Computers in Industry 72 (2015) 92–102

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

A cyber–physical system-based approach for industrial automation systems

Kleanthis Thramboulidis *

Electrical and Computer Engineering, University of Patras, Greece

ARTICLE INFO

Article history:
Received 15 November 2014
Received in revised form 23 April 2015
Accepted 28 April 2015
Available online 16 May 2015

Keywords:
Industrial automation systems
Cyber–physical systems
System-based approach
Mechatronics
UML/SysML
IEC 61131
Java
IoT

ABSTRACT

Industrial automation systems (IASs) are commonly developed using the languages defined by the IEC 61131 standard and are executed on programmable logic controllers (PLCs). Their software part is commonly considered only after the development and integration of mechanics and electronics. However, this approach narrows the solution space for software; thus, it is considered inadequate to address the complexity of today's systems. In this paper, we adopt a system-based approach for the development of IASs. Based on this, the UML model of the software part of the system is extracted from the SysML system model and it is then refined to get the implementation code. Two implementation alternatives are considered to exploit both PLCs and the recent deluge of embedded boards in the market. For PLC targets, the new version of IEC 61131 that supports object-orientation is adopted, while Java is used for embedded boards. The case study used to illustrate our approach was developed as a lab exercise, which aims to introduce to students a number of technologies used to address challenges in the domain of cyber–physical systems and highlights the role of the Internet of Things (IoT) as a glue for their cyber interfaces.

1. Introduction

Industrial automation systems (IASs) are composed of the physical plant, which performs the physical processes, and networks of embedded computers, which perform the computational processes required to monitor and control the physical ones. The cyber part of the system is constituted by computational processes, which receive inputs from the physical processes, calculate the required outputs and apply them to the physical plant. This is usually realized using time triggered control in the form of the well known scan cycle paradigm.

Computational processes are commonly implemented based on the de-facto standard IEC 61131, which defines a set of languages for programming on programmable logic controllers (PLCs) [1]. This standard has been around for at least 20 years and is attributed the introduction of basic concepts of object orientation through the construct of function block (FB) in the domain of industrial automation [2]. However, it is of question whether this technology is able to address the new requirements of today's industrial automation systems [3]. This is primarily due to their increasing complexity and the need for flexibility. In particular, these requirements include among others distribution, portability, configurability, interoperability and reconfiguration, which have all been identified as the high-level demands/requirements for future automation systems [4].

In order to address the restrictions imposed by version 2.0 of IEC 61131, as well as to address the new challenges in the development of today's complex industrial automation systems, the IEC has defined the IEC 61499 standard [5]. This standard "has emerged in response to the technological limitations encountered in the currently dominating standard IEC 61131", as claimed in [6], where IEC 61131 is characterized as "severely inadequate to meet the current industry demands for distributed, flexible automation systems." The IEC 61499 has been widely accepted by the academic community; a big number of publications have been produced and a debate on pros and cons is active [7,8]. Interestingly, the standard has not been accepted by the industry [3] owing to a number of reasons including the absence of support by the currently dominating tools and environments in industry and the absence of a variety of new mature tools and run-time environments to support the new standard. The lack of "mature engineering tools, reliable embedded control hardware, proven

* Tel.: +30 2610 996436.
E-mail address: thrambo@ece.upatras.gr

http://dx.doi.org/10.1016/j.compind.2015.04.006
0166-3615/

design methodologies, and trained engineers" is considered in [9] as the main barrier that prevents practitioners from using the IEC 61499.

On the other side, the IEC 61131 has been recently upgraded with a new version [10]. Version 3.0 provides support for the object-oriented (OO) paradigm. CoDeSys 3 [11] has already implemented an object oriented version of the IEC 61131 and other industrial vendors such as Beckhoff [12] are moving toward this direction. However, programming in an object oriented way is not a trivial task for industrial automation developers, who are already accustomed with version 2.0 of the IEC 61131. Thus, shifting to a complete object oriented approach will require long time. This transition will also be pushed by new developers, who enter the field, especially since object oriented programming and UML/SysML is already included in the curricula of both universities and technical schools, e.g., object oriented programming is now mandatory in the curriculum for technicians in Germany [13].

Apart from training, specific frameworks may facilitate this transition and are expected to bring the benefits of applying the object oriented paradigm and Model Driven Engineering [14,15] in the industrial automation domain. These challenges have already attracted significant research interest and numerous papers have been published toward this direction [16–18] to name a few. It is by now widely accepted that the factory automation industry is slowly but steadily experiencing a paradigm shift [19]; there is an increasing adoption of technologies and standards of business software systems in the control level of the industrial automation systems [20]. Moreover, there is a shift from the traditional device-centric programming paradigm adopted by the IEC 61131 to the application-centric one that utilizes mature state-of-the-art technologies for general purpose computing. These technologies are currently integrated into a coherent set for industrial automation systems in paradigms such as the Industrial Internet of Things or Industry 4.0 [21].

At the same time, there is a lot of criticism regarding the traditional development process of industrial automation systems, e.g., [22–24]. According to this traditional approach the constituent parts of the systems, i.e., mechanics, electronics and software, are developed independently and are only then integrated to compose the system. This is considered to be inadequate to address the always increasing complexity of these systems.

In this paper, we adopt a synergistic integration of the various disciplines involved in Mechatronic systems at the component level, thus aiming to address the new challenges in industrial automation systems. The Mechatronic component (MTC), which consists of mechanics, electronics and software, is considered the key construct for the composition of Mechatronic systems. Based on this, a cyber–physical system-based approach is described in this paper. Our approach exploits composability and compositionality [25] on the Mechatronic or cyber–physical component-levels and is used to define the system level functional and non-functional properties by applying synergistic integration at the cyber–physical component level. It adopts the object-oriented paradigm, and, exploits SysML for system level modeling and UML for modeling the software part of the system.

The main focus of this paper is on modeling the cyber part of the system with an emphasis on the software part. Two alternative implementations of the proposed design are discussed. One is based on the IEC 61131 3.0 for PLCs; the other is based on a general purpose object-oriented programming language to allow the use of the various embedded boards mainly based on ARM processors that have appeared in the market recently. The entire design and the prototype implementation are presented in the context of an educational approach that is designed for both students and industrial practitioners. We emphasize on the use of higher layers of abstractions and on the differences between the two implementation alternatives so as to successfully: (a) realize the shift toward the object-oriented paradigm, and, (b) apply the model driven development (MDD) paradigm. Moreover, we have been particularly careful to clarify the scan cycle model, which is widely used in PLC programming, given that the users of embedded boards are typically accustomed to the event triggered programming paradigm. The objective is for the framework to support (i) scheduling abstraction [26], which would enable the developer to neglect the scheduling of components, and, (ii) timing abstraction, which would allow the developer to neglect timing issues and focus on causality instead.

In order to illustrate all the steps of our approach starting from the system level and moving down to the executable code we utilize the Liqueur Plant system, which was first used in [3]. The model of the cyber part, which is derived from the system level model, is refined, following the proposed approach, for: (a) the cyber parts of cyber–physical components, and (b) the process controllers, which are modeled as cyber components. The layered architectural style is adopted for structuring the cyber–physical component, while the REST network-based architectural style is adopted for the integration of system level components regarding their cyber interfaces. Alternatively, for the case in which a tighter coupling between system components is not prohibitive, SOAP [20] or even a traditional middleware technology such as CORBA may be used. The software part of the prototype implementation was implemented using Java and the scan cycle model, and, it was tested with a simulator of the physical plant. Java is considered a technology that may speed up the adoption of advances in general purpose computing in the domain of industrial automation systems [27].

The remainder of the paper is organized as follows. Section 2 further discusses relevant work. In Section 3, the case study used is described and the fundamental concept of the system-based approach is briefly presented. Section 4 describes the proposed in this paper structure for the cyber part of the system. In Section 5, two implementation alternatives are discussed and finally the paper concludes in Section 6.

2. Related work

Progress in general purpose computing has attracted the interest of both industry and academia from the domain of industrial automation systems. Several approaches that exploit the new technologies have been proposed so far; object and service orientation, component based and model driven development are among the ones that have been extensively promoted since they have been able to improve the development process in this domain.

Object-orientation (OO) has already attracted the interest of the research community and various approaches have been proposed on how to exploit OO in this domain, e.g., [28,29]. However, as is claimed in [30], most of these do not take into account the OO aspects of the IEC 61131 in the direction of extending it to support the object oriented paradigm. This has resulted to inefficient proposals regarding the object-oriented extension of the IEC 61131 model.

Service oriented architectures (SOA) have already attracted the interest of researchers in the industrial automation domain, e.g., [19,31–34] and vendors are already moving toward exploiting the Interent of Things (IoT), e.g., [35]. To support discovery and composition of capabilities of entities that constitute cyber physical systems and their just-in-time assembly, authors in [36] describe an approach to enable the use of SOA methods for this domain.

UML and SysML are both utilized in industrial automation systems, e.g., [16–18,25,38]. SysML was defined as an extension of UML for system modeling and is replacing UML last years in mechatronic systems modeling, e.g., [24,37]. It may be used for