The ProFlex Methodology: Agile Manufacturing in Practice

Giovanni Di Orio[1], José Barata[1], Carlos Sousa[2], Luís Flores[2]

[1] CTS – UNINOVA, Dep. de Eng. Electrotécnica, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 2829-516 Caparica, Portugal, {gido, jab}@uninova.pt
[2] IntRoSys, SA – Global Control System Designers
2860 Moita, Portugal {carlos.sousa, luis.flores}@introsys.eu

Abstract. Today, manufacturing production systems are required to be responsive, re-configurable, adaptable, and flexible. The fulfilment of these requirements is directly related to the method used to design and develop their control applications. The paper presents the result of an investigation study to capture current techniques adopted by logic designers during the design and development of control applications at IntRoSys. The result of this study is then used as the foundation for the development of a methodology and its related tools for achieving agility in programming manufacturing control systems.

Keywords: Automation, Automotive Industry, Industrial Standards, Controllers, IEC-61131-3, Behavioural Modelling, Code Generation

1 Introduction

Today, European manufacturing industries are under high pressure in the global market due to a "pincer effect" as claimed in [1]: on the one side, the presence of competitors which operate in regions with low-wage and are absorbing very fast the available technologies, and on the other side, the need to keep pace with science-based innovation processes and products that are creating new markets and new business. This effect is incredible felt in the automotive industry, where the increasing demand and competition in market sharing are radically changing the way production systems are designed and products are manufactured in terms of improved flexibility/adaptability, reduced lifecycle costs, high products customization and quality, and reduced time-to-market [2].

In the recent years, the competition between manufacturing companies is going to be shifted from cost reduction to added value in products. The Original Equipment Manufacturers (OEMs) are demanding for more and more exclusive, efficient and effective production systems capable to produce as many different product variations as quickly as possible. Manufacturing companies are striving to introduce flexible and adaptive manufacturing techniques in order to better meet market needs whilst maintaining the low cost base of heavily automated mass production techniques [3]. In this context, the key to competitiveness is represented by the reduction of the

production costs during production system lifecycle and the capability to have systems that are able to quickly respond to markets variations and demands for new products. However as stated in [4], the capability of offering more variants per product, and introducing new products faster, is constrained by the adoption of technologies and equipment of mass production operations, implying that frequently changes in products may potentially cause changes in the plant's hardware structure and its functionality.

As existing plants cannot be rebuild due to high investment volumes, the changes in hardware components affect particularly the control software. Therefore, control software is the stage of continuous modifications during the production system lifecycle which implies an increasing in its amount and complexity [5]. Furthermore, every hardware/software change in the production environment has heavy consequences on the normal flow of operations: the production system needs to be shut-down, mechanically changed and reprogrammed in order to face the new configuration. The (re)programming task is often commissioned by the OEMs to external suppliers that are then responsible for fast and secure rump-ups. The time needed for this task is directly proportional to the dimension of changes, presence of a well-suited documentation, particular familiarity with the standard used by the OEMs, presence of an up-to-date control software version [6]. Moreover, the presence of rigid industrial proprietary standards, encompassing electrical systems and diagnostic as well as hardware types and control software structure, has two fundamental implications as exposed in [7]. From one side, standards contribute to reduce the flexibility of the programmer while discouraging any technique/technology other than the defined in the standard. From the other side, rigid and strong software structure aids the development of tools and methodologies for automatically generating the source code.

The present paper is motivated by the need to conduct an analysis of current methods for implementing monitoring and control solutions in the automotive industry in order to outline the challenges faced by the system integrators. The paper also serves as a starting point for research community for providing the baseline of current industrial software design and development practices. The baseline has then been used as the foundation of the ProFlex methodology intended to enhance the (re)programming task of control software during the shut-downs.

2 Contribution to Collective Awareness Systems

To face globalization challenges manufacturing industry needs to move towards new paradigms and methodologies aiming at enabling distributed knowledge creation and intelligent usage of manufacturing production data from the production environment. The main goal is creating awareness about relevant problems faced by industry in order to elaborate solutions by exploiting collective efforts. To remain competitive, manufacturing companies need to understand where and how to change. This paper aims at contributing to collective awareness by providing a theoretical background about the way current monitoring and control solutions are designed and developed and, most important, presenting a way to model manufacturing components/resources.

The findings of this paper have the intent to open the door to the introduction of new technologies and paradigms in industry that are expected to support environmentally awareness by framing the current practices for designing and developing control applications for the automotive sector.

3 Review on Current Design Methodologies

Today, programmable logic controllers (PLCs) represent the *de-facto* standard for implementing control and monitoring solutions in many manufacturing companies due to their features and characteristics. Since PLCs are established as the device of choice for automation. Then a good starting point for gaining background information and knowledge on how to develop PLC programs is the introduction of the International Electrotechnical Commission (IEC) 61131-3 standard.

3.1 IEC 61131-3 Standard

As stated in [8], after the introduction of the PLCs in 1969, various companies developed its own platform with different run-time environments, operating systems, and programming languages. To reduce the complexity for the PLCs users and their fragmentation, the IEC 61131-3 standard has been published.

The IEC 61131-3 contains five programming languages for developing programs to be executed inside a PLC, which are supposed to be supported by any compliant vendor. The main goals of the IEC 61131-3 standard is to unify all the programming concepts for developing industrial control applications while abstracting from proprietary peculiarities and creating a common user experience when programming and configuring industrial controllers. The supported programming languages are:

1. Instruction List (IL) – A textual programming language similar to assembly and consisting of simple operation codes.
2. Structured Text (ST) – A programming language with syntax similar to Pascal, Basic or FORTRAN suited for programming more complex algorithms.
3. Functions Blocks Diagram (FBD) – A graphical language providing a mechanism for encapsulating functionality into a module with a common external interface, and designed to promote the code reusability.
4. Sequential Functional Chart (SFC) – A graphical language derived from the GRAFCET. It is made up of graphical elements called steps and transitions; a step represents a specific state in process and/or machine sequence while a transition represents the set of conditions that allows the evolution from one step to another.
5. Ladder Logic Diagram (LLD) – A graphical language derived from the electrical relay diagrams and created to be easily understood by booth engineers and technician without any software skill.

Most of the industrial applications are designed using one of these languages. However as exposed in [9], and recently confirmed a poll realized by the Control

Engineering U.S. and Control Engineering Poland magazine [10], 96% of industrial developers uses LLDs. FBDs are the next most popular at 67%, followed by IL (37%). The large predominance of the LLD language for developing industrial applications is due to several factors that are well exposed in [11]. However, the main goal in this scope is not the evaluation of current practices for developing monitoring and control solution, but only their analysis in order to enable the identification of the challenges related to the development of logic control programs. With this in mind, this analysis represents the foundation upon which new methodologies can be designed for improving the control logic development process.

3.2 Control Logic Development Process

As stated in [7], if from one side there is an extensive published material outlining the code structures available to the practitioners of programmable controllers, from the other side, there is little reported on how these techniques are really applied for developing ready-to-use and above all standard compliant industrial control applications.

A practical perspective on the design and development process for control logic has been firstly given by Lucas and Tilbury in their work [12]. During their investigation, conducted from September to December 2001 into logic design techniques for automotive industry, Lucas and Tilbury have noticed that the control logic, needed to monitor and control a manufacturing process/machine, is obtained by gathering and combining information from three sources, namely: project specific specifications, unspecified requirements and standard specifications (see Fig. 1).

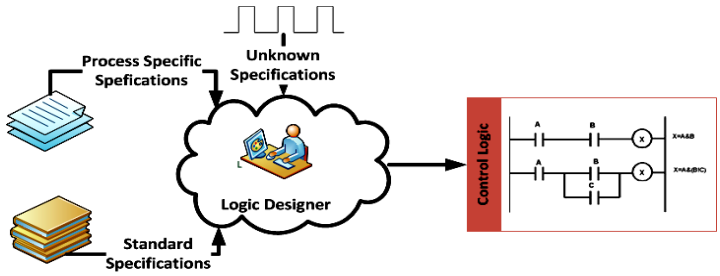

Fig. 1. Control Logic Development Process [12].

The project specific specifications comprise several documents that are primary used during the control logic development activity, namely: project specifications (functional requirements), mechanical drawings, electrical drawings and schematics, printout of previous projects, PLC memory map and process/machine timing bar (used to specify the desired behaviour of the process/machine along time during production activities). At this point, logic designers are responsible to integrate fragmented piece of information, extracted from these documents, and combine it with standard specifications (usually in the form of old project) to create the control logic. Finally, the control logic is refined taking into account the unspecified requirements that, in turn, can include late changes and/or unexpected constraints in the process/machine.

4 ProFlex Methodology

4.1 Research Activity

In order to gain a solid knowledge about current control logic development process for the automotive industry, a set of observations were conducted at IntRoSys SA in Portugal. Since 2004 the company has established itself in the market as one of the reference Controls Houses for the European Manufacturing Industry.

The observations were targeted at capturing current practices and activities followed by system integrators when creating control logic. The set of observations will be, then, used as inputs for developing an innovative methodology to improve the control logic development process by pushing software engineering techniques in the world of automation.

The investigation revealed that the control logic development process is today still the same as the one depicted by Lucas and Tilbury. The entire set of documents analysed and used by the logic designer during the development process is the same for all the car manufacturers meaning that all the information needed to create the control and monitoring logic can be extracted from them. Moreover, the particular software structure can be derived from standard template projects provided by the car manufacturers. Finally, new control and monitoring logic could be implemented on-site if late changes in process and/or machine are realized. In this context, the main activities carried out by logic designers are four, namely: Documents analysis, Modification of standard template functions/projects, Human Machine Interface (HMI) development and new code Development.

The fulfilment of the first three activities leads to 90% of control and monitoring logic ready to installation and test. The other 10% includes modifications on-site necessary to fit the logic to the real state of things. The conducted investigation has been used as the basis for the formation of the ProFlex methodology by providing necessary information on how to enhance the control logic development process.

4.2 Proposed Methodology

The methodology proposed in this paper address the automatic generation of standard compliant and ready-to-use control and monitoring logic starting from the behavioural model of the resources composing the considered manufacturing process. More extensive explanation of the methodology can be found in [13].

The proposed methodology is depicted in Fig. 2. The main activities of the methodology are five, namely:

- A0: Monitoring and Control system requirements analysis.
- A1: Identification of the resources of the considered manufacturing process.
- A2: Construct a universal behavioural model of each resource.
- A3: Software Generation.
- A4: Deployment and Tests.

The activities are organized in two macro activities the **Model Development Cycle** and the **Control Development Cycle** respectively. The activities A0, A1 and A2 are

executed during the Model Development Cycle while the activities A3 and A4 are executed during the Control Development Cycle.

Furthermore, inside the Model Development Cycle, the **Agentification Cycle** is executed. During the Agentification Cycle real manufacturing resources are considered as autonomous components which behaviour is modelled using a unique and formal representation. The result of this activity is a self-contained piece of software (agent) to be executed inside the PLC and, thus, implemented using one of the five languages of the IEC-61131 standard.

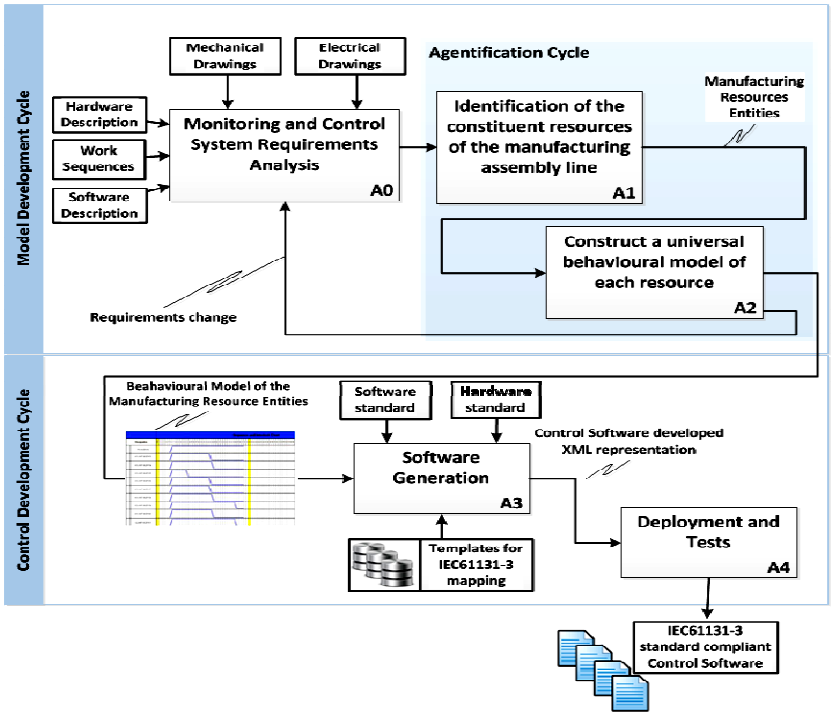

Fig. 2. Conceptual model of the Proposed Methodology.

The methodology is focused on a kind of "lingua franca", represented by the behavioural model of the system resources, as a way to enable fast coding, improve industrial acceptance of developed code and, most important, reduce the error during the programming activity.

4.3 A Tool for Applying the Model

The following tool has been implemented to assist the logic designers in applying and implementing the ProFlex methodology (see Fig. 3). The tool is in the form of an Excel spreadsheet providing a table which will be used to model the behaviour of the manufacturing resources inside the considered manufacturing process.

The behaviour of the resources is modelled using impulse diagrams that enable the description of the temporal course of states of system resources together with signals among them to activate state changes. To describe the value sequences and relations of signals over time, a global timescale is considered.

4.4 Case Study

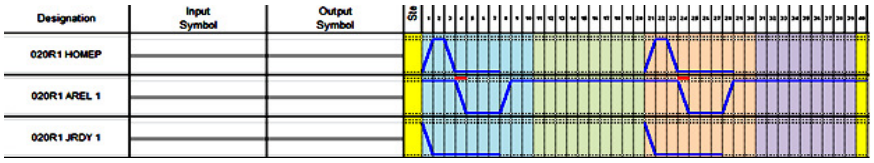

Fig. 3. Excel Tool for applying the ProFlex methodology.

In order to validate and assess the proposed methodology, two robotized manufacturing cells located at IntRoSys have been used. The cells include a loading station used to load car components, a robot manipulator and a welding station. The cells are identical in terms of functionalities, i.e. once a part is loaded the robots perform the same actions on it according to the sequence: pick-weld-place. However, they have been designed and developed using two different proprietary standards, from two distinct car manufacturers. The usage of different standards directly implies the way the control software is organized, structured and built, even if the process for developing the control logic is the same. As a matter of fact, the project specific specifications are analysed by the logic designer and all the acquired information is directly parsed into LLDs. The time and the effort needed to develop the control logic strictly depend on the logic designer experience, familiarity with the particular used standard, and the way the control logic is organized. In this context, the ProFlex methodology has been applied to develop the control logic for both the presented cells. In a first moment the **Model Development Cycle** activity is applied aiming to identify the system requirements and according to them to design the behavioural model of the resources of the robotic cell (during this activity the Excel spreadsheet is used Fig. 3). Once the behavioural model is built then the **Control Development Cycle** activity can be executed. This activity is completely automatic and aims to create a standard compliant ladder logic code based on the designed behavioural model.

According to a poll realized at IntRoSys, the methodology enables potentially the reduction of the overall time required to create the control logic and the possible errors related to this activity. Moreover, the initial modelling activity permits to abstract the development process from the particular OEMs standard used, while equalizing the effort needed during the development activity.

5 Conclusions

The paper present a software engineering approach based on a behavioural model of manufacturing resources for developing PLC-based OEMs standard compliant control

logic. The way manufacturing resources are modelled, is the result of the research activity on current control logic development process. The key element and innovative aspect of the ProFlex methodology is the usage of a unique model to generate different structured and organized control logics. Further, a case study has been presented with the goal to show the feasibility and repeatability of the proposed approach. Finally, the approach, denoted with the acronym ProFlex, makes the control systems satisfy requirements for flexibility and reconfiguration while offering competitive advantage to system integrators by enabling faster and securer ramp-ups. As a matter of fact, any change/modification in the hardware configuration of the manufacturing process/machine simply triggers a new modelling activity.

Acknowledgments. This work was funded by the Regional Operational Programme of Lisbon (POR Lisboa), in the scope of the National Strategic Reference Framework of Portugal (QREN), part of the European Regional Development Fund (FEDER). This work is also supported by FCT Fundação para a Ciência e Tecnologia under project grant Pest-OE/EEI/UI0066/2011.

References

1. F. Jovane, E. Westkämper, and D. J. Williams, *The ManuFuture Road: Towards Competitive and Sustainable High-adding-value Manufacturing*. Springer, 2009
2. F. Rosiná and S. Temperini, "Advanced Maintenance Strategies for a Sustainable Manufacturing,"*10th IFAC Workshop on Intelligent Manufacturing Systems*, Lisbon, 2010
3. V. Hajarnavis and K. Young, "An Assessment of PLC Software Structure Suitability for the Support of Flexible Manufacturing Processes," *Autom. Sci. Eng. IEEE Trans.*, vol. 5, no. 4, pp. 641 –650, Oct. 2008
4. G. Michalos, S. Makris, N. Papakostas, D. Mourtzis, and G. Chryssolouris, "Automotive assembly technologies review: challenges and outlook for a flexible and adaptive approach," *CIRP J. Manuf. Sci. Technol.*, vol. 2, no. 2, pp. 81–91, 2010
5. T. Wagner, "Applying Agents for Engineering of Industrial Automation Systems," *Springer*, pp. 1097– 1097, 2003
6. J. Barata, *Coalition Based Approach For ShopFloor Agility*. Amadora: Orion, 2005
7. V. Hajarnavis and K. Young, "An investigation into programmable logic controller software design techniques in the automotive industry," *Assem. Autom.*, vol. 28, no. 1, pp. 43–54, 2008
8. A. Otto and K. Hellmann, "IEC 61131: A general overview and emerging trends," *IEEE Ind. Electron. Mag.*, vol. 3, no. 4, pp. 27–31, 2009
9. M. R. Lucas, *Understanding and assessing logic control design methodologies*. University of Michigan., 2003
10. K. Pietrusewicz and L. Urbanski, "control system software programming," *Control Eng.*, vol. 59, pp. 32 – 37, 2012
11. T. Walter, "Ladder logic: Strengths, weaknesses.," *Control Eng.*, Mar. 2007
12. M. R. Lucas and D. M. Tilbury, "The practice of industrial logic design," in *American Control Conference, 2004. Proceedings of the 2004*, 2004, vol. 2, pp. 1350 –1355 vol.2
13. G. Di Orio, J. Barata, C. Sousa, and L. Flores, "Control System Software Design Methodology for Automotive Industry," presented at the IEEE International Conference on Systems, Man, and Cybernetics - IEEE SMC 2013, Manchester, (UK), 2013