Sequence Planner: Supporting Integrated Virtual Preparation and Commissioning

M. Dahl * **K. Bengtsson** * **P. Bergagård** * **M. Fabian** * **P. Falkman** *

* *Chalmers University of Technology, Department of Signals and Systems, 412 96 Göteborg, Sweden.*
{martin.dahl, kristofer.bengtsson, patrik.bergagard, fabian, petter.falkman}@chalmers.se

Abstract: It is essential to understand the operation sequences of a production system when designing or changing it. This paper will demonstrate how the software tool Sequence Planner (SP) not only supports this understanding by sequence visualization, but also improves the solution using optimization and verification. SP is a tool for modeling and analyzing automation systems. The tool has been developed since 2007 with an initial focus on supporting engineers when developing control code for programmable logical controllers. Today, SP is a micro-service architecture, usable in various areas like runtime control, online monitoring, energy optimization, and even emergency department patient planning. This paper presents a use case at an automotive company, where the operation sequences in a large number of automated robot stations, need to be modified. SP, together with virtual commissioning tools, automates this modification by identifying, optimizing, verifying and simulating operation sequences, and then updates the robot and control programs. This use case demonstrates the strength of SP and its architecture and how it is used for integrated virtual preparation and commissioning.

Keywords: Virtual manufacturing, automation, robot programming, sequence of operation

1. INTRODUCTION

Today's manufacturing industry is under constant pressure to deliver high quality products at lower cost and shorter time to market. Having robust, efficient, and flexible production systems is a requirement, but the development time of such systems must also be considered. One way to tackle this, is by having appropriate methodologies and *tools*. While virtual manufacturing tools today contain plenty of support for working with processes and running simulations, there is still a gap to be filled before these tools can support all parts of the design process.

Sequence Planner (SP) is a software developed at Chalmers University of Technology (Sequence Planner team, 2016). The main concept in SP is the formal modeling of operations and operation sequences (Lennartson et al., 2010). By using visualization (Bengtsson et al., 2012) as well as synthesis algorithms (Bergagård, 2015), the challenging task to model complex systems is simplified in SP (Bengtsson, 2012).

In this paper, the ideas and vision of SP are explored through a case study concerning visualization, verification, and optimization of robot program coordination in an automotive production setting. By integrating a virtual manufacturing tool with SP, a fast iterative work flow is enabled. A formal model of the system is generated and used for verification of robot interlocking and optimized coordination strategies are calculated. Validation of the optimized results are performed in the virtual manufacturing tool, where robot programs can be simulated in conjunction with other devices. In addition to plain visual inspection, these simulations provide collision detection and timing data.

The main contribution of this paper, is that it demonstrates how existing algorithms for formal verification (Ovatman et al., 2014), optimization (Shapiro, 1993) or supervisory control synthesis (Ramadge and Wonham, 1987) can be tightly integrated with currently used engineering tools to enable rapid design with short and fast iterations. SP is used as an enabling technology for these integrated iterations during an industrial use case. This integration is part of a framework called *Integrated Virtual Preparation and Commissioning* (IVPC) that was introduced by Dahl et al. (2016).

This paper is organized as follows: An introduction to IVPC is given in Section 2 and the ideas and concepts in SP are described in Section 3. In Section 4, the industrial case study is introduced. Sections 4.2 and 4.3 cover how to generate sequences from robot programs and how to verify them. In Section 4.4 the sequences are optimized, followed in Section 4.5 by a description on how simulation is performed.

⋆ 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).

2. VIRTUAL MANUFACTURING AND COMMISSIONING

Today's virtual manufacturing tools have the ability to upload and simulate robot programs from the shop floor. The tools also allow simulated devices to be controlled by a real control system. This setup is commonly referred to as *virtual commissioning* (VC) (Lee and Park, 2014).

The main motivation to use VC is to reduce testing and integration time during development (Hoffmann et al., 2010; Park and Chang, 2012; Lee and Park, 2014). This is achieved by being able to test and integrate the control system before the physical production system is completely installed. The hope is that using a simulation model of the production system, undesired behavior can be detected well ahead of physical installation. In fact, conducting VC enables tests that would be prohibitively expensive or even impossible to run on a physical system. In addition, having the simulation model makes it possible to test changes to the production system while the simulation is running and being able to incorporate last minute changes without worrying about their impact on the physical system.

Because VC uses the real control system, the simulation models need to be specified at the level of sensors and actuators (Lee and Park, 2014). This is now possible in simulation software from virtual manufacturing tool vendors, usually by allowing the user to define signals connected to the simulation and expose them to a control system via some interface (e.g. OPC, see Schwarz and Borcsok (2013)).

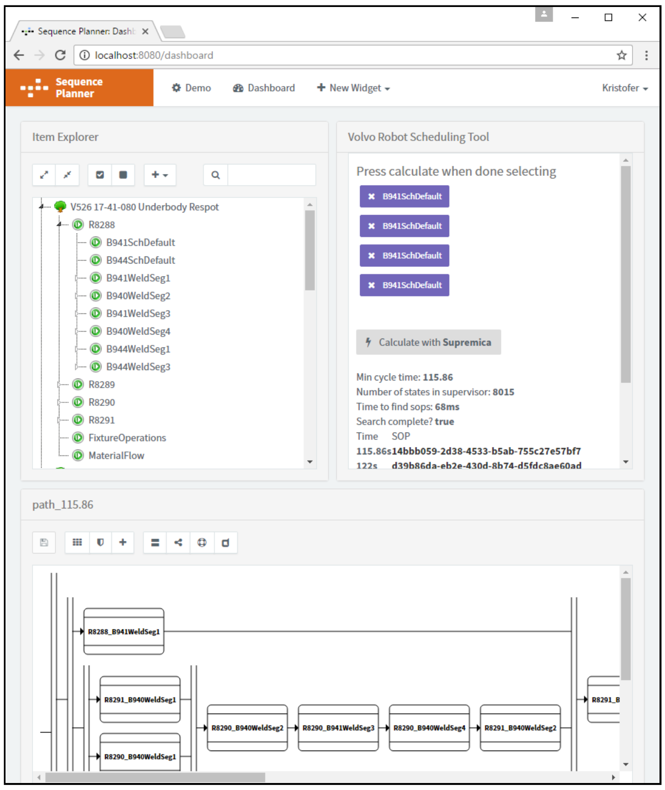
Oppelt and Urbas (2014) wrote that according to the The Association of German Engineers, current guidelines state that VC should be the last step in the automation engineering phase. But instead of conducting VC as the last step, Oppelt and Urbas (2014) suggest to extend VC to cover the entire automation engineering phase. This concept is called *Integrated Virtual Commissioning* since VC can enable continuous testing during the development; "The virtual plant is growing together with the automation software and thus enables simulation supported automation engineering" (Oppelt and Urbas, 2014).

This concept was extended by including a formal model of high level control logic in (Dahl et al., 2016), where a framework called *Integrated Virtual Preparation and Commissioning* (IVPC) was introduced. A formal model of control logic, combined with having the same simulation model of the production system shared between the preparation, control system implementation and VC phases (see the highlighted area in Fig. 1 in (Dahl et al., 2016)), enables both early validation of high level control and to perform early optimization based sequencing.

The methods described in this work are applied within this IVPC framework. However, the focus here is not on doing a full VC, but to use SP to optimize the robot sequences, and to simulate the optimized sequences in the virtual manufacturing tool. The reason for doing this is that the optimization may return several different robot sequences that are equally "good" according to the optimization criterion, and so an engineer has to be able to simulate these sequences to select the "best" one according to some criteria not expressed to the optimization engine.

3. SEQUENCE PLANNER

Sequence Planner (SP) is a tool for modeling and analyzing automation systems. Initially (Falkman et al., 2007), the focus was on supporting engineers in developing control code for programmable logical controllers (PLCs). During the first years, algorithms to handle product and automation system interaction (Bengtsson et al., 2009), and to visualize complex operation sequences using multiple projections (Bengtsson et al., 2012), was developed. Over the years, other use cases have been integrated, like formal verification and synthesis using *Supremica* (Bergagård and Fabian, 2012), restart support (Bergagård, 2015), cycle time optimization (Sundström et al., 2012), energy optimization and hybrid systems (Riazi et al., 2016), online monitoring and control (Theorin et al., 2016), as well as emergency department online planning support (Bengtsson et al., 2016).



Min cycle time: **115.86**
Number of states in supervisor: **8015**
Time to find sops: **68ms**
Search complete? **true**

```
Time    SOP
115.86s14bbb059-2d38-4533-b5ab-755c27e57bf7
122s    d39b86da-eb2e-430d-8b74-d5fdc8ae60ad
```

Fig. 1. Sequence Planner user interface.

SP is developed as a micro service architecture, where various services interact with each other by sending messages. One of these services is a web-server and a modern web-based user interface (see Fig. 1) where users can interact with the system. The user interface consists of so called *widgets*. A widget can for example show all items of a model, operation sequences, Gantt charts, settings for an optimization service or a control window for interacting with a running system. Fig. 1 shows to the left, a tree view showing the operations in the active model; to the right, an interface for solving the use case in this paper; at the bottom, a sequence of operations representing a possible coordination of the robot programs. In this paper, a number of widgets are presented that have been developed to support this specific case. Any widget can be added to the system while it is running, which makes it easy to add specialized widgets as needed.

The architecture is distributed by design and services are automatically identified when available and they can be started or stopped independently on separate computers. Any programming language can be used when implementing services since they communicate via a message bus. Also communication with external systems, like simulation software, the Robot Operating System (ROS), or PLCs, is done via the bus. This architecture also makes it easy to integrate various libraries like *Supremica*, Z3, IPOpt, AMPL, OscaR, etc.

This paper is a good showcase for the strengths of a modular architecture, since algorithms and methods were easy to reuse when implementing a specific use case. Currently, SP is used live at an emergency department in Sweden and will be used during virtual commissioning as well as for robot monitoring in car production. SP is open source and available on www.github.com (Sequence Planner team, 2016).

3.1 Modeling in Sequence Planner

An operation in SP is modeled as an Extended Finite Automaton (EFA) (Skoldstam et al., 2007), illustrated in Fig. 2. An operation has three locations, *init* (O_k^i), *executing* (O_k^e) and *finished* (O_k^f), as well as two events, *starting* ($O_k^\uparrow$) and *finishing* ($O_k^\downarrow$). The two events have *guards* ($C_k^\uparrow$ and $C_k^\downarrow$) that need to be satisfied for a transition to be possible. In this work, $C_k^\uparrow$ is commonly referred to as the *precondition* of an operation, and $C_k^\downarrow$ is the *postcondition*.

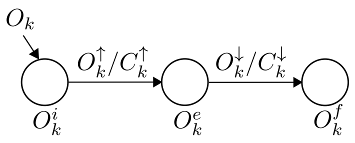


Fig. 2. Formal model of an operation O_k in SP.

SP uses a graphical language called Sequences of Operations (SOP) to represent operation sequences, showing the temporal relations among operations. An in-depth description of the modeling is given in (Lennartson et al., 2010; Bengtsson, 2012).

4. CASE STUDY

At the production facility that concerns this case study, robots (and other devices) operate on a first come, first served resource allocation principle. In each production cell, a PLC is in charge of allocating resources, such as mutually exclusive work zones. Robot programs communicate an intent to allocate a given resource, and the PLC grants access to that resource when it is available. In the current setup, the PLC has no additional logic to give priority to one device over another. This means that the coordinated sequence for all robots in a cell may differ between cycles. This may improve the flexibility of the system, but it can also make troubleshooting difficult and may occasionally lead to sub-optimal cycle times.

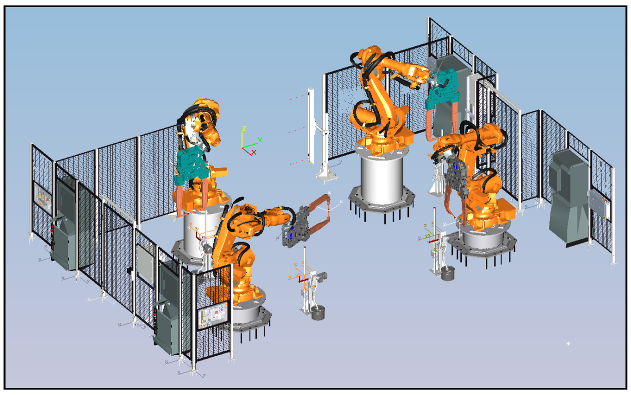


Fig. 3. View of a robot cell from a virtual manufacturing software tool. Figure credit: Volvo Car Corporation.

The company is currently planning to change from first come, first served order to a fixed order of execution. While a fixed order could be obtained by recording the current behavior, and manually re-programming the robots, this is complicated to do in practice due to the large variations in ordering and alternative product variants. Changing to a fixed order is also an opportunity to do more, especially in providing a verified and optimized ordering.

Due to the large number of robot stations to modify, the process needs to be fast and guaranteed to be correct. When changing over 1000 robots and PLCs in an entire plant, everything must work without introducing new problems. This use case is therefore a perfect situation where formal tools are needed as well as virtual manufacturing support.

4.1 Changing to a fixed ordering

One of the robot stations that will be changed to a fixed order is shown in Fig. 3. In contrast to when developing a new production system, robot programs and simulation models of the production cells already exist. To minimize redundant work, it is therefore essential to reuse this information. Given the robot programs and simulation models, the problem then becomes:

- Generate a model of the system from the robot programs that adheres to existing interlocking constraints.
- Use the generated model to verify that the interlocking constraints are as intended.
- Using optimization, generate a number of suitable sequences from the created model.
- Validate sequencing choices by simulation inside the virtual manufacturing tool.
- Manually add and/or change specifications until satisfied with the simulation results.

4.2 Robot programs to Extended Finite Automata

Initially, it is important to understand the current behavior of the production cell. Visualizing sequences from different perspectives, for example one sequence per robot, or the sequence w.r.t. one product, gives insight into the operation of the system (Bengtsson, 2012).

The production cells studied in this work, contain a number of robots, as in Fig. 3. For each product variant produced by the system, there exists a main sequence defining what operations to perform and in what order. This sequence is contained in the robot program that communicates with the PLC to allocate and release resources. An example of this can be seen in Listing 1.

The robot programs can automatically be imported into a virtual manufacturing tool (in this case, Siemens Process Simulate [1]). When imported into the virtual manufacturing tool, robot programs are automatically split into a tree of subprograms, which are instances of different types of operations. From this, SP identifies each sub-operation, their preconditions, and generates an EFA model of the system, to be able to visualize, verify, and optimize the behaviour.

```
PROC B941SchDefault()
   Reset O_Homepos;
   WaitSignal AllocateStation;
   WaitSignal AllocateZone1;
   !R01 Zone1 to R02;
   B941WeldSeg1;
   WaitSignal ReleaseZone1;
   WaitSignal AllocateZone3;
   !R01 Zone3 to R03;
   B940WeldSeg2;
   B941WeldSeg3;
   B940WeldSeg4;
   WaitSignal ReleaseZone3;
   WaitSignal ReleaseStation;
   WaitSignal WorkIsDone;
   AutoDress Gun311;
ENDPROC
```

Listing 1. Robot program example illustrating how the robot performs welding actions between zone allocation commands. Lines beginning with '!' are comments.

Each robot is modeled as set of operations with preconditions. The sequence contained in each main robot schedule is naturally expressed as the preceding (if any) operation being finished. This can be expressed simply as $C_k^{\uparrow} = O_i^f$, meaning that the preceding operation O_i must be in its finished location for the precondition of O_k to be satisfied.

In addition to this, the physical positions of the robot are also obtained from the virtual manufacturing tool. This enables a more flexible model where the operations can only start when the robot is in the correct position. Some of these positions are static and some are paths in space, corresponding to an entire robot operation. A robot's state is then encoded as being in one of these positions.

These positions are imported from the virtual manufacturing tool into SP, where they are encoded as a variable r^n_{state}, where $n \in$ (1..number of robots) for each robot. For example, for the robot containing the program in Listing 1: $r^n_{\text{state}} \in$ (*HOME*, *B941WeldSeg1*, *B941WeldSeg1_end*, ...). In this case, *B941WeldSeg1_end* and *HOME* represent points in space while *B941WeldSeg1* represents the robot being somewhere on this particular path, performing welding of segment 1.

Preconditions are added to each operation in SP to ensure that they can only start when its robot is in the correct position (e.g. $r^1_{\text{state}} = HOME$ meaning that the precondition for operation k is satisfied when robot one is in its home position).

In this stage, it may be necessary to add some details manually to the model, details that cannot be automatically extracted from the robot programs. For instance, it can be the case that the robot program waits for a certain signal given by the PLC. If this signal is triggered by something not found in any of the robot programs (e.g. waiting for an external device), this needs to be manually specified, since the PLC-programs are not part of the input data. This could for instance be expressed as additional sequencing constraints as described above.

4.3 Modeling by synthesis

Not only is it convenient to obtain a correct model of the zone allocation as the solution to a synthesis problem (see Bergagård et al. (2015) for details), it is also faster and less error-prone than doing it manually. Calculation of this is based on the Supervisory Control Theory (SCT) (Ramadge and Wonham, 1987). The SCT is a model-based framework for control of discrete event systems. Very briefly, the idea is to be able to synthesize a *supervisor* that disables (a minimum of) transitions leading to blocking or forbidden states.

As occupying a specific zone should be mutually exclusive between robots, all unique pairs of operations that share a certain zone are modeled as forbidden states of operations that are in their executing location (e.g. $O_i^e \wedge O_k^e$). These forbidden state expressions are then used to formulate a supervisor synthesis problem. Using the method described in (Miremadi et al., 2012), the solution to this synthesis problem can be obtained as additional conditions on the preconditions of the operations. The synthesis is performed by querying the software *Supremica* (Åkesson et al., 2006).

When performing the synthesis, *Supremica* calculates a supervisor that contains all valid states. This supervisor can now be used for verification. If there exists an additional specification containing a list of the operations belonging to each zone, the supervisor can be used to check whether a (partial) state where a pair of such operations are both active exists; this indicates that there is a problem in the robot programs. Of course, such a specification could also be taken into account directly into the synthesis problem formulation, which would guarantee that only correct zone allocations will be performed. But that would not reflect the system *currently* running.

With the additional preconditions on the operations in place, the model can now be visualized from different perspectives. Using the algorithms described by Bengtsson et al. (2012), a map of relations between the operations in the system can be calculated. In this case, three different types of sequences can be visualized by SP: sequences for the complete cycle, sequences for each robot, as well as sequences for each shared zone.

[1] Siemens PLM, http://www.plm.automation.siemens.com/, 2016

4.4 Sequence optimization

Returning to the goal of converting a robot cell from a first come, first served zone allocation principle to a fixed order of execution, it is desirable to find the fixed ordering that results in the shortest total time to complete the robot programs. The time each operation takes can be obtained either from simulation in the virtual manufacturing tool, or by measurements on the shop floor.

Rather than to use a graph based search method on the supervised system (which typically contains millions of states), another way to find the shortest total time (λ in (1)) is to construct a scheduling problem and solve using an optimization engine.

$$\begin{array}{lll}
\underset{s_i,\ \forall i \in 1..N}{\text{minimize}} & \lambda & \\
\text{subject to} & e_i = s_i + d_i, & \forall i \in 1..N \\
& e_i < \lambda, & \forall i \in 1..N \\
& e_{p_i} \leq s_{p_j}, & \forall \langle p_i, p_j \rangle \in P \\
& e_{m_i} \leq s_{m_j} \vee e_{m_j} \leq s_{m_i}, & \forall \langle m_i, m_j \rangle \in M \\
& e_{f_i} = s_{f_j}, & \forall \langle f_i, f_j \rangle \in F \\
& s_i = 0 \vee \exists e_j = s_i, & \forall i, j \in 1..N, i \neq j \\
& s_i \geq 0, & \forall i \in 1..N
\end{array} \tag{1}$$

In (1) the start times ($s_i, \forall i \in 1..N$) of N operations are the decision variables. To simplify expressing the constraints, the end times ($e_i = s_i + d_i, \forall i \in 1..N$) of the operations are also included, where d_i refers to the duration of each operation. Constraints relating to the sequence of each robot main schedule ($C_k^{\uparrow} = O_i^f$) are expressed using pairs of indexes for the start and end times in P. Similarly, mutual exclusion w.r.t. zone allocation ($O_i^e \wedge O_k^e$) are expressed using the pairs in M. The next constraint forces operations that are performed in sequence within the same zone to start right after each other, to avoid having to introduce new zone allocation calls, with pairs of indexes in F. Finally, the last constraint in (1) describes that operations can only start executing when another operation finishes (except at time $t = 0$), to model the event based nature of the system. When solving, some additional constraints are added to guide the search; these are not included in (1).

For the robot station depicted in Fig. 3, this (rather small) problem is solved using the Constraint Programming solver library OscaR (OscaR Team, 2012) in around one second on a standard laptop computer (Intel Core i5, 8gb ram). One result of the optimization can be seen in Fig. 4, and the same result but visualized as a SOP instead can be partly seen at the bottom of Fig. 1.

When solving, all solutions with different makespans are saved, to be able to give an overview of how fast or slow the system can perform depending on which operations get to execute first. Any of these solutions can be used to drive a simulation of the system, so that an engineer can decide on the most suitable sequence.

4.5 Validation by simulation

With the addition of VC support in virtual manufacturing tools, it is now possible to simulate the system based on

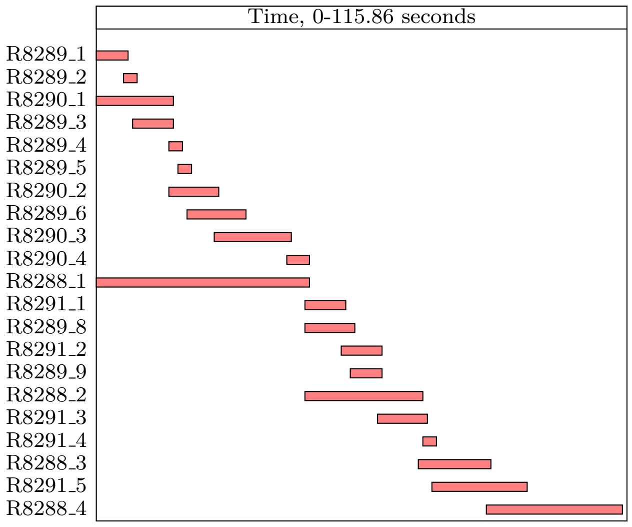


Fig. 4. Optimization result with four robots (the numbers correspond to the different operations for each robot).

the control logic contained in the preconditions of our synthesized system, as was suggested in (Dahl et al., 2016).

By adding the logic required to restrict robot program execution to satisfy (one of) the SOP(s) calculated in Section 4.4 (that provides one of the best possible fixed order of execution), it is possible to simulate in real-time the optimized coordination in the virtual manufacturing tool.

Signals to start operations in the virtual manufacturing tool are available to SP. By setting up an OPC server that exposes these signals to the virtual manufacturing tool and SP, SP can control the start of these operations, and also be notified when they complete. In addition, the virtual manufacturing tool communicates information regarding device state and timing directly to SP over the message bus.

By tracking timing data from the virtual manufacturing tool, cycle times per robot can be visualized in SP. A live Gantt view tracks which operations each robot executes. This has previously been done in SP by tracking the program pointer of the robots in a physical station in (Nord and Wahlqvist, 2016), and the setup here is more or less the same, except that the station is now simulated.

5. CONCLUSIONS

The aim of this paper was to demonstrate how Sequence Planner can be used when working with production preparation within Integrated Virtual Preparation and Commissioning. By going from uploaded robot programs to a live simulation of robots with optimized coordination with minimal manual intervention, we have shown how existing optimization and synthesis techniques can be applied within SP to solve a concrete industrial problem.

Since the model is generated from the existing programs, any problems in them would not be found unless there exists some external source of zone allocation specification. Future work include generating zone specifications from intersection of sweep volumes in the virtual manufacturing tool as was done in (Shoaei et al., 2010).

ACKNOWLEDGEMENTS

The authors thank the industrial partners of the VIRTCOM project for the inspiring discussions during meetings, and Volvo Car Corporation in particular for sharing the simulation models and robot programs used in this work.

REFERENCES

Åkesson, K., Fabian, M., Flordal, H., and Malik, R. (2006). Supremica-an integrated environment for verification, synthesis and simulation of discrete event systems. In *Discret. Event Syst. 2006 8th Int. Work.*, 384–385. IEEE.

Bengtsson, K., Bergagard, P., Thorstensson, C., Lennartson, B., Akesson, K., Yuan, C., Miremadi, S., and Falkman, P. (2012). Sequence planning using multiple and coordinated sequences of operations. *IEEE Transactions on Automation Science and Engineering*, 9(2), 308–319.

Bengtsson, K. (2012). *Flexible design of operation behavior using modeling and visualization.* PhD Thesis at Chalmers University of Technology. 231.

Bengtsson, K., Blomgren, E., Henriksson, O., Johansson, L., Lindelöf, E., Pettersson, M., and Söderlund, Å. (2016). Emergency department overview - improving the dynamic capabilities using an event-driven information architecture. In *IEEE International Conference on Emerging technologies and factory automation (ETFA) 2016.*

Bengtsson, K., Lennartson, B., and Yuan, C. (2009). The origin of operations: Interactions between the product and the manufacturing automation control system. *IFAC Proceedings Volumes*, 42(4), 40 – 45. 13th IFAC Symposium on Information Control Problems in Manufacturing.

Bergagård, P., Falkman, P., and Fabian, M. (2015). Modeling and automatic calculation of restart states for an industrial windscreen mounting station. *IFAC-PapersOnLine*, 48(3), 1030–1036.

Bergagård, P. (2015). *On restart of automated manufacturing systems.* PhD Thesis at Chalmers University of Technology.

Bergagård, P. and Fabian, M. (2012). Deadlock avoidance for multi-product manufacturing systems modeled as sequences of operations. In *2012 IEEE International Conference on Automation Science and Engineering: Green Automation Toward a Sustainable Society, CASE 2012, Seoul, 20-24 August 2012*, 515 – 520.

Dahl, M., Bengtsson, K., Bergagård, P., Fabian, M., and Falkman, P. (2016). Integrated virtual preparation and commissioning: supporting formal methods during automation systems development. *IFAC-PapersOnLine*, 49(12), 1939–1944.

Falkman, P., Lennartson, B., and Andersson, K. (2007). Specification of production systems using PPN and sequential operation charts. In *2007 IEEE International Conference on Automation Science and Engineering*, 20–25.

Hoffmann, P., Maksoud, T., Schumann, R., and Premier, G. (2010). Virtual Commissioning of Manufacturing Systems a Review and New Approaches for Simplification. *Proc. 24th Eur. Conf. Model. Simul.*, 2(Cd).

Lee, C.G. and Park, S.C. (2014). Survey on the virtual commissioning of manufacturing systems. *J. Comput. Des. Eng.*, 1(3), 213–222.

Lennartson, B., Bengtsson, K., Yuan, C.Y., Andersson, K., Fabian, M., Falkman, P., and Åkesson, K. (2010). Sequence planning for integrated product, process and automation design. *IEEE Transactions on Automation Science and Engineering*, 7, 791–802.

Miremadi, S., Lennartson, B., and Åkesson, K. (2012). A BDD-based approach for modeling plant and supervisor by extended finite automata. *Control Syst. Technol. IEEE Trans.*, 20(6), 1421–1435.

Nord, D. and Wahlqvist, H. (2016). *The tweeting robot - Collection and processing of data from industrial robots.* Master's thesis, Chalmers tekniska högskola.

Oppelt, M. and Urbas, L. (2014). Integrated Virtual Commissioning an essential Activity in the Automation Engineering Process From virtual commissioning to simulation supported engineering. In *Ind. Electron. Soc. IECON 2014 - 40th Annu. Conf. IEEE*, 2564 – 2570.

OscaR Team (2012). OscaR: Scala in OR. Available from `https://bitbucket.org/oscarlib/oscar`.

Ovatman, T., Aral, A., Polat, D., and Ünver, A.O. (2014). An overview of model checking practices on verification of PLC software. *Softw. Syst. Model.*, 1–24.

Park, S.C. and Chang, M. (2012). Hardware-in-the-loop simulation for a production system. *Int. J. Prod. Res.*, 50(8), 2321–2330.

Ramadge, P.J. and Wonham, W.M. (1987). Supervisory control of a class of discrete event processes. *SIAM J. Control Optim.*, 25(1), 206–230.

Riazi, S., Bengtsson, K., Bischoff, R., Aurnhammer, A., Wigström, O., and Lennartson, B. (2016). Energy and peak-power optimization of existing time-optimal robot trajectories. In *Automation Science and Engineering (CASE), 2016 IEEE International Conference on.*

Schwarz, M.H. and Borcsok, J. (2013). A survey on OPC and OPC-UA: About the standard, developments and investigations. In *Information, Commun. Autom. Technol. (ICAT), 2013 XXIV Int. Symp.*, 1–6. IEEE.

Sequence Planner team (2016). Sequence Planner. Available from `https://www.github.com/kristoferB/SP`.

Shapiro, J.F. (1993). Mathematical programming models and methods for production planning and scheduling. *Handbooks in operations research and management science*, 4, 371–443.

Shoaei, M.R., Lennartson, B., and Miremadi, S. (2010). Automatic generation of controllers for collision-free flexible manufacturing systems. In *Autom. Sci. Eng. (CASE), 2010 IEEE Conf.*, 368–373. IEEE.

Skoldstam, M., Akesson, K., and Fabian, M. (2007). Modeling of discrete event systems using finite automata with variables. In *Decision and Control, 2007 46th IEEE Conference on*, 3387–3392. IEEE.

Sundström, N., Wigström, O., Falkman, P., and Lennartson, B. (2012). Optimization of operation sequences using constraint programming. *IFAC Proceedings Volumes*, 45(6), 1580 – 1585.

Theorin, A., Bengtsson, K., Provost, J., Lieder, M., Johnsson, C., Lundholm, T., and Lennartson, B. (2016). An event-driven manufacturing information system architecture for industry 4.0. *International Journal of Production Research*, 1–15.