INVITED PAPER

Engineering Methods and Tools for Cyber–Physical Automation Systems

This paper considers the industrial context for the engineering of CPSs. It reviews engineering approaches that have been proposed or adopted to date and introduces a component-based CPS engineering toolset.

By Robert Harrison, Daniel Vera, and Bilal Ahmad, *Member IEEE*

ABSTRACT | Much has been published about potential benefits of the adoption of cyber-physical systems (CPSs) in manufacturing industry. However, less has been said about how such automation systems might be effectively configured and supported through their lifecycles and how application modeling, visualization, and reuse of such systems might be best achieved. It is vitally important to be able to incorporate support for engineering best practice while at the same time exploiting the potential that CPS has to offer in an automation systems setting. This paper considers the industrial context for the engineering of CPS. It reviews engineering approaches that have been proposed or adopted to date including Industry 4.0 and provides examples of engineering methods and tools that are currently available. The paper then focuses on the CPS engineering toolset being developed by the Automation Systems Group (ASG) in the Warwick Manufacturing Group (WMG), University of Warwick, Coventry, U.K. and explains via an industrial case study how such a component-based engineering toolset can support an integrated approach to the virtual and physical engineering of automation systems through their lifecycle via a method that enables multiple vendors' equipment to be effectively integrated and provides support for the specification, validation, and use of such systems across the supply chain, e.g., between end users and system integrators.

Manuscript received August 13, 2015; revised December 1, 2015; accepted December 3, 2015. Date of publication March 21, 2016; date of current version April 19, 2016. This work was supported by the U.K. Engineering and Physical Sciences Research Council (EPSRC), through the Knowledge-Driven Configurable Manufacturing (KDCM) research project under the Flexible and Reconfigurable Manufacturing Initiative, from Innovate UK on the Direct Digital Deployment project, and from ARTEMIS on the Arrowhead project.
The authors are with the Warwick Manufacturing Group (WMG), University of Warwick, Coventry CV4 7AL, U.K. (e-mail: robert.harrison@warwick.ac.uk; d.a.vera@warwick.ac.uk; b.ahmad @warwick.ac.uk).

Digital Object Identifier: 10.1109/JPROC.2015.2510665

KEYWORDS | Automation; cyber-physical systems (CPSs); engineering; lifecycle; manufacturing; methods

I. INTRODUCTION

Cyber–physical systems (CPSs) are distributed, heterogeneous systems connected via networks, and usually associated with the concept of the Internet of Things (IoT) [1]. The increasing availability and use of distributed industrial CPS devices and systems could radically change the nature of manufacturing and provide new opportunities to develop more effective, finer grained, and self-configuring automation systems; the context for CPS in this paper. A closely associated initiative is the Industry 4.0 platform, a specialization within the IoT and the Internet of Services, it facilitates the vision of the smart factory where CPSs monitor physical processes, create a virtual copy of the physical world, and make decentralized decisions [2].

Realizing CPSs for industrial automation implies the need for engineering tools capable of supporting distributed systems and is coupled to a major shift in emphasis from traditional monolithic, specialism-based, isolated engineering tools and methods toward integrated, cloud-based tool/system infrastructures based around an Internet of Services and associated data. CPSs also imply a combination of physical and virtual representations where physical devices and functionality are represented in data form and can be visualized virtually with the data model maintained in correspondence to the physical system throughout their lifecycle.

As noted by Aicher *et al.* [3], for many years the complexity and the number of components in production automation systems have been increasing. Additionally, the

size and complexity of the embedded software inside the components are also increasing rapidly. Hence, the effort and the cost for software engineering of production automation systems are consequently rising. Tools should manage engineering complexity and are at the heart of the engineering process. It is important to ensure that current good practice and understanding of necessary engineering workflows and functional capabilities can be transferred and embodied into CPSs and that the required new tools and methods can support this.

Current automation systems engineering methods are frequently criticized, e.g., for poorly supporting reuse and their inability to effectively validate automation solutions across supply chains. There is also poor integration between real system and virtual system representations, which need to be closely integrated throughout the automation system lifecycle from specification and design through commissioning, validation, operation, and reuse of systems.

Engineering tools traditionally have evolved to support the principle of "separation of concerns" to manage engineering complexity. Therefore, tools are typically vertically integrated with limited support (even intention) for horizontal integrability (i.e., integration across disciplinary boundaries).

II. REVIEW OF CPS ENGINEERING METHODS AND TOOLS

A. Utilization of Existing Tools and Standards

The majority of current tools are vendor specific and support largely closed control environments. They generally offer good point-solution functionality, are well supported, and can deliver robust operational systems, be it with limited agility.

In a CPS context it should be noted that within Industry 4.0 the technical working group WG2 is studying existing approaches and methods and has identified a series of usable, approaches. However as a rule these only address partial aspects of the holistic view of Industry 4.0 [2]. For end-to-end engineering, AutomationML, ProSTEP iViP, and eCl@ss were considered of most interest; for communication, OPC-UA-based IEC 62541; in the information layer, electronic device description (EDD) and field device tool (FDT); and in the approach for implementation of a functional and information layer, field device integration (FDI) as the integration technology.

CPSs need to utilize and maintain compatibility with traditional standards, where appropriate, and initiate the definition of new ones where necessary. Significant standards from an engineering perspective include IEC62264 enterprise control system integration, ISA Draft 88/95 technical report, using ISA88 and ISA95 together, IEC 62890 lifecycle and value stream, IEC 62264/IEC 61512 hierarchy levels. The reference architecture model RAMI4.0 has been put forward for standardization as DIN SPEC 91345. See Fig. 1.

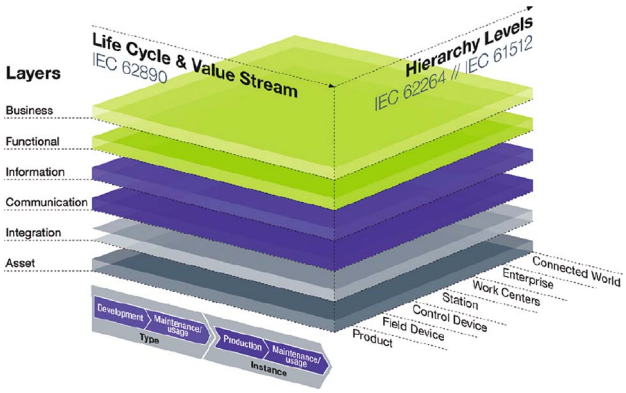


Fig. 1. ***Industry 4.0 reference architecture. Source ZVEI [2].***

B. Component- and Model-Based Systems

An overview of architecture frameworks and modeling languages is introduced by Paredis *et al.* [4]. There are many modeling approaches based on UML or SysML [5]–[7]. In some of the literature, a module is defined as a mechatronic component and the associated software including parameters. An approach for modeling mechatronic components structurally based on practical experiences is introduced by Luder *et al.* [8]–[10]. The approach was implemented with Siemens SIMATIC Automation Designer featuring the instantiation of parameters based on a library of mechatronic components.

In a component- and model-based design flow, system models are composed of component models guided by architectural specifications, which may form a reference architecture defining system scope, content, and composition. A reference architecture is a general model that provides a framework for the structuring, development, integration, and operation of the systems [11]. For example, the Industry 4.0 Reference Architecture model is very broad in that it aims to permit homogeneous consideration of the product to be manufactured and the production facility, with their interdependencies. The adoption of a component-based approach is at the heart of the recently published RAMI reference model of Industry 4.0, which features logically nestable components [2].

Model-based design has a proven track record and strong acceptance in many focused areas of engineering. A range of approaches have been pursued to model-based development; they may be descriptive and system independent at the initial modeling stage, or be target-system dependent from the outset [12]. Component-based development has been proposed by various research groups to increase flexibility and effectiveness of the development process [13], [14]. Vogel-Heuser *et al.* have developed a

nonformal methodology in the FAVA project for the systematic design of distributed automation systems in the domain of manufacturing [15]. The authors of this paper have also pioneered component-based systems founded around a common model [16]–[20] and discussed in Section III.

In the context of a component-based approach to CPSs, an important goal is the adoption of higher value design activities, e.g., the adoption of component- and model-based design in order to lay the foundation for establishing well-defined interfaces between design and deployment and across all lifecycle phases [21]. The appeal of component-based design is the potentially large productivity increase due to the reuse of design knowledge captured by the component models used.

An industrial CPS might typically consist of many components of widely varying scale and complexity, from individual sensors and mechatronic devices to complex control subsystems. The effective engineering of such industrial systems thus requires a new approach to enable such components to be combined and logically configured into evolvable automation systems whose behavior can be well specified, implemented, and validated. The many logical components (or objects) of such automation systems may be clustered on some larger physical control devices, e.g., programmable logic controller (PLC), or reside more individually on separate networked embedded devices. These sets of control devices, large or small, are networked together to form a CPS enabling new forms of logical control interaction, information collection and aggregation, and dynamic (re)configuration.

C. Component Programming and Deployment

Deployable CPS components (see Section III-C for details on vueOne component model) may be engineered in a range of forms. The engineering of distributed embedded systems requires the modeling and support of units of distributed functionality. A distributed system can be described as a composition of interacting components, such as function blocks or port-based objects, which are mapped onto distributed devices [22].

As reported by Fay *et al.* [23], a distributed system nodes, i.e., distributed components, may have different properties. Each node allows either the execution of parameterized predefined functions (in the form of a set of parameterized FBs) or free programming of control applications based on FBs, programming, and cyclic execution being based on the IEC61131 standard. Such manufacturing automation system configurations are widely used in the manufacturing industry [23]. This approach is also used by the authors in their vueOne system where a state-based design method is adopted for sequence and interlock behavioral definition and execution incorporating standard IEC 61131 notation. Note that in vueOne where clusters of components are local to a single physical node (e.g., on the same PLC) local FB state arrays are used in a data-model-driven approach. See Sections III and IV, with Section III-D providing an outline definition of the component-based data-model-driven approach and its runtime deployment.

Thus, systems design may be achieved directly in IEC 61131 (on a per component basis), via IEC 64199 or through a higher level of abstraction. In the vueOne system, the authors use a component-based system-level design environment. Component functionality is described using state-transition diagrams for process and component behaviors and through the propagation of states between components [16], [17], [24], [25]. IEC 61499 [26], [27] promises significant advantages to users in terms of simplicity of system level design and, more importantly, the possibility of distributing control programs across distributed hardware and throughout the entire network, but its utilization currently remains limited [26].

D. SOA Engineering Methods and CPS Communications

CPSs involve highly networked system structures incorporating large numbers of human beings, IT systems, and automation components. Industrial communication systems, such as a suitable fieldbus or industrial Ethernet, are available to establish communication among the distributed components. In the vision of CPS every component (be it for control, business, or engineering functionality) is potentially able to interact with any other component, as illustrated in Fig. 2 [28]. This enables a strong horizontal and vertical integration within automation systems. For realizing such an automation network, the question arises of how this communication is provided with high interoperability [29].

The factory of the future will be heavily based on Internet and web technologies and as devices become able to natively offer web services, they will provide an interoperability layer that leads to easier coupling with other components despite of the high heterogeneity. Device

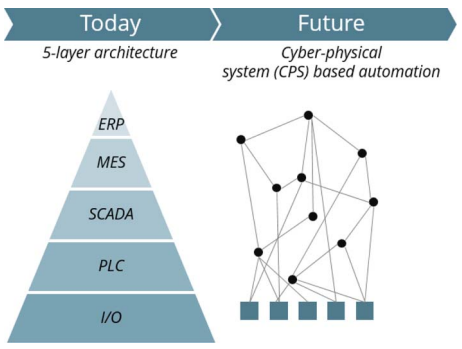


Fig. 2. ***Future distributed CPS functionality. The squares represent input/output (I/O) devices, the lines service interactions, and the circles distributed functionality. Source IoT analytics [28].***

profile for web services (DPWS) [30], OPC-UA [31], and representational state transfer (REST) are three of the emerging technologies for realizing web service-enabled device connectivity [32]–[34]. Such approaches are now being heavily utilized in CPSs by the authors and many other groups [35], [36].

The SOCRADES, SODA, and IMC-AESOP [37]–[39] projects investigated taking web services down to the device level to enable improved horizontal and vertical shop floor integration [22], [40]–[42]. It should also be noted that a whole new range of critical security issues arise in the context of communication for CPSs that are beyond the scope of this paper [11].

E. Interoperability and Information Models

Data exchange among mechanical plant engineering, electrical design, process engineering, process control engineering, human–machine interface (HMI) development, PLC programming, and other engineering tools is currently difficult. Furthermore, from a lifecycle perspective, there are significant gaps in the consistent use of information throughout the lifecycle phases, and although the concept of the smart factory is well understood where consistency between physical and virtual representations of production systems would be maintained at all lifecycle phases, in reality a largely fragmented information modeling environment currently exists.

Prösser *et al.* [43] report that one of the main barriers impeding the implementation of existing integration solutions relates to the mutually diverse requirements of the mechatronics disciplines involved. In order to overcome these barriers, a middle-in data modeling strategy to fulfil multidisciplinary requirements and to maintain data model consistency is proposed. The approach has been applied within the engineering design process concerning body shop production lines at Audi.

AutomationML (AML) is a neutral data format based on XML for the storage and exchange of plant engineering information modules, to interconnect heterogeneous engineering tools from different fields, e.g., mechanical plant engineering, electrical design, and PLC systems. AML is essentially combination of existing standard data formats for the storage of different aspects of engineering information. For example, CAEX is used as the top level format that connects the different data formats to comprise the plant topology, COLLADA is used for storage of geometric and kinematic information, and PLCopen XML serves for the storage of control logic. Following the publication of the AML standard IEC 62714, the implementation of a consistent engineering information model with AML has to be verified [44]. Aicher reports on a project where consistent verified engineering information has been implemented using AML [3].

Biffl *et al.* [45] introduce a model-driven engineering (MDE) approach for developing and providing versioning and linking support for AML. The MDE process builds on an AML metamodel to derive tool support 1) for representing sets of local engineering results as data elements in AML files, 2) for extracting AML data elements for versioning the local engineering results, and 3) for linking the versioned local engineering results to an AML tree representing the overall plant structure.

F. Simulation and Modeling for CPS

The concept of the virtual factory introduced by Westkämper and Jendoubi continuously integrates data from the real factory and in this respect is a predecessor of CPSs [46]. The realization of this concept necessitates the organization and coordination of interactions between virtual and real machines, plant control systems, and production management systems [11].

There is currently no integrated tool chain that supports design and simulation of automation systems from the first virtual prototype model to the fully functional model. A wide range of computer-aided design (CAD) tools supporting the various facets of the automation lifecycle are however available from many vendors including Siemens PLM and Delmia, e.g., supporting process planning, machine visualization, layout, and cycle-time assessment [47], [48]. As reported by Oppelt *et al.* [49], for a narrower scope, such as for virtual commissioning, some effort is made to extend standards like AutomationML with simulation-relevant information [50], [51]. Typical simulation and computer-aided engineering (CAE) tools that enable engineers to virtually commission the control systems are WinMOD, Emulate3D, TarakosVR, and Experior [52]. Hoffmann *et al.* reviewed existing tool chains for virtual commissioning of manufacturing systems [53].

In the context of the lifecycle of industrial CPS cosimulation is likely to be significant. In cosimulation the different subsystems which form a coupled problem are modeled and simulated in a distributed manner [9], [49].

III. ASG CPS ENGINEERING ENVIRONMENT: VUEONE

The ASG is focusing on the design and implementation of automation systems engineering tools and methods, aligned to the specific nature of CPSs, which can contribute to achieving the goals of Industry 4.0. This is being achieved via the KDCM, Arrowhead, and 3Deployment projects.

The ASG's research is delivering an engineering software environment, called vueOne, part of which is currently being used to support Ford's virtual engineering activity in powertrain assembly in the United Kingdom. vueOne is now also being applied to support the engineering of battery and electric motor make-like-production systems in partnership with industry.

The vision of Kagermann *et al.* for Industry 4.0 is of embedded manufacturing systems that are vertically

networked with business processes within factories and enterprises and horizontally connected to dispersed value networks that can be managed in real time, and that both enable and require end-to-end engineering across the entire value chain [11]. Fulfilling this vision, from an automation systems perspective, requires new engineering approaches and tools capable of enabling seamless system design, implementation, and lifecycle support, implicitly providing new levels of connectivity and data visibility. In relation to this, the vueOne engineering environment is lifecycle based and aims to support the necessary functionality and information interchange 1) between lifecycle phases; 2) between the cyber and physical; and 3) between engineers across lifecycle phases within and across organizations.

The vueOne environment is extendable so that modules can be progressively developed and added to the framework. Regarding technology readiness, with reference to Fig. 4, the common data model and editing/simulation tools exist at TRL 8, having been utilized by the first commercial customers. The deployment and runtime modules are at TRL 5 to 6, having been validated and demonstrated industrially. The server infrastructure, and fault and maintenance trackers are at TRL 4 applied in small-scale prototypes, with other vueOne functionality in development from TRL 3.

A. CPS Engineering Environment Framework

Fig. 3 provides a descriptive framework for introducing the ASG's research. Fig. 3 encompasses 1) the general architecture of system engineering environments and tools; 2) the main functionalities required to support system engineering throughout; and 3) the key phases of automation systems lifecycle. The representation uses a classical layered representation [2], placing the network of entities involved in the design of MS (e.g., supply chain organization, engineering departments/individual engineers, etc.) at the top level. System design (left part of Fig. 3) stakeholders feed data models describing specific aspects of automation systems (e.g., mechanical, control, layout, automated/manual processes, and products) through the use of domain or user-specific views (i.e., data representations) associated with specific engineering tools or services. Each data model is then deployed at physical level using specific processes and software tools, e.g., mechanical build and plant layout, control node programming and/or production systems' configuration.

During the operational phase of CPS lifecycle (right side of Fig. 3), information is generated by the system (i.e., by control systems, sensors, and other monitoring or data acquisition devices) and fed back 1) to the system itself (e.g., machine-to-machine control and communication [2]) or 2) stored and consumed at higher levels of the engineering environment supporting production analysis and management (e.g., SCADA systems), resources planning (e.g., ERP and associated data storage and management systems). The secondary purpose of Fig. 3 is to provide a summary view of engineering gaps identified (Section II) as being critical in developing environments that can be more effective in supporting Industry 4.0 concepts in general and CPS engineering in particular; mainly, integration gaps between 1) data models; 2) engineering software environments; and 3) communication/collaboration between engineering organizations involved through the lifecycle. Most importantly, the lifecycle of current production system is bisected into two completely decoupled phases (i.e., design and operation). The following sections provide a brief overview on how the engineering solution developed by the ASG aim at closing those gaps in order to more fully realize the concept of CPS and Industry 4.0 engineering environment.

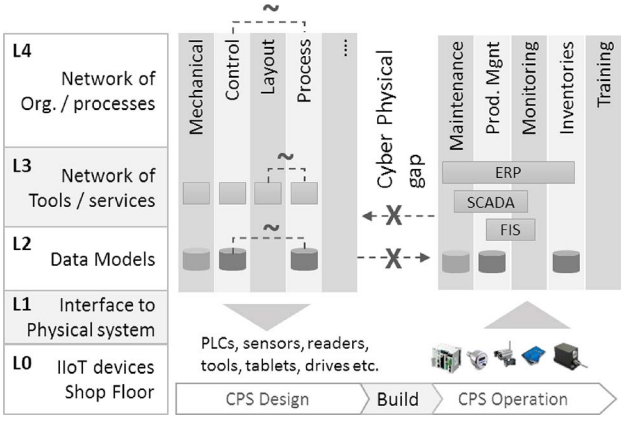


Fig. 3. ***Industry 4.0 oriented descriptive framework for CPS engineering environment.***

B. vueOne Engineering Environment

Using the descriptive framework presented in Fig. 3, an overview of the elements that compose the vueOne CPS engineering environment is provided in Fig. 4. The origin of vueOne was as an automation process and control engineering tool. Its functionalities were extended to broader virtual process planning (i.e., human operation, industrial robot processes modeling), simulations and analysis capabilities (i.e., energy consumption, complexity rating, fatigue rating) with the purpose of providing a collaborative engineering platform. The vueOne environment is a lightweight, low-cost, and easily deployable solution, which aims at 1) providing the supply chain, including SMEs, with access to virtual engineering (VE) capabilities; and 2) complement rather than replace commercially available VE solutions. As such it is built on standards or open data formats and implements interfaces for import/export of data to and from other engineering environments (e.g., X3D and PLCopen XML).

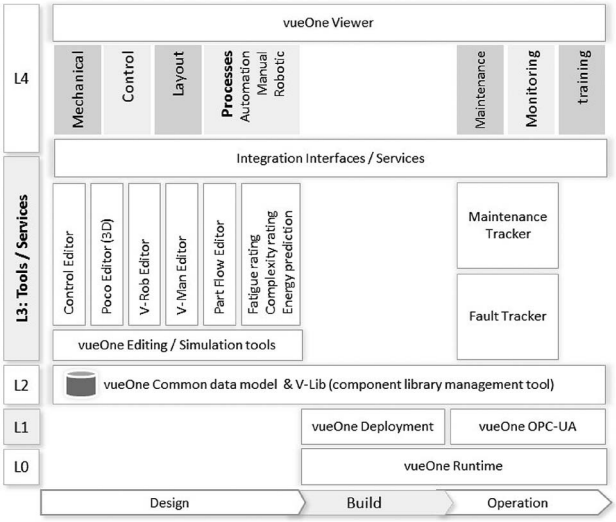


Fig. 4. ***vueOne engineering environment overview.***

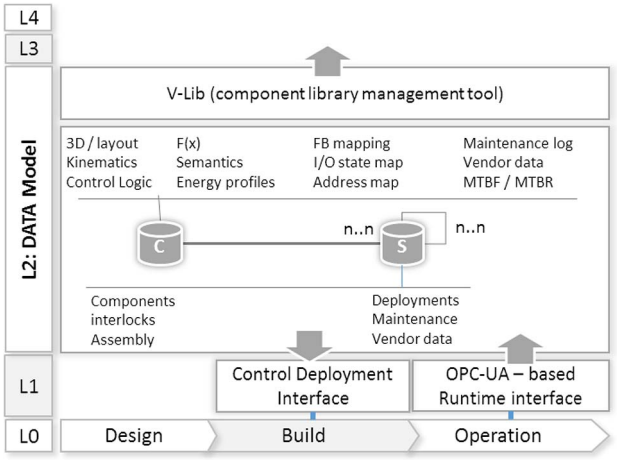


Fig. 5. ***Component-based data model for CPS lifecycle support.***

Sharing of models and simulations is achieved through the use of a viewer application. A key element of the vueOne environment is the common component data model that extends through the whole system lifecycle and provides a mean to close the gap that exists between the design and operation phases of CPSs. The component data model is detailed in Section III-C. The link between the design phase and the operational phase supports component instantiation via automatic PLC control code generation and deployment capabilities and the use of a PLC runtime architecture. The runtime interface allows data to be collected from the physical system components and mapped back onto the digital (or cyber) system data set. More details are provided in Section III-D.

C. CPS Data Model

A large amount of data is generated and handled throughout the system lifecycle; during design, information describing various aspects of the future system is produced and stored in the form of digital data (e.g. CAD, control, process, etc.). During the operation phase, of CPS in particular, a large amount of data is generated by the system itself (e.g., control state information, sensor and monitoring device readings, production data, etc.). A major enabler in achieving the vision and goals of Industry 4.0 is to enable seamless linkage of data sets across both vertical and horizontal dimensions of Fig. 3; a core aspect of the ASG approach in achieving this goal is centred around a common component data model (see Fig. 5) that provides a structured framework for storing both system design data and operational data.

The concept of the component ("C" in Fig. 5) defines a level of data granularity that practically translates into a reusable data set describing a subset of a system. Reusability is better understood in system engineering as system modularity or commonality [54], [55], and aims at encapsulating the description or engineering knowledge associated with part of a system in order to facilitate its subsequent redesign or reconfiguration. The concept of modularity is supported to some extent by existing engineering environments, in specific engineering domains. Data management systems are in some instances used to link data sets across engineering domains, e.g., Autodesk Vault ane Dassault Enovia are example databases that provide process, product, and resource PPR relation management [56]. However, the use of consistent data models across domains is typically not reinforced through consistent engineering tools, methods, and processes, which result in disparate level of modularity or reusability across subsystems.

The vueOne component data model is used during design as a common and structured repository of design data. The core vueOne data set aims at supporting virtual process planning and validation, and therefore consists of the mechanical data (CAD/3D, kinematics), and a description of the control logic in the form of logical states and transition elements represented as IEC 61131 compliant state-transition diagrams. Components can be edited by the user using the vueOne component editor module (e.g., custom machinery component including actuators and sensors) and predefined components for specific manufacturing resources such as human worker (vMan module) and industrial 6R robots (vRob library) have also been developed in order to extend the vueOne tools capabilities (see Fig. 4).

The use of the component data model across the whole system lifecycle (i.e., from design to operation) is reinforced through the vLib module providing access and management of component data. This aspect of the developed engineering environment is critical in closing the gap between the digital system representation and

the real system; the component data model enables data generated by CPS's component (e.g., energy monitors, controller data, and sensors) to be collected in a structured way and mapped back to the design data in order to use data collected during operation for refining initial design components (e.g., component energy consumption data, actuation time/speed, cycle time, and control sequence) and make design data (e.g., 3-D models and design control layout) directly usable to support system operation and related functionalities (e.g., remote monitoring, maintenance, and training).

The component data model is complemented by a system data model that defines the system architecture and system-level data required for consistently managing the composition of components. Examples of system-level data are assembly information, sequence of operations, and interlock definitions.

D. CPS Environment Interfaces

Industry 4.0 and other CPS paradigms rely largely on seamless connectivity and interfacing between organizations and people, computational resources (e.g., databases and application servers), software environments, shop-floor level devices, sensors, and mobile devices [11]. The vueOne engineering environment design is based on three types of conceptually defined interfaces: 1) organization level interfaces enabling communication of information between engineers within or across organizations; 2) software level connectivity allowing exchange of data between engineering tools; and 3) cyber to physical environment connectivity which focus on the configuration and connectivity with components on PLC-based control or embedded devices.

PLCs are widely used in industry to build automation systems. A direct link can be established between the digital representation of a production system and the physical system itself to enable it to be tested virtually, e.g., via virtual commissioning and hardware in the loop [57]–[59]. The left side of Fig. 6 describes the vueOne control software deployment tool (composed of deployment and mapper modules) that was implemented by the ASG to enable 1) automatic generation of PLC control code from the state-transition-based component logic defined using the vueOne editing tools, and 2) deployment of this component control code to the target PLC(s).

The vueOne deployment module was developed with the objective of maintaining consistency between the digital (or cyber) system representation and the real control system configuration. This is achieved through key functionalities. 1) An exact copy of both components and system control definition, edited using the vueOne editor and described as logical states and transition conditions, is installed on the PLC as a data block. 2) This data block containing the definition of the system control logic is scanned and updated by a logic engine that supports one or more vueOne components per PLC or embedded

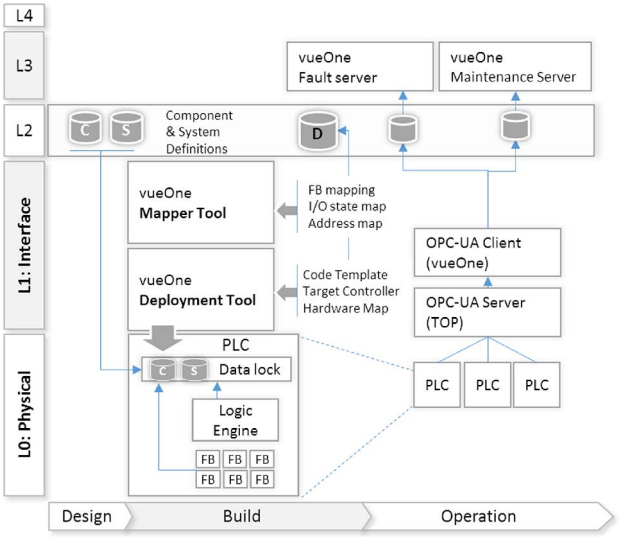


Fig. 6. ***vueOne physical system interfaces.***

controller. Component-state values are propagated between devices (e.g., multiple PLCs) where the control is distributed. The mapper module allows mapping between component, function blocks, I/O and memory addresses, and the storage and version management of the mapping information referred to as a deployment entry. The vueOne deployment module can generate code for controllers from various vendors (i.e., currently for Siemens, Schneider, and PLCopen, plus embedded device support) [60], [61]. The deployment capabilities include automatic HMI-screen generation for machine-specific screens, and support for multiple modes of operation (automatic, manual) and cycle types (continuous, single, step-by-step cycles, and dry run) [62]. The deployment tool also allows the PLC code to be formatted according to specific requirements; for instance, extensive development was made in partnership with Ford UK to generate code compliant with Ford's global PLC programming standard FAST [63].

This approach creates a solid and consistent link, or interface between the virtual (or cyber) environment and the physical system (both logic information and runtime system are consistent) and allows control logic to be deployed automatically to PLC controllers. In addition, the control logic information resides on the PLC in the same format (data array) as defined and stored in the virtual engineering environment database. In practice, this means that control logic installed on the real system can be viewed or edited using the same tools and user interface as the ones used for virtual engineering making it possible to deploy engineering level capabilities on the shop floor.

The right side of Fig. 4 provides a description of the runtime link with PLC control systems, which was

implemented by the ASG using OPC-UA. A standard OPC-UA server was used [64] as it provides access to drivers for a variety of PLCs. An OPC-UA client was developed that can retrieve configuration data from the deployment database (e.g., OPC-UA and application servers IP) and maps the live system information (e.g., working state and faults) according to the corresponding component. A set of application servers provide engineering level services and functionalities which currently include a fault tracker application target at mobile devices and developed using the Unity 3-D engine, and a maintenance information management environment which features a web-based user interface developed using the BootStrap framework.

IV. USE CASE

A. ASW Demonstrator

A full-scale automation system workbench (ASW) is installed in the Warwick Manufacturing Group (WMG), University of Warwick, Coventry, U.K., to support the research and development activities of ASG (see Fig. 7). It is a modular and reconfigurable system and hence the application can be progressively changed as new requirements emerge. Machine stations can be exchanged physically and also virtually, i.e., new virtual station models can be swapped in (and out) in place of physical stations.

The ASW features state-of-the-art control system and automation equipment from leading vendors, e.g., Siemens, Bosch-Rexroth, Rockwell Automation, ABB, Schneider Electric, Mitsubishi, Festo, and SMC. The system has been implemented to support the latest control system design and programming standards. The ASW aims to provide a full-scale demonstrator for new manufacturing automation methods, tools, and technologies with the objective to support the entire lifecycle, e.g., enabling the digital validation, verification and visualization, control code generation, and cloud-based engineering services. The ASW is also used with industrial collaborators (e.g., JLR, Ford, and their supply chains) for demonstration of product assembly. The ASW is currently configured to carry out a battery submodule assembly demonstration as a part of an Innovate UK project. The product assembly consists of 18 650 form-factor cylindrical cells to be assembled into a submodule incorporating bus bars and an integrated cooling system.

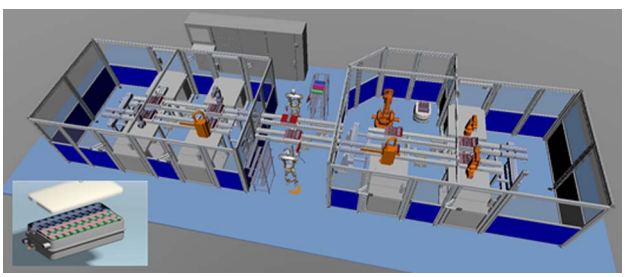


Fig. 7. *Automation system workbench (ASW).*

B. vueOne Engineering Environment

1) Automation Process Modeling: Component modeling is the first step for modeling a manufacturing system. The vueOne component editor module is provided to create components and store them in the vueOne component library. The component editor module provides component modeling functions that cover three domains, i.e., 3-D geometry, kinematics, and control logic. A component could consist of a combination of these three domains.

Link points (i.e., location points within the model space) based assembly functions are provided to enable Lego-like assembly of CAD geometry. Once a component is assembled, kinematic behavior (such as type of joint, displacement, acceleration, and velocity) can be defined to animate the component. Finally, the control behavior of a component can be described via IEC 61131-formatted state-transition diagrams to define the high-level functional states in which a component can exist. Components can be animated in the component editor to verify that the kinematic joints and control behavior are modeled correctly. The transition from one state to another is controlled by the transition conditions. The transition conditions can be defined at system modeling level. An example state-transition diagram to define the control behavior of a two-position actuator component is shown in Fig. 8.

In the vueOne system editor, components of the system are instantiated from the component library and assembled via link points. At system level, sequence of operations is defined using IEC 61131 process logic. Process logic is modeled as a set of step and transition pairs

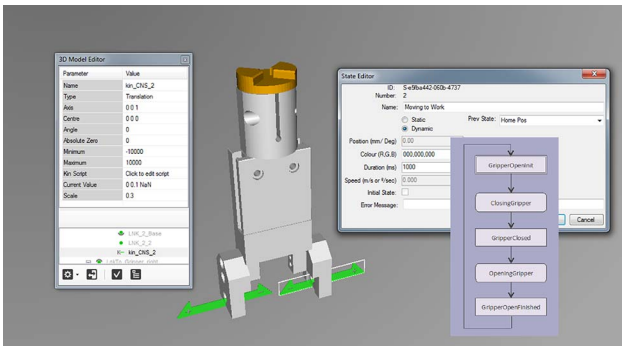


Fig. 8. *Modeling of gripper component in vueOne component editor.*

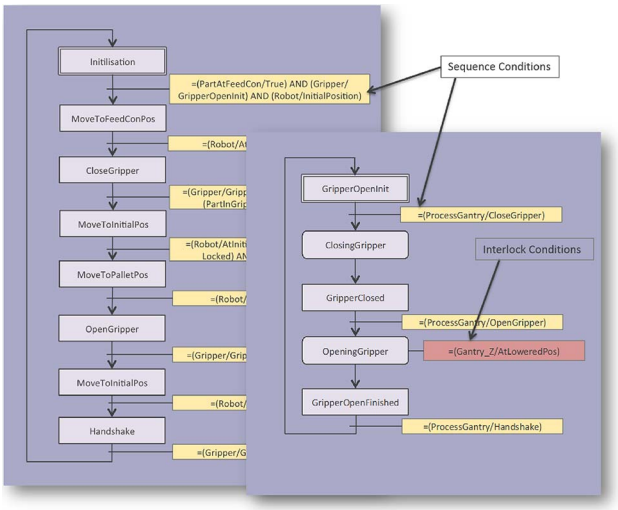


Fig. 9. ***Control logic definition in process logic to define sequence of operation on the left, actuator control behavior on the right.***

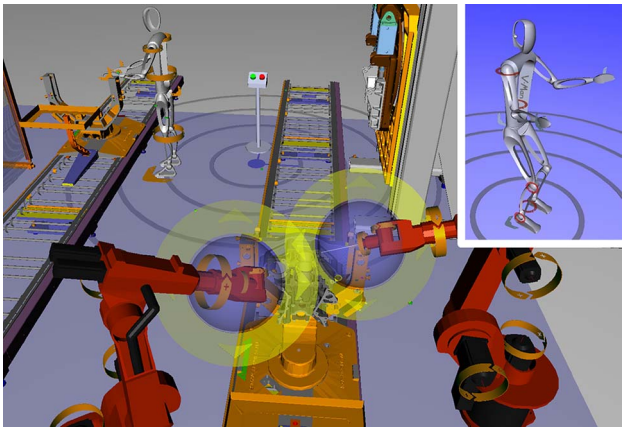


Fig. 10. ***vMan/vRob component in vueOne simulation.***

to combine and orchestrate the service functionalities of a group of components. A machine can have several process logic sequences, communicating with each other to work in an integrated and controlled manner. An example of control logic definition (i.e., sequence of operations, sequence checks, and interlock checks) for a robot station to pick and place parts on pallet is shown in Fig. 9.

2) Human Work Process Modeling: Manufacturing processes still involve a significant amount of semi-automatic and fully manual operations, typically interspersed with automation. Functionalities related to human operation in most commercially available VE tools are typically oriented toward advanced ergonomic assessment and provide little support for validation of human-automation processes interactions. In addition, the required resource investment (i.e., software purchase, training, specific IT hardware) often prevents such solutions being used for cross supply chain collaboration.

The vMan (virtual manikin) module, shown in Fig. 10, developed by the ASG adopts a complementary approach to available solutions by providing 1) a simplified manikin skeleton (eight body joints) and 2) an intuitive process modeling interface that allows engineers to rapidly define and simulate operator processes in the 3) same environment as that used for automation systems modeling.

The vMan module is currently calibrated based on MODular Arrangement of Predetermined Time Standards (MODAPTS) [65] time increment for manual task and can generate MODAPTS formatted description of the simulated process. Future work will focus on implementing settings for using MTM and MOST [66].

3) Industrial Robot Process Modeling: The vRob (virtual robot) module shown in Fig. 10, was designed to enable the inclusion of industrial robots in the vueOne environment to enable simple robot sequence emulation and provide a collaboration platform for engineers implementing various types of processes. The vRob module was not designed to replace offline robot programming tools (e.g., Robot Studio, RT Toolbox) but rather to complement them. As such, interfaces to import/export robot sequences (position, time) to and from offline programming tools are being implemented (currently import from the ABB RAPID language is supported). A model library is provided to allow easy access to robot components comprising CAD, preconfigured inverse kinematics models and tooling fixture points (currently 15 6R robots from various manufacturers).

4) System Visualization and Validation: The manufacturing process plan, mechanical configuration, control behavior, and cycle time of the system can be validated using vueOne system editor as well as vueOne viewer.

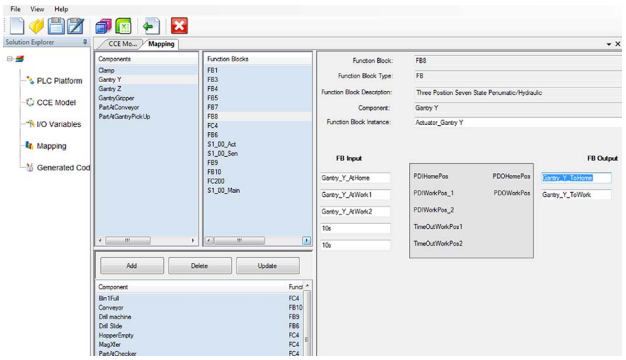


Fig. 11. ***User interface of vueOne deployment for input–output and function block mapping.***

The vueOne viewer is a lightweight application that is designed to be used across the supply chain with minimal or no training for both technical and nontechnical users. Validation of the system behavior can be carried through simulation, timing chart, and sequence and interlock chart. After validation, the data model can be exported in XML format for use at later phases of the engineering process, such as control logic generation, discrete event simulation, and energy analysis.

5) PLC Control Deployment Module: The deployment module, shown in Fig. 11, enables the deployment of control code for the vueOne system on the target PLCs and supports generation of the related HMI screens. The control code deployment process can be broken down into the pre-engineering phase and the system-engineering phase. The pre-engineering phase is composed of resource-specific function block development and software template development for the target platform. Software templates are developed once and stored in the deployment module library, whereas resource-specific function blocks are prevalidated control software components to control a mechanism or family of mechanisms or other devices. The software templates define the structure of the generated control code.

The tasks carried out during the system engineering phase are specific to the targeted system, which includes target PLC/device selection, function block mapping, input–output mapping, and code generation.

The PLC code generated by the vueOne deployment for a Pick&Place station is shown in Fig. 12. The Pick&Place station is controlled by Siemens SIMATIC S300 PLC with distributed input–output modules and SIMATIC MP277 8″ touch panel. The code for Siemens PLCs is generated as text files which are imported in the Siemens STEP 7 software tool for compilation and download into the PLC.

The vueOne deployment module supports automatic generation of HMI for both web-based and vendor-specific platforms. The web-based HMI screens can be deployed on industrial PCs and tablets. An example manual mode control screen generated for MP277 touch panel based on Ford's FAST standard for the Pick&Place station is shown in Fig. 12.

In addition, the deployment module also generates tags for configuration of OPC-UA applications and a REST server to send information (such as production information, fault information, and status of sensors and actuators) to the data servers. This enables seamless integration between shop floor and virtual engineering environments.

6) Fault Tracker and Maintenance Tracker Tools: To support maintenance operations at both shop-floor and top-floor levels, a fault tracker application was developed. The fault tracker application is targeted at smartphones, tablets, and laptops, which are considered as interfaces to humans operating in the shop floor and who interact with (and are part of) CPSs by receiving, processing, and generating data/information.

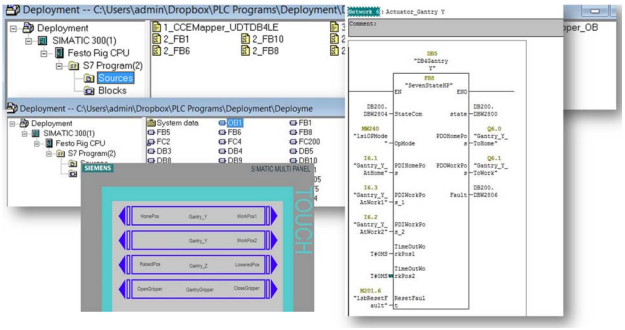


Fig. 12. ***Autogenerated PLC code and HMI screen.***

The fault tracker screens shown in Fig. 13 target three main functionalities. 1) Providing a gateway for accessing and visualizing live system fault information generated by PLCs devices, propagated through the UPC-UA interface and managed by the vueOne fault server (see Fig. 6, Section III-D above). Information provided by the fault tracker includes fault code and HMI screen message, system/component identification, fault status, log time, and priority level. 2) The second set of screens aims to provide identification of the attended system/component through QR code scanning and access to system design data such as 3-D models/simulation used during the VE phase, vendor-specific information and maintenance information (e.g., robot data sheets, assembly schematics, safety and hazard warnings, etc.) or repair instructions and breakdown or replacement procedures. 3) The last screen aims at enabling

Fig. 13. ***vueOne fault tracker mobile application's main screens.***

maintenance technicians or engineers to upload fault and maintenance information to the database (textual information, camera device capture).

V. CONCLUSION AND FUTURE WORK

A system-engineering environment has been presented aimed at supporting the complete lifecycle of distributed component-based automation from design to operation and reconfiguration. The vueOne engineering environment functionalities focus on 1) virtual engineering and validation of CPSs physical layout and control logic; 2) direct deployment of component-based control to PLCs and other embedded devices; 3) the connectivity between the virtual data set and the real system; 4) enhanced support for operation related tasks (e.g., maintenance); and 5) maintain consistency between the virtual (cyber) and physical systems and therefore enhance capitalization and reuse of engineering knowledge and data.

The appearance of open platforms, tools, and standards data formats has the potential for dramatically expanding the scope of players in the CPS automation innovation processes, disrupting old business models. Standards are on the critical path of CPS engineering tools' development as they accelerate the implementation and introduction of nonproprietary solutions (which are essential for Industry 4.0), and also enable smaller companies to adapt more rapidly. The ASG is focusing its research on implementing lightweight, usable, and open solutions based on open standards and technologies (e.g., XML data representation, web-based interface, and web-services-based software integration). The vueOne future development map includes the implementation of interfaces to import and export to interchange formats, most notably AutomationML.

Defining and reinforcing the use of common data interchange formats, achieving connectivity between engineering organizations and the engineering software those organizations use, is critical. Industry 4.0 promotes the use of web-services-based methods in order to achieve agile integration of engineering software. The ASG has implemented REST APIs for a number of the vueOne software components and is testing integration with a number of commercial applications.

For SMEs, the cost associated with the deployment of engineering software (e.g., license purchase, maintenance, support, updates, and training) is often a major barrier to establishing advanced engineering capabilities. The ASG aims at providing virtual engineering capabilities currently inaccessible to most SME and therefore impacting on their ability to effectively collaborate with larger companies. The ASG is also focusing on developing the vueOne environment as a portal of services or Software as a Service (SaaS) which, associated with a subscription-based business model, will in the long-term enable SMEs to deploy engineering capabilities at lower costs and in a more agile manner. ■

REFERENCES

[1] E. Lee, "Cyber physical systems: Design challenges," in *Proc. 11th IEEE Int. Symp. Object Oriented Real-Time Distrib. Comput.*, 2008, pp. 363–369.

[2] VDI/VDE Society, "Reference architecture model Industrie 4.0 (RAMI4.0)," Status Rep., 2015.

[3] T. Aicher, D. Schutz, and B. Vogel-Heuser, "Consistent engineering information model for mechatronic components in production automation engineering," in *Proc. 40th Annu. Conf. IEEE Ind. Electron. Soc.*, 2014, pp. 2532–2537.

[4] C. J. Paredis *et al.*, "An overview of the SysML modelica transformation specification," in *Proc. INCOSE Int. Symp.*, 2010, pp. 709–722.

[5] A. A. Shah, A. A. Kerzhner, D. Schaefer, and C. J. Paredis, "Multi-view modeling to support embedded systems engineering in SysML," in *Graph Transformations and Model-Driven Engineering*, New York, NY, USA: Springer-Verlag, 2010, pp. 580–601.

[6] Y. Cao, Y. Liu, and C. J. Paredis, "System-level model integration of design and simulation for mechatronic systems based on SysML," *Mechatronics*, vol. 21, pp. 1063–1075, 2011.

[7] K. C. Thramboulidis, "Using UML in control and automation: A model driven approach," in *Proc. 2nd IEEE Int. Conf. Ind. Inf.*, 2004, pp. 587–593.

[8] A. Luder, N. Schmidt, M. Foehr, T. Schaffler, and J. Elger, "Evaluation of the importance of mechatronical concepts in practical applications," in *Proc. IEEE 18th Conf. Emerging Technol. Factory Autom.*, 2013, pp. 1–8.

[9] A. Luder, N. Schmidt, and R. Rosendahl, "Behavior validation of production systems within different phases of the engineering process," in *Proc. 39th Annu. Conf. IEEE Ind. Electron. Soc.*, 2013, pp. 6906–6911.

[10] A. Lüder, E. Estévez, L. Hundt, and M. Marcos, "Automatic transformation of logic models within engineering of embedded mechatronical units," *Int. J. Adv. Manuf. Technol.*, vol. 54, pp. 1077–1089, 2011.

[11] H. Kagermann, J. Helbig, A. Hellinger, and W. Wahlster, "Recommendations for implementing the strategic initiative Industrie 4.0: Securing the future of German manufacturing industry," ForschungsunionFinal Report of the Industrie 4.0 Working Group, 2013.

[12] O. Niggemann and B. Kroll, "On the applicability of model based software development to cyber physical production systems," *Proc. IEEE Emerging Technol. Factory Autom.*, pp. 1–4, 2014.

[13] K. Thramboulidis, "Model-integrated mechatronics—Toward a new paradigm in the development of manufacturing systems," *IEEE Trans. Ind. Inf.*, vol. 1, pp. 54–61, 2005.

[14] E. Estevez *et al.*, "A novel approach to attain the true reusability of the code between different PLC programming tools," in *Proc. IEEE Int. Workshop Factory Commun. Syst.*, 2008, pp. 315–322.

[15] B. Vogel-Heuser *et al.*, "Challenges for software engineering in automation," *J. Softw. Eng. Appl.*, pp. 440–451, 2014.

[16] R. Harrison and A. A. West, "Component based paradigm for the design and implementation of control systems in electronics manufacturing machinery," *J. Electron. Manuf.*, vol. 10, pp. 1–17, 2000.

[17] R. Harrison, S. M. Lee, and A. A. West, "Component-based distributed control system for automotive manufacturing machinery under the Foresight Vehicle Program," presented at the *SAE Conf.*, 2002.

[18] D. W. Thomas, A. A. West, R. Harrison, and M. C. S., "A process definition environment for component based manufacturing machine control systems developed under the foresight vehicle programme," in *Proc. SAE World Congr. Expo. Foresight Vehicle Technol., Consumer Driven Design, Manufacturing, Supply Chain and Purchasing*, Detroit, MI, USA, 2002.

[19] R. Harrison, A. Colombo, A. West, and S. Lee, "Reconfigurable modular automation systems for automotive power-train manufacture," *Int. J. Flexible Manuf. Syst.*, vol. 18, pp. 175–190, 2006.

[20] A. W. Colombo and R. Harrison, "Modular and collaborative automation: Achieving manufacturing flexibility and reconfigurability," *Int. J. Manuf. Technol. Manage.*, vol. 14, pp. 249–265, 2008.

[21] J. Sztipanovits, T. Bapty, S. Neema, L. Howard, and E. Jackson, "OpenMETA: A model- and component-based design tool chain for cyber-physical systems," in *From Programs to Systems. The Systems Perspective*

in Computing, New York, NY, USA: Springer-Verlag, 2014, pp. 235–248.

[22] R. Harrison *et al.*, "Next generation of engineering methods and tools for SOA-based large-scale and distributed process applications," in *Industrial Cloud-Based Cyber-Physical Systems*, New York, NY, USA: Springer-Verlag, 2014, pp. 137–165.

[23] A. Fay *et al.*, "Enhancing a model-based engineering approach for distributed manufacturing automation systems with characteristics and design patterns," *J. Syst. Softw.*, vol. 101, pp. 221–235, 2015.

[24] R. Harrison and A. W. Colombo, "Collaborative automation from rigid coupling towards dynamic reconfigurable production systems," in *Proc. 16th IFAC World Congr.*, Czech Republic, 2005, pp. 1–9.

[25] R. Harrison, A. A. West, and L. J. Lee, "Lifecycle engineering of future automation systems in the automotive powertrain sector," in *Proc. IEEE Int. Conf. Ind. Inf.*, 2006, pp. 305–310.

[26] V. Vyatkin, "IEC 61499 as enabler of distributed and intelligent automation: State-of-the-art review," *IEEE Trans. Ind. Inf.*, vol. 7, pp. 768–781, 2011.

[27] W. Dai and V. Vyatkin, "Redesign distributed PLC control systems using IEC 61499 function blocks," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, pp. 390–401, 2012.

[28] K. L. Lueth, "Will the industrial Internet disrupt the smart factory of the future?" 2015. [Online]. Available: http://iot-analytics.com/industrial-internet-disrupt-smart-factory/.

[29] D. Zühlke and L. Ollinger, "Agile automation systems based on cyber-physical systems and service-oriented architectures," *Adv. Autom. Robot.*, vol. 122, pp. 567–574, 2012.

[30] F. Jammes, A. Mensch, and H. Smit, "Service-oriented device communications using the devices profile for web services," in *Proc. 3rd Int. Workshop Middleware Pervasive Ad-Hoc Comput.*, 2005, pp. 1–8.

[31] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC Unified Architecture*. New York, NY, USA: Springer-Verlag, 2009.

[32] S. Karnouskos *et al.*, "Real-world service interaction with enterprise systems in dynamic manufacturing environments," in *Artificial Intelligence Techniques for Networked Manufacturing Enterprises Management*, New York, NY, USA: Springer-Verlag, 2010, pp. 423–457.

[33] I. M. Delamer and J. L. M. Lastra, "Loosely-coupled automation systems using device-level SOA," in *Proc. 5th IEEE Int. Conf. Ind. Inf.*, 2007, pp. 743–748.

[34] OPC Foundation, "Interoperability for Indsutrie 4.0 and the Internet of Things," 2015.

[35] M. J. A. G. Izaguirre, A. Lobov, and J. L. M. Lastra, "OPC-UA and DPWS interoperability for factory floor monitoring using complex event processing," in *Proc. 9th IEEE Int. Conf. Ind. Inf.*, 2011, pp. 205–211.

[36] X. Xu, "From cloud computing to cloud manufacturing," *Robot. Comput.-Integr. Manuf.*, vol. 28, pp. 75–86, 2012.

[37] IMC-AESOP, "Architecture for service-oriented process—Monitoring and control," FP7 theme ICT. [Online]. Available: http://imc-aesop.org.

[38] "Service-oriented cross-layer infrastructure for distributed smart embedded devices," FP7 theme ICT. [Online]. Available: http://www. socrades.net.

[39] "Service-Oriented Device & Delivery Architectures (SODA)," ITEA 3. [Online]. Available: https://itea3.org/project/SODA.html.

[40] J. M. Mendes *et al.*, "Software methodologies for the engineering of service-oriented industrial automation: The continuum project," in *Proc. 33rd Annu. IEEE Int. Comput. Softw. Appl. Conf.*, 2009, pp. 452–459.

[41] A. W. Colombo and F. Jammes, "Integration of cross-layer web-based service-oriented architecture and collaborative automation technologies: The SOCRADES approach," *Inf. Control Probl. Manuf.*, pp. 2101–2106, 2009.

[42] T. Kirkham *et al.*, "SOA middleware and automation: Services, applications and architectures," in *Proc. 6th IEEE Int. Conf. Ind. Inf.*, 2008, pp. 1419–1424.

[43] M. Prösser, P. Moore, X. Chen, C.-B. Wong, and U. Schmidt, "A new approach towards systems integration within the mechatronic engineering design process of manufacturing systems," *Int. J. Comput. Integr. Manuf.*, vol. 26, pp. 806–815, 2013.

[44] P. Leitao, J. Barbosa, M.-E. C. Papadopoulou, and I. S. Venieris, "Standardization in cyber-physical systems: The ARUM case," in *Proc. IEEE Int. Conf. Ind. Technol.*, 2015, pp. 2988–2993.

[45] S. Biffl, R. Mordinyi, and T. Moser, "Automated derivation of Configurations for the integration of software (+) engineering environments," in *Proc. ACoTA*, 2010, pp. 6–13.

[46] E. Westkämper and L. Jendoubi, "Smart factories-manufacturing environments and systems of the future," in *Proc. 36th CIRP Int. Seminar Manuf. Syst.*, 2003, pp. 13–16.

[47] Siemens, "Process simulate: Manufacturing process verification in powerful 3D environment," 2011.

[48] Dassault Systems, "DELMIA V5 automation platform: Merging digital manufacturing with automation," ARC Advisory Group, 2006.

[49] M. Oppelt *et al.*, "Automatic model generation for virtual commissioning based on plant engineering data," in *Proc. 19th World Congr. Int. Fed. Autom. Control*, 2014, pp. 1–6.

[50] M. Bergert and J. Kiefer, "Mechatronic data models in production engineering," presented at the *10th IFAC Workshop Intell. Manuf. Syst.*, Lisbon, Portugal, 2010.

[51] M. Tenorth and M. Beetz, "Exchanging action-related information among autonomous robots," in *Intelligent Autonomous Systems 12*, New York, NY, USA: Springer-Verlag, 2013, pp. 467–476.

[52] S. Seidel, U. Donath, and J. Haufe, "Towards an integrated simulation and virtual commissioning environment for controls of material handling systems," in *Proc. Winter Simul. Conf.*, 2012, p. 252.

[53] P. Hoffmann, R. Schumann, T. M. Maksoud, and G. C. Premier, "Virtual commissioning of manufacturing systems: A review and new approaches for simplification," in *Proc. ECMS*, 2010, pp. 175–181.

[54] D. A. Vera, A. West, and R. Harrison, "Innovative virtual prototyping environment for reconfigurable manufacturing system engineering," in *Proc. Inst. Mech. Eng. B/J. Eng. Manuf.*, 2009, vol. 223, pp. 609–621.

[55] H. ElMaraghy, "Flexible and reconfigurable manufacturing systems paradigms," *Int. J. Flexible Manuf. Syst.*, vol. 17, pp. 261–276, 2005.

[56] E. Red, D. French, G. Jensen, S. S. Walker, and P. Madsen, "Emerging design methods and tools in collaborative product development," *J. Comput. Inf. Sci. Eng.*, vol. 13, 2013, 031001.

[57] K. David, "Virtual commissioning of factory floor automation: The new paradigm in vehicle manufacturing," presented at the *SAE World Congr. Exhibit.*, Detroit, MI, USA, 2010.

[58] X. Kong *et al.*, "Realising the open virtual commissioning of modular automation systems," in *Proc. 7th CIRP Int. Conf. Digit. Enterprise Technol.*, Athens, Greece, 2011.

[59] S. Makris, G. Michalos, and G. Chryssolouris, "Virtual commissioning of an assembly cell with cooperating robots," *Adv. Decision Sci.*, vol. 2012, p. 11, 2012.

[60] B. Ahmad, "*A component-based virtual engineering approach to PLC code generation for automation systems*," Ph.D. dissertation, Wolfson Schl. Mech. Manuf. Eng., Loughborough Univ., Loughborough, U.K., 2014.

[61] X. Kong, B. Ahmad, R. Harrison, Y. Park, and L. J. Lee, "Direct deployment of component-based automation systems," in *Proc. IEEE 17th Conf. Emerging Technol. Factory Autom.*, 2012, pp. 1–4.

[62] B. Ahmad, X. Kong, R. Harrison, J. Watermann, and A. W. Colombo, "Automatic generation of human machine interface screens from component-based reconfigurable virtual manufacturing cell," in *Proc. 39th Annu. Conf. IEEE Ind. Electron. Soc.*, 2013, pp. 7428–7433.

[63] Ford Motor Company, "FAST PLC structure manual," 2013.

[64] Software Toolbox Inc., "TOP server product details-configuration," 2012. [Online]. Available: http://www.toolboxopc.com/html/configuration.html.

[65] P. Carey, "Heyde's MODAPTS: A language of work," Heyde Dynamics, 2001.

[66] K. B. Zandin, *MOST Work Measurement Systems*. Boca Raton, FL, USA: CRC Press, 2002.

ABOUT THE AUTHORS

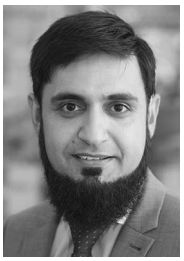
Robert Harrison received the B.Tech. and Ph.D. degrees in mechanical and manufacturing engineering from Loughborough University, Loughborough, U.K., in 1981 and 1991, respectively.

He is Professor of Automation Systems in the Warwick Manufacturing Group (WMG), University of Warwick, Coventry, U.K. He has been principal investigator on more than 35 industrially oriented European Union, U.K. Government, and commercial R&D projects related to manufacturing automation, with current projects focusing on lifecycle engineering and virtual commissioning, control deployment, and augmented reality in applications including future production systems for batteries, fuel cells, and electric machines. He led the U.K. research related to Ford's Technology Cycle Plan for powertrain manufacturing automation.

Dr. Harrison was recipient of a Royal Academy of Engineering Global Research Award to study "Lifecycle Engineering of Modular Reconfigurable Manufacturing Automation."

Daniel Vera received the M.Sc. degree in mechanical and manufacturing engineering from E.N.I. Tarbes, France, in 2000 and the Ph.D. degree in manufacturing engineering from Loughborough University, Loughborough, U.K., in 2004.

He has been working in the domain of manufacturing engineering for over ten years. His research interests are focused on various aspects of manufacturing from the modeling, analysis, and optimization of engineering processes to the design and development of 3-D-based virtual engineering and collaboration tools for supporting the manufacturing system lifecycle, which formed the focus of his Ph.D. disseration. He has been involved in numerous U.K. and European projects as a Research Associate at Loughborough University, Loughborough, U.K. and now a Research Fellow at the University of Warwick, Coventry, U.K. He is currently taking a leading role in the commercialisation of new automation systems engineering tools and services.



Bilal Ahmad (Member, IEEE) received the MS.c. degree in mechatronics and the Ph.D. degree in automation systems from Loughborough University, Loughborough, U.K., in 2007 and 2014, respectively.

He is working as a Research Fellow in the Warwick Manufacturing Group (WMG), University of Warwick, Coventry, U.K. His research interests are in the area of virtual modeling and control software engineering of industrial automation systems. He has worked on a number of U.K. and European Union engineering research projects in collaboration with automotive manufacturers, machine builders, and control vendors to develop tools and methods to support lifecycle engineering of industrial automation systems. He has previously worked as a Research Associate at the Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University.