Assessing Complexity of Component-Based Control Architectures Used in Modular Automation Systems

Bugra Alkan*, Daniel Vera, Malarvizhi Kaniappan Chinnathai, Robert Harrison
Automation Systems Group, WMG, University of Warwick, CV4 7AL, Coventry, West Midlands, UK.

* Corresponding author. Tel.: +44 7469290652; email: B.Alkan@warwick.ac.uk
Manuscript submitted February 25, 2017; accepted April 1, 2017.
doi: 10.17706/ijcee.2017.9.1.393-402

Abstract: Component-based development (CBD) supports hierarchical decomposition of manufacturing control architectures through data and procedural abstraction, allowing designers to handle system development complexity better than function-oriented methods. Although the CBD approach helps managing complexity of the software design and development process, it does not reduce or eliminate complexity of control systems. In fact, large and highly coupled system architectures make entire software very difficult to understand and modify, especially during manufacturing system re-configuration and scale up/down processes. Therefore, it is essential to maintain simplicity in control system design, without disregarding the required modularity and functionality. This paper proposes an information-theoretic measure to quantify the complexity of component-based manufacturing control systems. The proposed measure is tested over the auto-generated control codes of Festo MPS system for its validity. The authors believe that the proposed approach can serve as a proactive design support, especially useful for early design stages as it allows designers to select the optimal control architectures with least complexity and provides a clear understanding of the potential stress points.

Key words: Complexity, component-based development, distributed control, software metrics, information theory.

1. Introduction

The recent works on advanced automation technologies shows that component-based control systems tend to improve agility and robustness in automation systems [1]. A component-based manufacturing control system aims at reusing pre-developed software components from project to project, thereby, enabling reduction in software development time and cost [2]. This approach establishes the divide and conquer principle in software design by dividing software systems into relatively small components, therefore providing a mean for designers to handle system development complexity. However, software complexity is unavoidable, and is the result of increasing level of functionalities expected from the control systems being designed. Moreover, deprived design choices during system development result in very complex software architectures that are difficult to understand, modify and maintain [3].

Complexity management in a manufacturing control system begins with an accurate prediction of stress points at early design stages. This paper introduces an information theoretic complexity measure for analysing component-based control architectures. The proposed measure analyses several attributes of software components and flags components with overcomplicated information and control flow and lack of cohesion. Furthermore, the introduced measure is used as a design support mechanism in the vueOne virtual system design and validation tool developed by the Automation Systems Group (ASG) at the University of Warwick; where, auto-generated control codes of various design alternatives can be analysed and compared.

2. Research Background

The CBD in manufacturing is a paradigm that employs predefined components to assemble production control systems [4]. This knowledge reuse results in drastic increase in productivity. In the CBD, entire control software can be built from pre-defined components, in a modular form, rather than writing monolithic control code each time from scratch [5]. The behaviour and capabilities of components are visible through interfaces; however, their detailed implementation is often designed as hidden. In this context, encapsulation provides a higher level of abstraction and avoids the need to give attention to component details since the component interfaces can be exploited. Libraries that store a set of pre-defined, pre-tested components and re-usable components [6] can therefore be created. By arranging components from the library in a specific configuration, control systems can be built. These components can be later re-arranged, added, removed by reconnecting their interfaces to generate new configurations. Thus, the system can be effectively reconfigured to meet new demands. Moreover, CBD can be applied in the virtual engineering domain, to simulate real systems for the purpose of process planning and testing. This enables significant savings in time and cost.

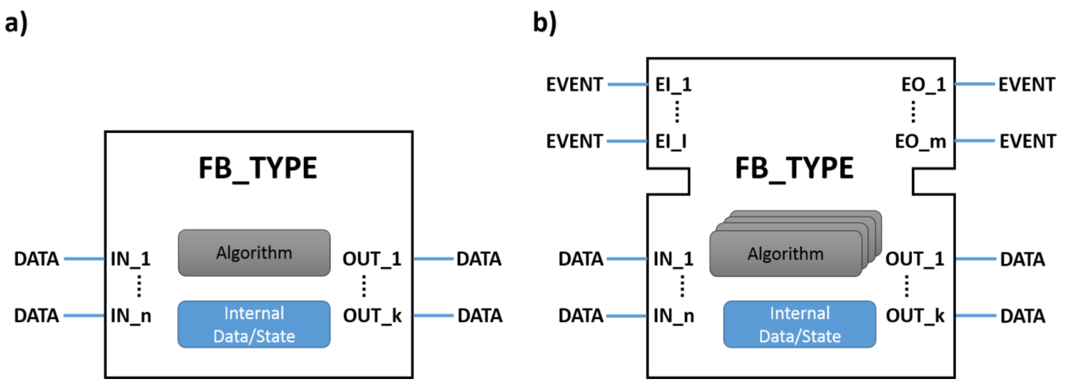

Fig. 1. a) The IEC 61131-3 and b) the IEC 61499 FB architectures (adapted from [12]).

According to Dai and Vyatkin [1], IEC 61131-3 standard [7] is best suited for the design of Programmable Logic Controller (PLC) architectures in industrial applications. In the IEC 61131-3 standard, programing organisation units (POU) (i.e. functions (FCs) and function blocks (FBs)) are often referred as reusable software components. FBs in the IEC 61131-3 standard include a certain functionality and can be connected to other FBs via component interfaces [8] (Fig. 1a). In this standard, FB's source code covers one algorithm written in one of five languages supported by the IEC 61131-3 [9]. It should be noted that, FBs in the IEC 61131-3, also include data that is required for maintaining its state between calls. Similarly, control applications can be modularised by encapsulating the application parts in the FBs, enabling the reuse of the application parts [10]. As a summary, the drawbacks that impede the reusability of the IEC 61131-3 standard are further discussed. In this standard, the global data acts like an invisible interface among FBs and subsequently leads to highly coupled control architectures. This reduces the modularity and interpretability of the control code. Moreover, this standard allows limited control over the execution order of FBs. In addition to this, there might be compatibility issues while running applications in different control devices [1].

Although the IEC 61131-3 standard has been widely used in industrial automation domain, this standard is incapable of addressing the requirements of today's complex industrial systems [11]. To overcome this limitation, the IEC 61499 standard which was upgraded from the existing IEC 61131-3 architecture was introduced [12]. The IEC 61499 includes event driven FBs. These FBs generally remain passive until they are invoked by an event casted through an input event (Fig. 1b). Additionally, the event interface was introduced in this standard. The connection of events makes the implementation of this standard more complex, but it provides designers with added flexibility by allowing explicit specification of the sequence of FB execution [12]. The absence of global data in IEC 61499, contrary to that of the IEC 61131-3 allows the reusability of the FBs without impacting the whole system, while just the connected FBs are affected. Moreover, FBs in the IEC 61499 may contain different algorithms which are neither visible nor accessible from the outside. As a result of these properties, a FB in the IEC 61499 standard is capable of acting as an independent software component that can be implemented, tested and used independent of other FBs. In spite of the benefits of IEC 61499, its prevalence in industry is limited [10] primarily due to the reluctance and the effort involved in the change.

According to Crynkovic and Larsson [13], component-based control systems are not general enough, and components are often considered as difficult to use, adapt and maintain. However, CBD provides reusability and flexibility, since it is possible to reuse components stored in the library by the connection of event interfaces. Although components can be reused, it is important to note that the definitions and functionalities of components should be changed depending on the customer requirements, new product introduction and site-specific functionalities [1]. This affects the size of components and coupling between them, leading to situations where the entire control code becomes difficult to understand, modify and maintain.

3. The Proposed Complexity Measure

As production systems become larger, users demand more reliable and maintainable software systems. A key need in the development of control systems is the design simplicity. As said by Grady [14], complexity is one of the major contributor that impacts the development and maintenance costs of software systems. This is due to the fact that, the increase in complexity of control systems makes it difficult to detect and correct faults [15]. In this manner, systematic minimisation of complexity during early design stages without compromising the required system functionality, will result in "a lean control system" that provides significant benefits, such as: ease of reconfiguration and maintenance, and increased predictability. In this study, by following the information axiom principle introduced in Suh's axiomatic design theory [16], complexity is related to the information content of the system components which is calculated in terms of information entropy as shown in Eq. (1); where C_i can have any values between 0 and 1 implying the respective conformity of component design quality governed by the probability of fulfilling the design requirements (DRs).

$$I_i = -\log_2\left(C_i\right) \tag{1}$$

Accordingly, the conformity of design quality increases as the following DRs are satisfied.

- DR1 Minimising inter-module complexity: This complexity type defines the degree of linkage between (i.e. type and amount of information exchange) the components within the same system. Reducing inter-module complexity without impeding functional requirements, improves changeability and modifiability of the system, thus increases the design quality [17].
- DR2 Minimising intra-module complexity: This complexity type arises due to two main reasons, i.e. lack of cohesion and complicated control flow within the component. If a component has to perform a

wide range of functions or designed to support a wide range of application, it is exposed to lack of cohesion. According to Phukan, Kalava and Prabhu [15], components with lack of cohesion are often difficult to maintain and less reliable, thus, the components with lack cohesion should be divided into smaller components with the increased degree of cohesion. Moreover, components with excessive use of loops, jumps or program selections, are subjected to a complicated control flow, making them difficult to modify and change. Hence, overcomplicated control flows should be avoided, if they do not assist in achieving the required functionality.

In this study, the probability of satisfying the mentioned DRs is calculated by the conformity ratios representing the deviation between the actual structure and what software experts desire to reach in terms of tolerance. Accordingly, the information content of a software component Ii is defined as follows;

$$I_i = w_1 \log_2\left(\frac{1}{C_{\mathrm{intra},i}}\right) + w_2 \log_2\left(\frac{1}{C_{\mathrm{inter},i}}\right) \tag{2}$$

$$w_1 + w_2 = 1 \tag{3}$$

where, $C_{\mathrm{intra},i}$ and $C_{\mathrm{inter},i}$ are the intra-module and inter-module design conformities of ith component, respectively. Also, the equation contains weight coefficients (i.e. w_1 and w_2), which are proposed to achieve flexibility during decision-making stages. Furthermore, complexity of a control system is considered as the sum of the information content of all components and is calculated as follows;

$$H_m = \sum_{i=1}^{i=k} I_i \tag{4}$$

where, H_m is the complexity of control system m, k is the number of software components with the system m, and i is the component index. Accordingly, designs satisfying all required functional requirements with minimal information content is considered as the optimum design.

Measuring information content is useful to flag stress points and to select best among many acceptable design solutions. Moreover, it enables a theoretical basis for design optimisation and robust design. However, having a high conformity of fulfilling the DRs leading to a simple control system may not be useful and realistic in most cases. It is important to note that, reducing complexity of physical entity may increase the uncertainty in satisfying system's functional requirements given in a specific range, and vice versa. Therefore, while designing a control system, both functional requirements and design simplicity should be satisfied, simultaneously, i.e. lean system design.

3.1. Intra-Module Design Conformity

The intra-module design conformity of a component is described as the function of the program code volume proposed in Halstead's information science model [18], i.e. the total and unique number of operator and operand used, and a design tolerance coefficient x, (Eq. 5);

$$C_{\mathrm{intra},i} = \begin{cases} 1 - \dfrac{(P_1 + P_2)\log_2(p_1 + p_2)}{x} & if\ C_{\mathrm{intra},i} > 0 \\ 0 & if\ C_{\mathrm{intra},i} \le 0 \end{cases} \tag{5}$$

where, P_1 and P_2 represent the total number of operands and operators, whereas p_1 and p_2 depict the unique number of operands and operators, respectively. According to experts in the field, the coefficient of intra-module design tolerance for FCs and FBs are 2000 and 10000, respectively. Note that, depending on the site-specific requirements, values can be varied subjectively. Accordingly, components with 0 intra-module design conformity is considered as invalid design. Also components with low C_{intra} scores indicate a lack of cohesion (i.e. wide range of tasks performed) and/or complicated control flow (i.e. more loops, jumps, and program selections, etc.). In industrial applications, such components are often preferred as they provide a wide range of functionality and re-usability in different applications. However, excessive use of components with low intra-module design in a control architecture may affect the maintainability of the code.

3.2. Inter-Module Design Conformity

In this research, a modified version of information flow metrics proposed by Henry and Kafura [19], is used to calculate inter-module design conformity of a software component:

$$C_{\text{inter},i} = \begin{cases} 1 - \dfrac{(f_{in} + f_{out})\log_2(F_{in} + F_{out})}{y} & if \;\; C_{\text{inter},i} > 0 \\ 0 & if \;\; C_{\text{inter},i} \le 0 \end{cases} \tag{6}$$

where, F_{in} and F_{out} represent the total number of information flows that terminate at and emanate from the component, whereas f_{in} and f_{out} are the number of components that the information is received from and updated by the component, respectively. The inter-module design tolerance is taken as 150 for all FCs and FBs. Accordingly, components with low inter-module design conformity indicate a stress point in the system which means a change in such components would have the tendency to ripple across many other components, making implementations and modifications difficult to realise [15]. It is also believed that the high degree of coupling should be avoided in order to realise modular design and the high degree of reusability.

4. Assessment of Auto-Generated Control Codes

The theoretical model proposed in the previous chapter has to be integrated into control code development processes, such that the complexity of the control code can be visualised by the designer as and when the design is generated. This serves as an enabler for concurrent engineering allowing modifications with time savings and comparison of different control architecture designs. To demonstrate the proposed approach, the vueOne toolset developed in the Automation Systems Group (ASG) at the University of Warwick is utilised. The vueOne toolset provides a simulation environment in which processes for automated systems, industrial robots and human operators can be modelled, integrated and validated virtually. The vueOne has a component library that provides building blocks to configure complete industrial automation systems by aggregation of components. The component in the vueOne is an encapsulated design block built from several data sets representing: mechanical data, process data, control (FCs and FBs) and data integration and mapping information (mapping between process state and 3D/physical actuator position and/or motion time, mapping between FBs' I/Os and process states, etc.). In the vueOne, architecture of the system control software is automatically generated using mentioned standard library components, which are driven by the control logic defined in the manufacturing process simulation tool. The control system architecture, as shown in Fig. 2, consists of control data model, resource control components (RRC) and logic engine (LE).

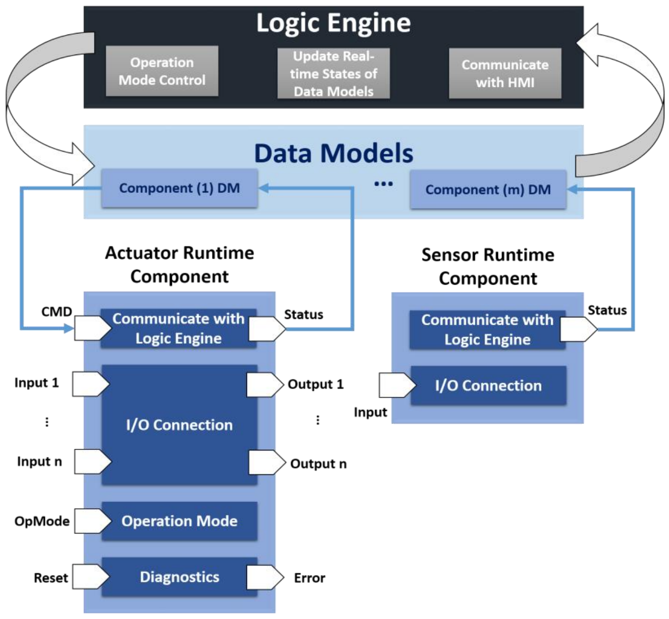

Fig. 2. The VUEONE control system architecture (adapted from [20]).

- Control data model: is an auto-built memory including various types of information i.e. system configuration, execution sequence, control behaviour of field devices, interlocks and fault messages.
- Resource control components (RCC): are resource specific FBs [5]. RCCs can be sensors or actuators in a PLC runtime environment and are embedded with the control behaviour of a family of actuators and sensors with integrated diagnostics [20]. They are developed once and stored in RCC library and can be reused across different implementations. All events (e.g. faults, etc.) of RCCs are communicated to the LE and control data model. RCCs are directly deployable in a PLC program and are interfaced via direct parameterisation to increase the visibility of input and output variables during online PLC monitoring.
- Logic engine (LE): is a pre-written and validated FB, which orchestrates the system such that manufacturing processes can be executed in a controlled manner [21] based on the information contained in the data model. The entire source code of LE is generic and is independent of system and process configuration.

The automatic generation of control codes in the vueOne toolset is realised in three stages, i.e. control model generation, component mapping, and source code generation. In control model generation, control information of each component is extracted from the simulation XML and converted into structured data sets using arrays. Using the platform-specific templates, the control data model is automatically converted to a data format which is liable on the targeted PLC platform. Component mapping refers to the mapping of RCCs with I) virtual actuator and sensor components of the system and ii) physical I/O addresses of sensors and actuators. In the vueOne, component mapping is performed using a mapper module. This module provides interfaces to import simulation model of manufacturing systems, define I/O addresses, component mapping and target PLC platform selection. Furthermore, it provides libraries for storing and managing RCCs and platform-specific elements. To generate the source code, all the software elements created are integrated within the platform-specific templates. In the auto generated code, RCCs of sensors appear as FCs, whereas logic engine and actuator components appear as FBs. The generation of required POUs for RCCs is based on the component mapping information. A POU is created for every RCC which is mapped to virtual sensor and actuator components at the component mapping process. The main program is composed of instances of RCCs that are used in the project and an instance of the logic engine.

To realise the design stage assessment of control systems, a parsing module is integrated into the vueOne tool, where various control architectures can be analysed and compared, simultaneously with the system development process. The module can read and analyse FCs and FBs which are written in structured text language which is the one of the five languages supported by the IEC 61131-3 standard for PLCs. During the control code generation, the module automatically reads system components and mapping information taken from the generated source code and displays design conformity scores of each component.

5. Use Case

The proposed theoretical model and its integration with engineering tool is demonstrated with the help of a test rig (Fig. 3). The test rig is designed to showcase and conduct training related to modular automation systems. The basic operation of the test rig is to convey a workpiece from one end of the machine to the other while performing a number of operations such as transferring, indexing, clamping, drilling and gauging, etc. The test rig is composed of four stations, i.e. distribution station, buffer station, processing station and handling station, controlled via single PLC. There are ten actuators with embedded position detection sensors and seven workpiece detection sensors. A digital twin of the test rig is created and commissioned in the VUEONE. The validated virtual model of the rig is then exported to XML file format. As mentioned previously, each actuator and sensor component is assigned with an RCC. For sensors, the design of RCCs depends on the number of output states. While for actuators, the design of FB depends on the number of states and driving power (such as pneumatic or electric) of actuator component. If the numbers of states and driving power are similar, then the same RCC can be replicated for similar actuators.

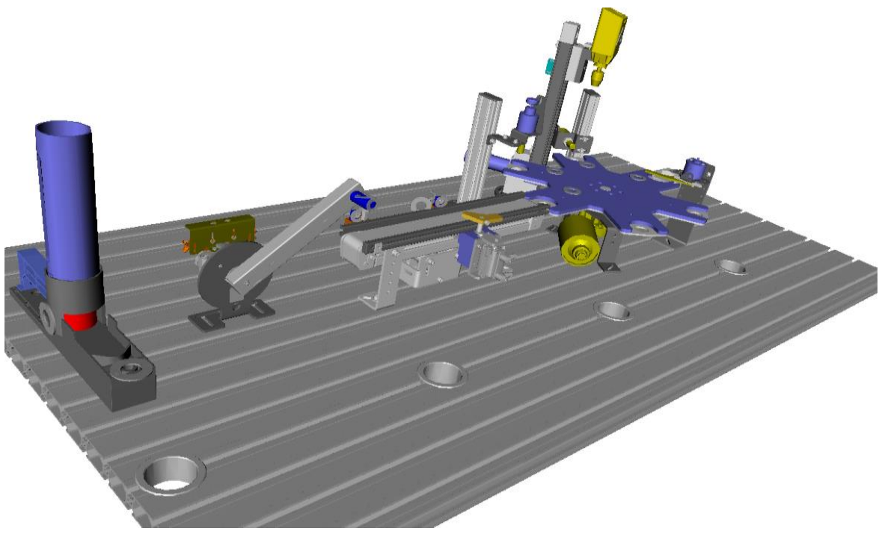

Fig. 3. Test rig's virtual prototype.

Table 1 shows the complexity results of the non-replicated components in the control system. According to the results, all generated components are inside allowable design tolerances, thus validating the control system architecture. It can be seen from the table that the logic engine has the highest complexity (3.022 bits), as it is the orchestrator component controlling the manufacturing execution by monitoring the sequence rules and interlock conditions of system components. Since, the system is built as a centralised network, LE is responsible for most of the communication. Moreover, other FBs have same inter-module design conformity due to the following reasons; similar information flow pattern, invocation of same functions for passing parameters, machine status updates and diagnostic checks, etc. On the other hand, the difference in intra-module design conformity, occurs due to operational behaviour variations. In industrial domain, preferences for intra and inter module component designs can be varied. In most cases, it is preferably to use FBs with low intra-module design conformity since they contain several functionalities for multiple variants. On the other hand, the use of multiple FBs, provides high modularity, but it increases the coupling between the modules subsequently increasing complexity. It is hence necessary to find an optimal area between these two, where the proposed model helps to capture this information.

Table 1. Complexity Assessment Results (W1=W2=0.5)

#	Component	Rep.	N_1	N_2	n_1	n_2	C_{intra}	F_{in}	F_{out}	f_{in}	f_{out}	C_{inter}	I_i (bits)
1	2-state sensor (FC)	7	42	32	10	8	0.846	1	1	1	1	0.987	0.131
2	2-position 5-state pneu. act. (FB)	1	189	297	27	20	0.730	6	4	2	2	0.911	0.294
3	Pneumatic gripper (FB)	1	159	205	25	21	0.799	6	4	2	2	0.911	0.229
4	2-position 5-state elec. act. (FB)	1	185	257	32	20	0.748	6	4	2	2	0.911	0.276
5	2-position 3-state elec. act. (FB)	3	143	199	19	23	0.816	6	4	2	2	0.911	0.214
6	Indexing table (FB)	1	92	90	15	18	0.908	6	4	2	2	0.911	0.136
7	2-state electric act. (FB)	1	64	68	21	29	0.926	6	4	2	2	0.911	0.123
8	3-position 7-state pneu. act. (FB)	2	220	374	30	19	0.666	6	4	2	2	0.911	0.360
9	Logic engine (FB)	1	596	646	23	101	0.136	29	18	17	7	0.111	3.022
	Total												6.355

6. Conclusion

This paper introduces a method for assessing complexity that acts as a design support tool to highlight stress points and compare concept control design alternatives. The proposed measure is integrated with a virtual system development and process planning tool, where auto-generated control codes can be analysed, simultaneously with the virtual validation steps. As a future work, the proposed model will be validated on several industrial test cases of both distributed and centralised control system, to highlight the differences in their design complexity.

Acknowledgment

The authors greatly acknowledge the support from UK EPSRC via Knowledge Driven Configurable Manufacturing (KDCM) research project.

References

[1] Dai, W., & Vyatkin V., A. (2013). Component-based design pattern for improving reusability of automation programs industrial electronics society. *Proceedings of IECON 2013-39th Annual Conference of the IEEE,* (pp. 4328–33).

[2] Harrison, R., Lee, S. M., Ong, M. H., & West, A. A. (2006). Distributed engineering of modular reconfigurable automation systems. *Information Control Problems in Manufacturing*, *12*, 553–8.

[3] Alkan, B, Vera, D., Ahmad, M., Ahmad, B., & Harrison, R. (2016). Design evaluation of automated manufacturing processes based on complexity of control logic. *Proceedings of 26th CIRP Design Conference, Vol. 50*, (PP. 141–6).

[4] Harrison, R., Lee, S. M., & West, A. A. (2004). Lifecycle engineering of modular automated machines. *Proceedings of 2nd IEEE International Conference on Industrial Informatics*, (pp. 501–6).

[5] Harrison, R., Vera, D., & Ahmad, B. (2016). Engineering methods and tools for cyber-physical automation systems. *Proceedings of the IEEE 2016*: *Vol. 104* (pp. 973–85).

[6] Morton, Y. T., Troy, D. A., & Pizza, G. A. (2002). An approach to develop component-based control

software for flexible manufacturing systems. *Proceedings of the 2002 American Control Conference (IEEE Cat NoCH37301): Vol. 6* (pp. 4708–13).

[7] IEC IEC. (2003). 61131-3: Programmable controllers–part 3: Programming languages. Geneva.

[8] Campanelli, S., Foglia, P., & Antonio, C. (2015). Computers in industry an architecture to integrate IEC 61131-3 systems in an IEC 61499 distributed solution. *Computers in Industry,* 47–67.

[9] John, K. H., & Tiegelkamp, M. (2001). *IEC 61131–3: Programming Industrial Automation Systems: Concepts and Programming Languages, Requirements for Programming Systems, Aids to Decision-Making Tools.*

[10] Vyatkin, V. (2011). IEC 61499 as enabler of distributed and intelligent automation : State of the art review. *IEEE Transactions on Industrial Informatics, 7*, 768–81.

[11] Thramboulidis, K. (2012). IEC 61499 as an enabler of distributed and intelligent automation: A state-of-the-art review A different view. *Journal of Engineering*.

[12] Zoitl, A., & Vyatkin, V. (2009). IEC 61499 architecture for distributed automation: The "glass half full" view. *IEEE Industrial Electronics Magazine, 3*, 7–22.

[13] Crnkovic, I., & Larsson, M. (2000). A case study: Demands on component-based development. *Proceedings of ICSE 2000-22nd International Conference on Software Engineering*, 23–31.

[14] Grady, R. B. (1992). Practical software metrics for project management and process improvement. *Information and Software Technology*, 282.

[15] Phukan, A., Kalava, M., & Prabhu, V. Complexity metrics for manufacturing control architectures based on software and information flow. *Computers & Industrial Engineering*, *49*, 1–20.

[16] Suh, N. P. (1995). Designing-in of quality through axiomatic design. *IEEE Transactions on Reliability, 44,* 256–64.

[17] Adamov, R., & Baumann, P. (1987). Literature review on software metrics. *Universität Zürich-Irchel*, *87.*

[18] Halstead, M. H. (1977). *Elements of Software Science.*

[19] Henry, S., & Kafura, D. (1981). Software structure metrics based on information flow. *IEEE Transactions on Software Engineering, SE-7*, 510–8.

[20] Ahmad, B. (2014). *A Component-Based Virtual Engineering Approach to Plc Code Generation for Automation Systems.* Loughborough: Loughborough University.

[21] Kong, X., Ahmad, B., Harrison, R., Park, Y., & Lee, L. J. (2012). Direct deployment of component-based automation systems. *Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, IEEE* (PP. 1–4).

Bugra Alkan received the M.Sc. degree in mechanical engineering from Izmir Institute of Technology, Izmir, Turkey, in 2012. He is currently pursuing his Ph.D. degree in manufacturing engineering at Warwick Manufacturing Group (WMG), University of Warwick, UK. His main research interests include statistical experimental design, design for manufacturing and assembly, performance optimisation in manufacturing systems, assembly process planning and, industrial ergonomics and robotics.

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 modelling, 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. dissertation. 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.

Malarvizhi Kaniappan Chinnathai received the M.Sc. degree in manufacturing systems engineering from University of Warwick, UK, in 2016. She is currently working as a research assistant on AMPLiFII (Automated Module-to-pack Pilot Line for Industrial Innovation), which is a project to bring innovation in battery module and pack assembly, in the Warwick Manufacturing Group. Her main research interests include product variety management, virtual factories, reconfiguration and PLM.

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 a professor of automation systems in the Warwick Manufacturing Group (WMG), University of Warwick, 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."