm encapsulated, that can be coded in any programming language, leaving this point for the programmer to choose, but suggesting the five languages of IEC 61131-3 [2]. The CFB is just a block with the inclusion of other blocks of any type internally. To access inputs and outputs of devices and make communication between hardware and software are used the Service Interface Function Blocks.

C. The Open Group

It is true that both IEC 61131 and IEC 61499 are constantly being updated to integrate current and emerging industry requirements. In a scenario of open, consensual and interoperable automation architecture, the Open Group was created, which is a global consortium that enables the fulfillment of business objectives through technological standards. With the participation of consumers, manufacturers, system integrators, standardization organizations and academia, the group has several forums for technical discussion and standards production.

The Open Process Automation™ Forum (OPAF) is a consensus-based group of end users, suppliers, system integrators, standards organizations and academia. It addresses both technical and business issues for process automation. Leading the development of open, vendor-neutral technology standards and certifications they intend to ensure openness, interoperability and consensus to their standards (O-PAS), which is called to be the ”standard of standards”.

Inside the OPAF, there are several working groups for each of its standards. Within the technology working group, the team in charge of part 6.5 is responsible for maintaining the IEC 61499 standard. To this end, it holds weekly meetings to address issues such as parameter portability between development tools. Meetings are also held with work sessions to develop functional blocks and extra meetings that can cover any topic related to the team.

D. UniversalAutomation.org

Aiming the creation of an ecosystem of portable, interoperable, “plug and produce” solutions, a community of automation users, technology vendors and universities organized around an independent non-profit association that was called UniversalAutomation.org (UAO). Its main goal is to be a technology enabler based on the IEC 61499, managing the implementation of an industrial automation shared source runtime execution engine. By sharing a reference runtime execution engine implementation of the IEC 61499 standard, UAO intends to create a new era of openness and collaboration, decoupling software and hardware.

This shared runtime execution engine approach not only accelerates the adoption of the technology, but also maximizes portability by standardizing on a common runtime platform. Its software tool to development of 61499 systems is the EcoStruxure™ Automation Expert (EAE), which will enable vendors, users and academics to share a common automation software layer across their automation technology regardless of branding.

Moving away from proprietary systems, the adoption a common runtime execution engine, shared across vendors, will provide limitless opportunities for growth and modernization across industry. Portability and reuse of software components will be responsible for an extension of the lifespan of existing industrial tools, since problems of hardware will be left behind, unleashing the power of software.

UniversalAutomation.org is always open to new members looking to advance the world of automation by working together to develop and increment the runtime execution engine following shared source principles. Their role is to provide platforms for research and development demonstrating the ability of IEC 61499 to deliver sophisticated, distributed intelligent systems and also offering support with training and skills for usage of the tools.

III. Literature Review

As previously seen in the background section, the IEC 61499 standard was initially created to complement IEC 61131, adding features that aim to meet portability, interoperability and configurability. It was also expected to simplify specifications and correct flaws present in the predecessor standard. Therefore, mass adoption by industry and academia of this new standard was expected, as it was with 61131.

In order to analyze the interest and identify trends over time regarding the two standards, a survey of publications was carried out in the Scopus and Web of Science databases. The results were organized taking into account their year of publication, as shown in Figure 2 and Figure 3.

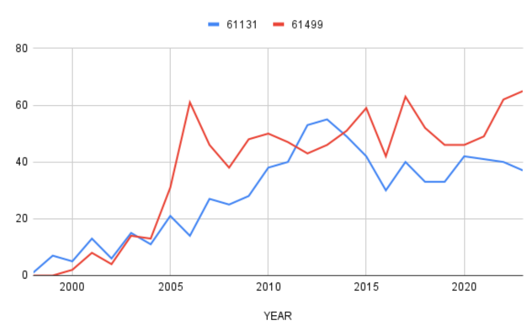

Fig. 2. Publications per year - Scopus.

Analyzing the two charts, it can be concluded that since its release, the newer standard has shown more interest among almost all the years. Even though this, 61131 has never lost its importance, due to the fact that it presents a relevant number of publications, especially between the years of 2012 and 2013, coinciding when the standard had important revisions on Part 6 and Part 3. This shows that there are still working groups in charge of keeping the standard updated by releasing new revisions.

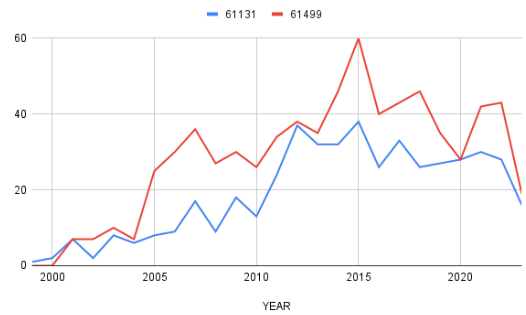

Fig. 3. Publications per year - Web of Science.

The graphs shows that IEC 61499 has had a good adoption by the academic sphere, taking in account the number of scientific work produced. It can be seen in both graphs that IEC 61499 is having an increasingly demand, with a peak of publications in 2015. Anyways, it can still be considered a small number, since it has never reached 100 publications a year.

Even though it is highly promoted by the academic community, 61499 standard was not accepted by industry [14] in a first moment. Despite it has been launched later, this does not means that it has newer features seeing that both are undergoing revisions. By today, 61131 has even included concepts present only in the newer standard, such as object orientation and interoperability features.

”The IEC 61499 standard has emerged in response to the technological limitations encountered in the currently dominating standard IEC 61131” [14]. It is a fact that that 61131 imposes several restrictions for the development of today's complex systems. Its cyclic scan model of computation adopted is “severely inadequate to meet the current industry demands for distributed, flexible automation systems” [14].

In manufacturing industry, distributed control is highly desirable since it provides some benefits such as improved flexibility, reliability, maintainability and reduced wiring costs [5]. In other words, 61499 is readier to the industry 4.0 then its predecessor, as it provides the required infrastructure to industrial IoT applications. Intelligent automation working with decentralized control, enables a smart, automated and independet production, unleashing a step-change in productivity, flexibility and speed for industry.

Even with all that has been pointed, IEC 61131-3-based systems are still prevalent in industry, due to legacy systems and well-trained staff for this type of programming model [9]. ”The event-driven execution of IEC 61499 brings for a number of applications, but far not for all, significant advantages” [19]. In a distributed paradigm, designing new applications following 61499 is highly recommended, but the replacement of existing systems based on older standards could be not that simple.

Allowing the use of the same programming languages of IEC 61131 for implementing algorithms is by far not enough to convince users to adopt the new technology [19]. There are challenges of both technical and economic nature that hinder the industrial reception of the 61499 standard.

Considering the technical barriers, it can be cited that a distributed control system is in fact more difficult to design than a centralized system. Also, as there are differences in execution and data handling, the portability of 61131 based applications to 61499 systems is not possible. This leads to a loss of experience according to the usage of these control systems, requiring training for different qualification and background of control engineers.

Thinking of the large amount of installed IEC 61131-3 systems and large engineering investments undertaken, a fast replacement of the existing systems seems unreasonable [19]. The financial costs related to purchasing of important development tools and developers' training may make impactible the use of 61499 standard in some scenarios.

Other industry interest could also be having a role as impeding the adoption of 61499. While application developers would like to have no limitations when building applications, control system vendors wish to have such extensions limited to their own product portfolio. This way, they can sell additional features through bundling, limiting possible product combinations to the own product spectrum [19].

With that said, it can be noted diverging interests, having vendors on one side and developers and academy on other. It seems to be clear that efforts toward portability, configurability, and interoperability need the common work and specification of a broad positioned user organization [19]. While PLCopen took this role regarding IEC 61131, for the newer standard, The Open Group is leading the efforts to ensure openess and achieve business objectives on accordance with 61499.

Because of the event interface present on 61499, new connections should be added between Function Blocks compared with IEC 61131. This complicates the design of new systems, but it is considered as the value that the developer has to pay in order to be able to explicitly define the execution sequence of Function Blocks [13].

Hereupon it is expected that application codes based on 61499 will have larger measurements than those of the previous standard. Nonetheless, applying software metrics to evaluate code complexity, quality, and effort, it has been shown that depending on the implementation (such as sequencer applications), IEC 61499 could be more suitable [14].

When concepting an automated system from sketch, there is no doubt that by using IEC 61499 concept, it is expected an reduction in engineering cost and system will be more flexible and maintainable. In addition, rapid configuration of industrial process will be used much more frequently to recover from machines and process faults with minimal loss of production [16].

The reticent adoption of 61499, can be explained due to no support to preservation of know-how and already-taken investments in engineering. Despite there is no natively support for integration of IEC 61131 and IEC 61499 function blocks, actually there are some approaches for assist the migration between both standards. While some discuss about the transformation of each block individually [7] [8] [10] [17] [18], others suggests an architecture based on development tools with runtime environment that supports both standards [5] [9].

Those proposals can be helpful when moving from an existing 61131 system through its adequation to 61499. Still it can be advantageous at first time, it may also cause more work for the designer in long term, since it will be expected to know both standards to act in every machine maintenance. With this, industries may prefer to maintain its older systems, bringing to a discussion about the importance of the coexistence and harmonization of both standards, seeing IEC 61499 more as a complementary standard to IEC 61131, rather than a replacement [5].

To assure the widespread adoption and deployment of the IEC 61499 architecture for the next generation of intelligent devices, machines, systems and enterprises, software tools, runtime platforms, and libraries of software components are indispensable [6]. These software tools enables the development of systems conformant to the respectives standards, as well as creation and interpretation of applications.

About the 61131, the main tools available are CODESYS, GEB Automation IDE, Arduino PLC IDE, and so others. The execution model and data handling of the newer standard, make it impossible to directly port IEC 61131-3 applications in an IEC 61499 based runtime environment [5]. This means developers needed to first create the freeware and open source software tools and runtime platforms for the IEC 61499 architecture.

For development of 61499 systems by now, there are some available software tools that are either commercially supported or available as open source. An analysis of FBDK, 4DIAC-IDE, ISaGRAF and nxtSTUDIO was made, taking in consideration portability of elements, configurability of systems and interoperability of devices, and was observed a lack of consensus compliance profile across vendors [6]. To further extend the study of 61499 software tools, it was intended in this present work to create an application using a development environment other than these four.

With this in mind, it was firmed a partnership with UAO through university program, to guarantee access with license to EcoStruxure™ Automation Expert software as well as runtime platform. The reason of this choice was because UAO intend this to be the main platform for creation of 61499 systems. They also firmed a partnership with The Open Group for the utilization of EAE tools, showing they are both walking to develop consensus and facilitate interoperability, to evolve and integrate specifications and open source technologies.

IV. Results and Discussion

Aiming evalute the stage of maturity of EAE tool, validation tests were made in order to verify its conformance to IEC 61499 and O-PAS standards. For this, it was developed a AI-PID-AO system with simple blocks (Figure 4), in a way that all the system's logic is encapsulated inside each block (Figure 5 shows AI block, as an example). For the communication of variables, it was followed O-PAS Part 6.2 that defines a custom data type called O-PAS Signal, which contains information of value, timestamp, status, engineering unit and range, and is responsible for the data exchange between blocks.

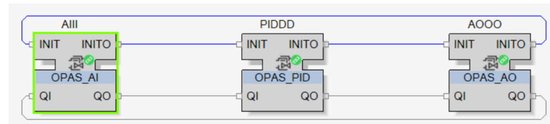

Fig. 4. AI-PID-AO System.

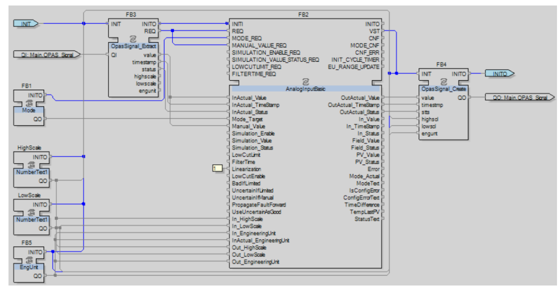

Fig. 5. AI block internal logic.

The three main blocks (namely AnalogInputBasic, PID-ControlBasic and AnalogOutputBasic) were obtained from the Part 6 Technology Working Group repository, responsible for Information and Exchange Models. Each individual block has its own Execution Control Chart in the form of a state machine (Figure 6 shows the ECC of PID block, as an example) as well as its algorithms (in Structured Text), which defines the block's behavior.

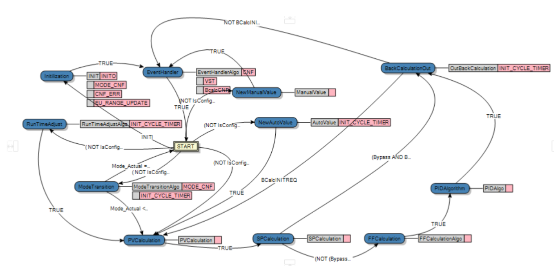

Fig. 6. PID Execution Control Chart.

After the execution of informal tests and demonstrations, it was evaluated its ability to achieve the open system attributes of crossplatform software portability, device configurability and interoperability. To test realtime behaviors of the implementation on runtime, it was developed a graphical interface (Figure 7), that enables variables monitoring and system parameters configuration. It works as Human Machine Interface where the user can have better visualization and control of the system in realtime.

With the development of the application based on 61499, the next goal is to test its interoperability with other applications. To stablish integration with 61499 and 61131 systems, there

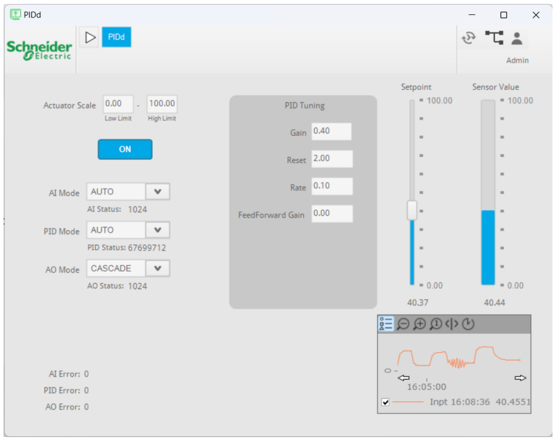

Fig. 7. Graphical interface.

are three possible approaches: using a 61131 based runtime with an 61499 emulation environment to interpret 61499 logic, using a 61499 based runtime with an 61131 emulation environment to interpret 61131 logic or having both runtimes working in parallel with a communication interface in between.

Taking in consideration the third method, it is possible to have both original applications running simultaneously with no need of code transformation or translation. For the development of this work, it was followed the proposed method described in [5], which consists of a software layer based on a shared memory approach that realizes information exchange between the two runtime environments.

For the implementation of the present work, the interface was made following the O-PAS Part 4, which defines a Connectivity Framework to allow interoperability between instances of Distributed Control Nodes (DCNs). Any linkage between blocks in the application and the outside world should be made through an O-PAS Connectivity Function Blocks (OCFBs). The used mechanism for handling the information flows between instances of O-PAS DCNs is the OPC Unified Architecture (OPC UA) communication model.

A server based on the OPC UA architecture was then implemented, whose protocol is platform independent, simplifying industrial connectivity. The OPC UA provides a framework that can be used to represent complex information as nodes in an AddressSpace which can be accessed with standard web services [3]. To acquire the variables present on the server, clients were tested on the EAE platform itself, as well as using the external software UaExpert®.

To better detail the technique in [5] it is defined by one or more devices that are able to execute both IEC 61131 and IEC 61499 code using different execution environments. These runtimes are independent and interact with each other through a communication interface, composed by a set of PLC Data Exchanges (PDE). In an O-PAS view, these PDEs represents OCFBs and they could be of Data Transfer or Procedure Call, differing whether they expect response data or not.

The focus of this work is to create a communication model and test whether the IEC 61499 environment is able to send and receive data from it. In other words, it was intended at this first moment to test data transfer from an IEC 61499 application. For this, the PLC Interface was represented as a Service Interface Function Block, which permits to perform data exchange requests, confirmations, indications or responses. The created block (Figure 8) simulates the communication layer, having encapsulated both PDEs for data transfer from 61499 logic to the server and from the server to the application.

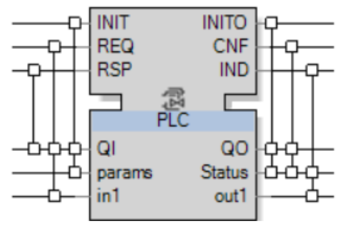

Fig. 8. PLC Block - Interface.

To further understand the PLC block, it is needed to look inside its Function Block Network (Figure 9). The event variables are responsible for invoking the PDEs, depending on the communcation way. The OPCUA_EXPOSE block can be seen as a publisher, exposing the input variables over the OPC UA Server. Similarly, OPCUA_READ acts as a subscriber or client, scanning the defined nodes for any variable change. It can be seen after the READ block that it has an 'OpasSignal_Create block, that gathers all the variables and creates the O-PAS Signal on its structured form. Similarly, there is an 'OpasSignal_Extract block inside the OPCUA_EXPOSE, encharged for the split of O-PAS Signal into independent variables.

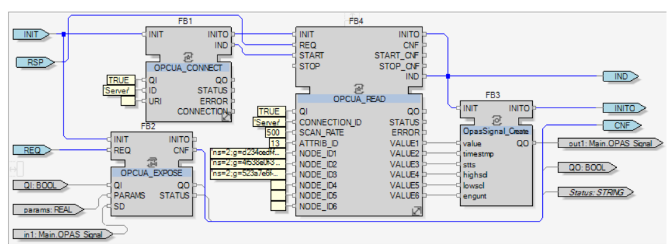

Fig. 9. PLC Block - Function Block Network.

Better describing the implementation of the communication layer, based on OPC UA, the server is automatically created by the EAE runtime, leaving the user the ability to select which variables from the SIFB they want to expose. To read data from the server, the OPCUA_READ is a predefined block, inside the OPCUAClient library, capable of reading any elementary variables from the nodes, after stablished connection with the server. Due to this characteristc of the development tool, it was not possible to communicate the O-PAS Signal completely through the server, since the signal consist on only one variable containing all information inside its attributes.

The attributes of multi-vendor open systems were assured by building the application following the Compliance Profile defined in Part 4 of O-PAS. Through its construction, it was endorsed that EAE successfully achieved the exchange of library elements, device configurations and the ability to work together with other tools. On the other hand, it was found a bottleneck on the transmission of variables using OPC UA, that currently only supports elementary data types. Then, it was not possible to send the structured data type O-PAS Signal completely, instead, it was needed to extract each individual parameter of it and send separately.

With this first part of the study, it can be concluded that the proposed methodology is validated for communication on IEC 61499 based applications using EAE. Regarding the development tool, it has been seen that it still needs some improvements, specially about the OPC UA treatment of variables and objects. Future works includes perfomance tests for evaluating the efficiency of the presented model.

V. Conclusion

Current challenges point to a more distributed control, so that several equipments work together to perform tasks. To achieve this, it is necessary to define standards to guarantee the operation of devices from multiple manufacturers, with different software tools. To ensure market insertion, companies must always be up to date with market trends in the field of industrial automation.

Under the pretext of developing open solutions, IEC 61499 was created to standardize specifications for automated industrial processes to achieve portability, interoperability and configurability between equipment. Among other differences to its predecessor (IEC 61131), solutions developed in accordance with 61499 allow abstraction at a system level, in which it is possible to visualize the entire process, with devices, resources and applications, in addition to the functional blocks that compose them.

In the industrial world, free and open source software contributes to idea sharing and collaboration, contraryly to proprietary systems that act as a barrier to innovation. To reach the adoption of a new standard, the involved community must continuously work on updates and test the feasibility of proposed changes as well as ongoing improvements in Compliance Profiles, including automated compliance tests. The production of teaching and training materials are also essential for increasingly diffusion of intelligent automated systems, serving as a source of technology transfer and consultation to the automation and control market.

In this work, it was analyzed the reception of the new standard across the time, since its release and updates. It was also developed, implemented and validated function blocks, in order to evaluate the degree of maturity of EAE platform, whose access was guaranteed through licenses granted by UniversalAutomation.org, by its Universities Program. Another achievement was the validation of a communication layer based on OPC UA on a 61499 environment. With the execution of this work, it is expected to contribute to the 61499 studies aiming the integration with other standards, such as 61131.

Acknowledgements

This project has been technically and financially supported by Nova SMAR. The authors also would like to thank LAUT and LAMP laboratories at UFRN and the UniversalAutomation.org team for their assistance with licenses and training for use of EcoStruxure Automation Expert.

References

[1] Iec 61131-1 function blocks – part 1: Architecture. Standard, International Electrotechnical Commission, 2012.

[2] Iec 61131 programmable controllers - part 3: Programming languages. Standard, International Electrotechnical Commission, 2012.

[3] O-pas™ standard, part 4 – o-pas connectivity framework (ocf). Standard, The Open Group, 2020.

[4] Don Bartusiak. International society of automation's - virtual process industry conference. 2020.

[5] Stefano Campanelli, Pierfrancesco Foglia, and Cosimo Antonio Prete. Integration of existing iec 61131-3 systems in an iec 61499 distributed solution. In *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, pages 1–8. IEEE, 2012.

[6] James H Christensen, Thomas Strasser, Antonio Valentini, Valeriy Vyatkin, Alois Zoitl, J Chouinard, H Mayer, and A Kopitar. The iec 61499 function block standard: Software tools and runtime platforms. *ISA Automation Week*, 2012, 2012.

[7] Wenbin Dai and Valeriy Vyatkin. Ontology model for migration from iec 61131-3 plc to iec 61499 function block. In *2011 Sixth IEEE International Symposium on Electronic Design, Test and Application*, pages 172–175. IEEE, 2011.

[8] William Wenbin Dai and Valeriy Vyatkin. A case study on migration from iec 61131 plc to iec 61499 function block control. In *2009 7th IEEE International Conference on Industrial Informatics*, pages 79–84. IEEE, 2009.

[9] Peter Gsellmann, Martin Melik-Merkumians, Alois Zoitl, and Georg Schitter. A novel approach for integrating iec 61131-3 engineering and execution into iec 61499. *IEEE Transactions on Industrial Informatics*, 17(8):5411–5418, 2020.

[10] M Melik Merkumians, Peter Gsellmann, and Georg Schitter. Hierarchization and integration of iec 61131-3 and iec 61499 for enhanced reusability. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4. IEEE, 2021.

[11] Rosslyn. Programmable controllers part 1: General information. Standard, National Electrical Manufacturers Association, 2005.

[12] Maryam Sadeghi. Developing iec61499 in industrial processes, measurement and control systems (ipmcs). *WSEAS Transactions on Systems and Control*, 5(04), 2010.

[13] Kleanthis Thramboulidis. Iec 61499: Back to the well proven practice of iec 61131? In *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, pages 1–8. IEEE, 2012.

[14] Kleanthis Thramboulidis. Iec 61499 vs. 61131: A comparison based on misperceptions. *arXiv preprint arXiv:1303.4761*, 2013.

[15] Rosana Trevisan. Michaelis dicionário brasileiro da língua portuguesa. *São Paulo: Melhoramentos*, 2015.

[16] Yang Wei. Implementation of iec61499 distributed function block architecture for industrial measurement and control systems (ipmcs). *degree thesis, National Univ. Singapore*, 2002, 2001.

[17] Monika Wenger and Alois Zoitl. Re-use of iec 61131-3 structured text for iec 61499. In *2012 ieee international conference on industrial technology*, pages 78–83. IEEE, 2012.

[18] Monika Wenger, Alois Zoitl, Christoph Sunder, and Heinrich Steininger. Transformation of iec 61131-3 to iec 61499 based on a model driven development approach. In *2009 7th IEEE International Conference on Industrial Informatics*, pages 715–720. IEEE, 2009.

[19] Alois Zoitl, Thomas Strasser, Christoph Sunder, and Thomas Baier. Is iec 61499 in harmony with iec 61131-3? *IEEE Industrial Electronics Magazine*, 3(4):49–55, 2009.