Virtual Engineering Framework for Automatic Generation of Control Logic including Safety

Adnan Khan*, Petter Falkman*, and Martin Fabian*

Abstract— **In this paper a new virtual engineering framework based on *clever components* for automatic generation of control logic including safety is presented. Manual practices of modeling plant components and writing PLC programs is an error-prone and time consuming task. This new virtual engineering framework enables automatic generation of control solutions based on cloud based repositories, containing clever components with formalized logic descriptions provided by vendors. Consequently reduced time for virtual engineering and commissioning can be achieved by avoiding current manual practices. Other advantages include testing and validation of PLC logic and plant models in the engineering phase, and less human errors due to automatic generation of both plant models and logic. In addition, this framework will help in the development of more reliable and robust safety logic and assist the procedure of issuing *safety certificate*.**

I. Introduction

In the manufacturing industry, *virtual engineering* is the process of using computer based models to prepare, develop, implement, and test a manufacturing system. Different softwares are used in the engineering phase to implement various types of models and logic. Some examples are CAD models, kinematic models, robot programs etc. Building models in the engineering phase helps in early verification and validation of different specifications of a manufacturing system. But all system properties are tested in a decoupled manner i.e. CAD, kinematics and robot programs etc are tested separately. After the engineering phase, comprehensive behavior models are implemented to test control logic.

This verification of control logic against behavior model is known as *virtual commissioning*. Virtual commissioning is carried out separately outside the virtual engineering chain, as the information/data gathered from different tools in the engineering phase is heterogeneous (CAD, Kinematics etc.). Different tools used in the engineering chain have their own data formats, due to which the data from one tool cannot be re-used by other tools and seldom is the data in a computable format. Due to this issue, simulation and modeling engineers have to model complete behavior of the manufacturing system again to carry out virtual commissioning. And this is the typical case, even though by integrating virtual commissioning in the virtual engineering chain carries benefits as described by the authors of [1], [2].

This work has been carried out at the Wingquist Laboratory VINN Excellence Centre within the Production Area of Advance at Chalmers. It has been supported by Vinnova FFI VIRTCOM (ref 2014-01408), and ITEA3 Vinnova ENTOC (ref 2016-02716).

*Department of Electrical Engineering, Chalmers University of Technology, Göteborg, Sweden {kadnan, petter.falkman, fabian}@chalmers.se

Commissioning practices can be classified in four major categories [3]:

- Real controller with real plant (Physical commissioning)
- Real controller with virtual plant (Virtual commissioning)
- Virtual controller with real plant
- Virtual plant with virtual controller

For the purposes of this paper, the top two categories will be described, and the other two will not be mentioned further. Please see [3] for interesting points about those approaches.

Virtual commissioning, which is routinely used nowadays in many companies, involves a real physical controller and a simulated plant, this is also known as HiL, *hardware in the loop*. In order to perform virtual commissioning, a detailed plant simulation model is built and a physical controller is connected to the simulation so that commands from the controller will initiate responses from the simulation. Thus, the simulation model needs to be complete and detailed down to the level of sensors and actuators [3], [4]. The control logic can then be developed and tested against the simulation model, so that errors found can be fixed before any physical commissioning.

The main focus of virtual commissioning is on the correctness of the control functions before physical commissioning. These functions are typically implemented in programmable logic controllers, PLCs [5]–[7]. Using virtual commissioning to implement the control logic is beneficial, as many tests do not have to be made on the factory floor, which decreases the risk of damage. During physical commissioning, the control programs are tested and corrected manually on the shop floor, with the machinery running slowly and the operator having one hand on the emergency stop button. This is time consuming and carries no guarantees that the final result is correct under all circumstances; only under those circumstances that actually arise during the physical commissioning phase can there be some confidence about the correct functionality. For virtual commissioning, since a computer model is available, computational approaches, such as formal methods, can help to guarantee full correctness of the control logic [2].

Control logic for manufacturing plants can be broadly classified in to two categories:

- Control logic for nominal aspects
- Control logic for safety aspects

The control logic for the nominal behavior is implemented to control the plant behavior under standard operating conditions, while she safety logic is implemented to control a plant's behavior under critical conditions to avoid machine breaking and human injuries.

Currently in manufacturing industries, dedicated programmable systems, so called *safety PLCs*, and specific actuators and sensors are used to guard humans from hazards. Verification of safety logic is typically performed manually during the physical commissioning. After the verification, which is usually done by a third party engineer, a *safety certificate* is issued by the *safety validator* confirming that the safety logic has been tested and verified. However, it is hard to know that all possible scenarios have been tested and verified, and the checklists for safety logic verification are generated based on design documentations manually. Manual generation of checklists is a tedious and error prone task.

The authors of this paper believes that virtual commissioning would be beneficial allowing the safety engineers to test and verify safety scenarios in much the same way as the code for the nominal behavior is developed. This requires of course that the virtual commissioning simulation model appropriately includes the plant's behavior under critical conditions; it must for instance be possible to inject sensor and actuator failures. This is currently not the case, typically, mainly due to the amount of time and effort necessary to build such a model.

Though the above benefits of virtual commissioning is and can be made real, the main obstacle is building the detailed simulation model. This is a cumbersome, mainly manual task that many companies refrain from since the gains are perceived not to outweigh the costs. However, if the virtual commissioning simulation model was easier to build and in addition could be used for more than just virtual commissioning and safety logic development, then things would be different. To make the manual modeling and programming practice easier and faster, ready made models of appropriate detail is a requirement. A similar approach is discussed by [8], but it does not treat safety and virtual engineering.

The presented virtual engineering framework using vendor provided *clever components* will help engineers to generate models and PLC logic quickly. Using the proposed framework virtual commissioning is integrated in the virtual engineering chain and can be carried out at any stage of the project cycle. As a result, speedy modeling, logic generation and verification including verification of safety PLC logic can also be performed in the virtual environment.

A. Contribution

In this paper, a novel virtual engineering framework is proposed. In the proposed framework, it is assumed that vendors will provide components with all geometric, kinematic, dynamic and formalized logical descriptions, so called *clever components*. As a result, virtual engineering and virtual commissioning will be joined together and performed iteratively. This means that there will be a single plant model used for engineering, optimization, simulation, and virtual commissioning.

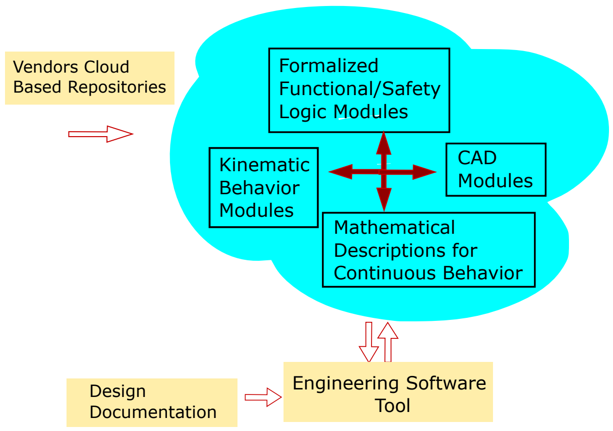
Vendors Cloud Based Repositories
Formalized Functional/Safety Logic Modules
Kinematic Behavior Modules
CAD Modules
Mathematical Descriptions for Continuous Behavior
Design Documentation
Engineering Software Tool

Fig. 1. Proposed Virtual Engineering Framework

The framework is aimed to counter the problems associated with manual practices of modeling and PLC programming. The idea is to tap, not only component models or control logic for nominal behavior, but also safety logic. The proposed framework will result in the following benefits:

- Automatic generation of complete control solution (drag and drop)
- Comprehensive testing of human and robot interactions
- Virtual commissioning/verification can start already in the engineering phase
- Overall reduction in commissioning time
- Less risk of human error due to automatic generation

B. Outline

This paper is structured in the following way: In Section II, the virtual engineering framework see Fig. 1 based on clever components is presented and key elements related to this framework are explained using an example cell see Fig. 2. In section III, industrial practice regarding safety logic verification is described and how the proposed framework can help in this regard is explained. Section V concludes the paper with potential future work direction.

II. Framework Overview

The proposed virtual engineering framework see Fig. 1 for automatic generation of control logic including safety is inspired by [9].

A. Virtual Engineering Design Documentation

The term *plant* covers a collection of all the different equipments, resources, and machines etc. set up in a tactical way to produce or manufacture certain products. For physical commissioning before setting up a plant, engineers from diverse backgrounds and fields of study, mainly electrical, mechanical, control and safety generate design documents manually based on the requirements in the planning and engineering phase.

Each equipment/resource are made of different components, which can range from simple screws, metal plates, sensors, buttons, actuators etc. to arbitrarily complex constructions. Such a component can be described by number of different aspects, such as logical behavior, geometry of the component, kinematic behavior, safety aspects, to name a few.

Currently in industries, there are a number of informal tools used for design process of control logic. These informal tools include piping and instrumentation diagrams, verbal descriptions of the process to be controlled, and sketches [10]. Using verbal descriptions and diagrams to specify logic or behavior is a potential cause for errors, as these can be interpreted differently. These errors are then usually corrected during physical commissioning, consequently adding more time.

This added time can be reduced or even avoided if the design documents are formalized. At the moment, research carried out in the area of virtual engineering provides no encouraging proof of using formal methods for complete virtual engineering including safety logic.

In the proposed framework, it is assumed that the design documentation will be formally specified, so that it can be deciphered by the engineering software. Formalization of design documentation will allow the automatic generation of control solutions in the virtual engineering phase. The requirements in the design documents along with vendor provided clever components can be used to generate a formal model of the plant. The generated formal model will provide the basis of generating different checklists for safety logic verification. Hence, the formalization of design documentation is an important aspect of the proposed framework.

B. Clever Component Repositories

A *clever component* is a collection of information modules defining all the different aspects of the component which enables it to be used by software algorithms. The clever components provided by different vendors will have the *clever* property to interact with each other using the formalized design documentation entered in the engineering software. Each clever component is described by the following essential information modules:

- CAD models
- Kinematic models
- Mathematical description of continuous dynamic behavior
- Formal control logic description for nominal aspects
- Formal safety logic description

The CAD models basically describe the dimensions of the component (a 3D drawing of the component with all its parts and dimensions). It provides all the geometrical details of the component (mechanical and electrical). The kinematic models, on the other hand, describe how different parts of the CAD model interact and behave in the real world. The continuous dynamic behavior describes essential mathematical description e.g. equations governing the system. The control logic descriptions for both nominal and safety logic describes the logical behavior of the clever component, which can be expressed in a variety of high level logical description, for details and example see [11].

The formalization of the logic description can allow the logical behavior to be interpreted by a software for PLC logic development in an automatic way. There are a number of academic references providing encouraging results that motivates the use of formal methods for both plant models and PLC logic verification, to name a few [11]–[18].

The references mentioned above provide encouraging conclusions for vendors to provide clever components along with associated formalized control logic description (for both nominal and safety aspects). The information modules mentioned above can be stored in cloud based repositories and can be retrieved as per requirement during plant modeling.

The clever components will aid the engineers in modeling the plant; based on the requirements mentioned in design documents, the user can select the clever components from the repositories and modeling of plant can begin instantly. Engineers in the engineering phase will only have to drag and drop the component modules into the engineering software, instead of manually modeling each component used in the plant. As a result, modeling efforts are significantly reduced and engineers having no or little background can also build a plant model.

The formalized description of control logic will provide the starting point of PLC programming for nominal aspects and due to information formalization, software can translate the logic directly to PLC specific code. Each formalized logic module present in the repository will be specific to the clever component. So when a clever component is selected, the associated formalized logic description will be selected automatically, unlike in current engineering practices, where implementation of PLC logic starts after modeling and simulating the complete plant.

Similar to above, the associated safety logic modules of clever components can be retrieved into the local library. As a result, both formal verification and testing of human safety logic can be carried out in the virtual environment (model based testing) in the engineering phase. This practice is more automatic in nature and can also be performed by simulation and modeling engineers, who may lack in-depth knowledge of PLC programming and risk analysis.

C. Clever Component Parametrization

In today's virtual engineering practices, values and parameters related to the components are assigned after building the components manually. During plant modeling, clever components will provide the basis for engineers to assign parameters directly, already in the engineering phase. After finalizing the required clever components from the clever components repository space, components can be parametrized according to the final design goal. Parametrization may require either adding more detail to the clever component or removing/changing some details depending on the requirements.

This may include the following activities:

- Assigning constants or variables to clever components;

- Input and Output assignments to PLC logic;
- Removing or adding analog values for continuous behavior

The purpose of this activity is to tailor the model developed from clever components, so that the plant model with control logic gives the required behavior. After parametrization, the modified component can be saved in the repository of clever components. This provides variation of same component in terms of usage based on parameters and application.

D. Engineering Software

For automatic generation of complete control solutions, the formal specifications of clever components have to be understood and deciphered by some software tool. In the proposed framework, it is assumed that both plant modeling and PLC programing will be carried out in an *Engineering software*. As each clever component will have different modules (CAD, kinematic, logic etc.), each information module will be translated by the engineering software so that both plant modeling and PLC programming tasks can start under one roof.

The engineering software will also enable different clever components to communicate with each other based on relation and or association. The communication (relation and association) among different clever components is established through the formalized design documentation. The design documentation entered in the engineering software will provide the basis of clever components interaction. When a clever component is selected, other clever components associated with the selected clever component will appear as options for the user, hence assisting the plant modeling process. As a result, a complete plant model (formal) can be created in a very short time.

Similar to above, the high level formalized description for PLC logic will be translated, the engineering software will help in the translation from the high level description to the specific PLC language. This direct translation will help the engineers to automatically generate PLC logic for the generated model.

E. Example

To further elaborate the proposed framework and how it will help the engineers in the engineering phase, a simple example will be given, providing an overview of how this framework will change the existing virtual engineering and commissioning practices. Consider a cell in a plant, which is to be modeled, the cell consists of a robot with a gripper, a conveyor, a rack, a light curtain, and a drilling unit. This cell is depicted in Fig. 2.

In terms of operations to be performed, the robot has to pick a part placed on rack *X* by the operator and place that part on the drilling unit, so the drilling operation can be executed. When the drilling operation has been completed, the robot will pick the finished part and put it on the conveyor.

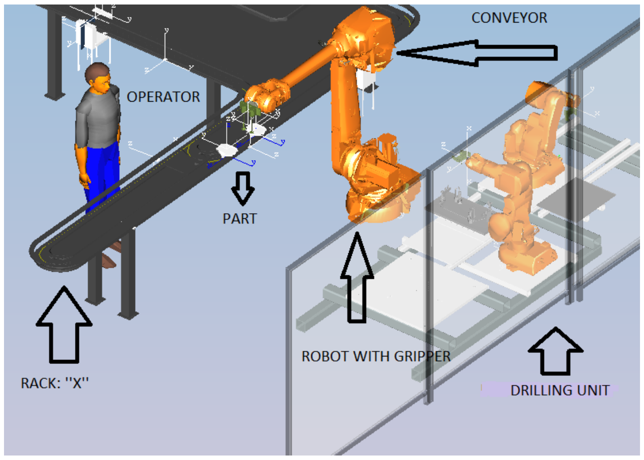

Fig. 2. Example Cell: plant model.

The modeling starts in the engineering phase, the user will start selecting clever components from cloud based repositories based on *design documentation*. All the related information modules (CAD, kinematic, formal logic description etc.) can be retrieved from the corresponding repositories. After retrieving the ready-made clever components, the user can drag and drop the components in the engineering software.

Due to the presence of CAD and kinematic information, the user does not have to worry about each component's geometrical aspects and parts interaction, as the engineering software will govern and direct part placements and interactions based on the information already associated with each clever component and with respect to the formalized design documentation. For example, when the robot component is retrieved, and placed in the cell, the software will ask the user to place a gripper with it, due to its association with the robot so that the gripping operation can be carried out. Similarly the conveyor parts and components will be placed. When different parts of the conveyor are retrieved, other associated parts (e.g stoppers, proximity sensors, etc.) will appear and can be retrieved by the user. As time progresses, the plant model becomes detailed due to the addition of more clever components and this task can be carried out by an engineer having no or little background in modeling.

At the same time, control logic information associated with each clever component can be directly translated by the software and PLC logic development can be initiated by an engineer already in the engineering phase. Finally, the parametrization of both clever components and PLC logic is carried out. The appropriate variables and I/Os are assigned to all clever components in the model and to the associated PLC logic. Now the model is ready for testing. Later with the passage of time other clever components can be added to the same plant model, as a result virtual commissioning and validation tasks can be carried out at any stage during virtual engineering.

III. Safety Logic Verification and Validation

As mentioned in Section I, this framework will not only help in automatic generation of complete control solution but also in the earlier verification (*are we building the product right?*) and validation (*are we building the right product?*) of safety logic in the engineering phase [10].

A. Industrial Practice

The verification and validation of the safety system and safety logic is typically done manually using checklists and visual inspection [19]. All the checklists for the verification are prepared in detail in conjunction with the equipment to be tested and is managed by *safety personnel*. The checklists are defined by control experts in manufacturing engineering and are based on regional standard. These lists are also updated by the authorities with respect to programming standards and requirements.

The safety personnel is one or more authorized persons from one of the engineering departments within the company. The responsibilities of the safety personnel include maintaining safety documents related to verifications, checklists, development process (V-model), and buy-offs.

During the commissioning, a *certified engineer* is assigned by the lead engineer to provide technical support and to perform the verification of the safety circuits, safety software and safety devices. The certified engineer can be third party personnel hired to give support, or the company's own employee. The assigned engineer must have sound knowledge of control engineering, the company's hardware and software standards and should be experienced in verification of safety systems. The certified engineer verifies the following aspects of the safety code:

- Completeness and accuracy of safety code
- Implementation of predefined "standard non-editable routines"
- Correctness of I/O mapping and configurations
- Implementation of correct interlocks
- Exclusive implementation of safety tasks and routines from nominal tasks

The validation task in industries is performed by a *certified validator*, who is an engineer either from the same company or from a third party. The certified validator performs additional functional and dynamic tests based on design documentation, regional control organisation norms and regulations. After the validation task, a certificate of correct and successful validation is provided to the company.

B. Using a Virtual Commissioning Model

The underlying problem with the verification and validation procedure mentioned in Section III-A is that the checklists are generated manually according to the system specifications so the risk of human error is always there. Also, most of the time the manufacturing company personnel do not know what has been tested from the lists and what has not.

Currently to verify and validate the complete safety logic in a virtual commissioning model is a difficult task. As some tests mentioned in check-lists involves real external/internal hardware interruptions (e.g.taking some wires out from the I/Os to test logic, testing logic in case of failure of internal components) etc. Such safety scenarios can only be authenticated in the real world.

In the proposed framework, where models will be built using clever components, it is assumed that the vendor's provided clever components will contain all functional details of real components including safety logic aspects, allowing generation of comprehensive and realistic checklists for verification of safety logic. The models generated using the proposed framework in the virtual environment will be authenticated in a scientific/logical manner due to the use of formal specifications. This will allow the virtual verification and validation practice to be standardized.

The proposed framework can help in this regard by giving a baseline for generating different checklists and test cases based on the requirements described in formally specified design documentation of real plant. As the logical behavior of clever components will be formally specified, a complete formal model of the plant can be synthesized [20]. The information regarding positions of resources, parts, parts interaction and logical behavior can be extracted from the formally specified design documentation mentioned above and then the engineering software will do the translation automatically. The complete plant will be modeled in the engineering software depicting all sequences of operations [21]. By creating a complete formal plant model in terms of sequence of operations will allow generation of checklists as operations to be tested.

After creating the formal model, verification and validation of the safety functions and logic can be carried out on the virtual commissioning setup according to the check-lists generated in the engineering phase. Each test performed will invoke one or more sets of operations in the formalized plant model. This list of executed operation will be saved and time stamped in the engineering software, hence providing a proof of each executed safety operation to the manufacturing company personnel.

The generated checklists along with tested PLC logic in the virtual engineering phase can then be given to the safety engineer. Now, the safety engineer can implement the tests on real plant during physical commissioning from the check-lists provided. The engineers working at the manufacturing company will have the lists of executed tests in the virtual environment as a record, which will eventually be compared with the results of verification on real plant provided by the safety engineer after physical commissioning. Finally, in case of successful validation the plant model gets validated and the manufacturing company will know that all the safety aspects are verified and validated.

C. Verification and Validation Use Case

Consider the same cell as in Fig. 2. To fulfill the safety specification of the cell, the clever component related to the light curtain will be retrieved in the software using the proposed framework like other clever components mentioned above (robot with a gripper, conveyor, drilling unit and rack).

The safety specification for the given cell is: *both drilling and gripping operations should stop when an operator is present in the cell*. This safety specification is supposed to be fulfilled as a pre-condition for drilling and gripping operations for the given cell. By using this framework, PLC interlocks can be generated for each operation automatically instead of manually doing it for each operation.

Based on the above mentioned safety specification, a checklist will be generated. For verification and validation of the safety logic, the same PLC logic generated for the plant model in the engineering phase along with the checklist will be given to the verifier. The verifier will connect the real plant to a real controller.

After setting up the connection, the testing can begin based on the check-list made for the real plant. To check the safety logic regarding the drilling operation and gripping operation, the scenario of an operator entering the cell will be mimicked in the real plant while the drilling and gripping operation is being carried out. If the safety verifier finds the implementation to be a success on real plant, it will be informed to the manufacturing company's personal in writing. Confirming the implementation of suggested tests mentioned on the generated checklist during virtual engineering phase, hence validating the plant model developed.

IV. Framework Implementation and Significance

The advantage of the proposed framework is in terms of reduced errors due to automatic generation of complete control solution. In addition, the modeling and programming efforts will be significantly reduced due to vendor provided clever components.

As mentioned above, formalized descriptions of clever components will be provided by different vendors. In terms of implementation, the major issue lies in defining a *standard* for developing clever components. If vendors does not specify clever components in a uniform format using a specific *specification language*, then the engineering software will not be able to decipher different clever components.

Another issue is related to identification of design documents which needs to be formalized. A detail survey is required in order to identify the documents used in the industrial sectors for implementation of both nominal control logic and safety logic.

V. Conclusion

In this paper a new virtual engineering framework based on clever components for automatic generation of control solutions including safety is presented. The proposed framework will help in reducing time consumed in plant modeling and writing PLC control programs due to current manual practices. In addition, the proposed framework can help in automatic safety logic generation, verification and validation during the virtual engineering phase. In order to implement this framework following future research work direction has been identified:

- Further elaboration of clever components for vendors
- Procedure of formalization of engineering documents

References

[1] S. Seidel, U. Donath, and J. Haufe, "Towards an integrated simulation and virtual commissioning environment for controls of material handling systems," in *Proceedings of the winter simulation conference*. Winter Simulation Conference, 2012, p. 252.

[2] M. Dahl, K. Bengtsson, P. Bergagård, M. Fabian, and P. Falkman, "Integrated virtual preparation and commissioning: supporting formal methods during automation systems development," *IFAC-PapersOnLine*, vol. 49, no. 12, pp. 1939–1944, 2016.

[3] C. G. Lee and S. C. Park, "Survey on the virtual commissioning of manufacturing systems," *Journal of Computational Design and Engineering*, vol. 1, no. 3, pp. 213–222, 2014.

[4] P. Hoffmann, R. Schumann, T. M. Maksoud, and G. C. Premier, "Virtual commissioning of manufacturing systems a review and new approaches for simplification." in *24th European Conference on Modelling and Simulation (ECMS 2010)*, 2010, pp. 175–181.

[5] O. Mathias, W. Gerrit, D. Oliver, L. Benjamin, S. Markus, and U. Leon, "Automatic model generation for virtual commissioning based on plant engineering data," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 11 635–11 640, 2014.

[6] Z. Liu, C. Diedrich, and N. Suchold, *Virtual Commissioning of Automated Systems*. INTECH Open Access Publisher, 2012.

[7] A. Jain, D. Vera, and R. Harrison, "Virtual commissioning of modular automation systems," *IFAC Proceedings Volumes*, vol. 43, no. 4, pp. 72–77, 2010.

[8] S. Süß, S. Magnus, M. Thron, H. Zipper, U. Odefey, V. Fäßler, A. Strahilov, A. Kłodowski, T. Bär, and C. Diedrich, "Test methodology for virtual commissioning based on behaviour simulation of production systems," in *Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on*. IEEE, 2016, pp. 1–9.

[9] O. Givehchi, H. Trsek, and J. Jasperneite, "Cloud computing for industrial automation systems -— a comprehensive overview," in *2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2013, pp. 1–4.

[10] G. Frey and L. Litz, "Formal methods in PLC programming," in *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, vol. 4. IEEE, 2000, pp. 2431–2436.

[11] G. Canet, S. Couffin, J.-J. Lesage, A. Petit, and P. Schnoebelen, "Towards the automatic verification of PLC programs written in instruction list," in *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, vol. 4. IEEE, 2000, pp. 2449–2454.

[12] B. Riera, R. Benlorhfar, D. Annebicque, F. Gellot, and B. Vigario, "Robust control filter for manufacturing systems: application to PLC training," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 14 265–14 270, 2011.

[13] R. Benlorhfar, D. Annebicque, F. Gellot, and B. Riera, "Robust filtering of PLC program for automated systems of production," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 8705–8710, 2011.

[14] D. Soliman and G. Frey, "Verification and validation of safety applications based on PLCopen safety function blocks using timed automata in UPPAAL," *IFAC Proceedings Volumes*, vol. 42, no. 5, pp. 34–39, 2009.

[15] M. Perin and J.-M. Faure, "Building meaningful timed models of closed-loop DES for verification purposes," *Control Engineering Practice*, vol. 21, no. 11, pp. 1620–1639, 2013.

[16] V. Gourcuff, O. De Smet, and J.-M. Faure, "Efficient representation for formal verification of PLC programs," in *Discrete Event Systems, 2006 8th International Workshop on*. IEEE, 2006, pp. 182–187.

[17] J. Provost, J.-M. Roussel, and J.-M. Faure, "Translating Grafcet specifications into Mealy machines for conformance test purposes," *Control Engineering Practice*, vol. 19, no. 9, pp. 947–957, 2011.

[18] O. Ljungkrantz, K. Akesson, C. Yuan, and M. Fabian, "Towards industrial formal specification of programmable safety systems," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 6, pp. 1567–1574, 2012.

[19] L. Fransén, *National Electric Vehicle Sweden, Manufacturing Engineer, Private conversation*, 30 March 2016.

[20] S. Miremadi, K. Akesson, and B. Lennartson, "Extraction and representation of a supervisor using guards in extended finite automata," in *Discrete Event Systems, 2008. WODES 2008. 9th International Workshop on*. IEEE, 2008, pp. 193–199.

[21] K. Bengtsson, *Flexible design of operation behavior using modeling and visualization*. Chalmers University of Technology,, 2012.