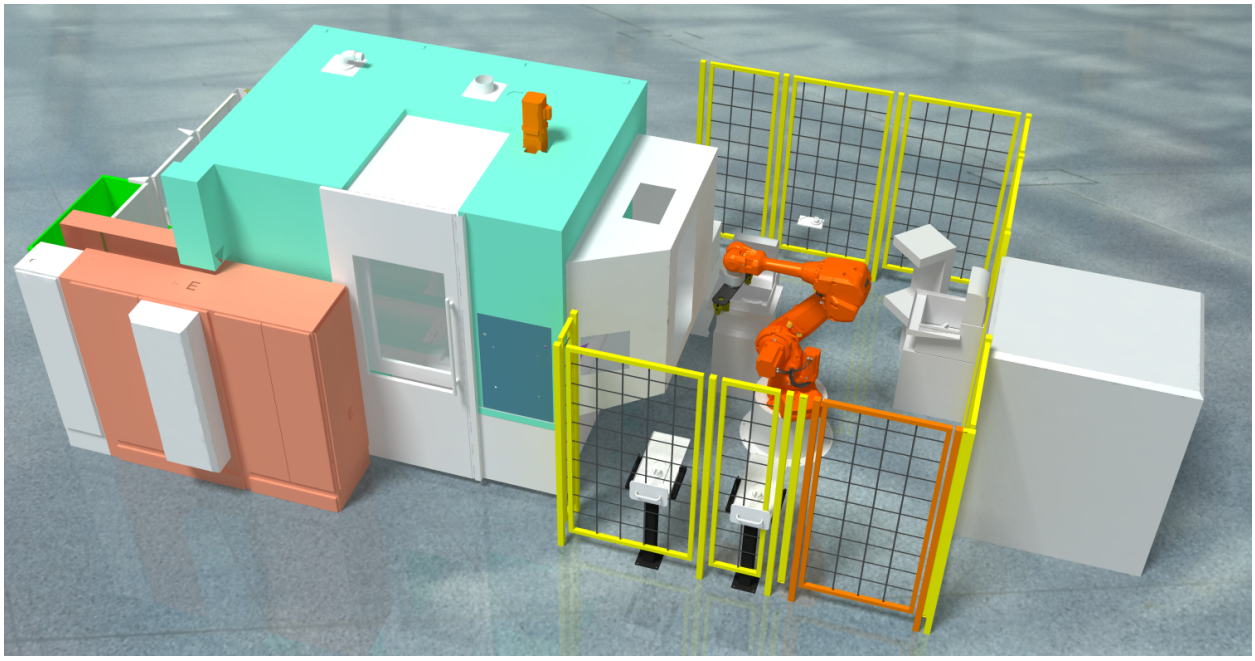
CHALMERS
UNIVERSITY OF TECHNOLOGY



Testing and Evaluation of Virtual Commissioning

Case study of an existing robot cell at Scania modelled with 3DExperience

Master's thesis in Systems, Control and Mechatronics

OSCAR JOHANSSON

Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Master's thesis 2017:NN

Testing and Evaluation of Virtual Commissioning

Case study of an existing robot cell at Scania modelled with 3DExperience

OSCAR JOHANSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Signals and Systems
Division of Division name
Name of research group (if applicable)
Chalmers University of Technology
Gothenburg, Sweden 2017

Testing and Evaluation Virtual Commissioning
Case study of an existing robot cell at Scania modelled with 3DExperience
Oscar Johansson

Supervisor: Andreas Rosén, Scania
Examiner: Assoc. Prof. Petter Falkman, Department of Signals and Systems

Master's Thesis 2017:NN
Department of Signals and Systems
Division of ~~Division name~~
~~Name of research group (if applicable)~~
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Picture showing a 3D model of the invstigated cell, created in 3DExperience.

Typeset in LaTeX
Printed by [Name of printing company]
Gothenburg, Sweden 2017

Testing and Evaluation of Virtual Commissioning
Case study of an existing robot cell at Scania modelled with 3DExperience
OSCAR JOHANSSON
Department of Signals and Systems
Chalmers University of Technology

Abstract

A big challenge in industry is to efficiently validate industrial control code. There are a number of ways to do this and this project investigates the feasibility of using Dassault Systèmes 3DExperience to carry out a Virtual Commissioning. This is done by creating a virtual replica of an existing manufacturing cell, producing cog wheels, at Scania. A 3D model is created in 3DExperience and then connected via the OPC protocol to be controlled by a simulated Siemens PLC.

It is concluded that it is possible to carry out such a project with this software tool but some modifications to the programs in the cell was needed. The existing robot program were too advanced to be handled by the Delmia translator and could not be imported. The PLC program was functional with the addition of an open source add-on making it possible to communicate via OPC and an extra code block to remap the signals to the correct address.

In order to simulate the robot movements, the targets and speed/acceleration paramters were imported from the robot program. All movements of the robot had to be re-programmed in Delmia.

Suggestions for how to construct robot- and PLC programs in order to work with this software is presented as well as suggestions for improvements in the software.

Keywords: Virtual Commissioning, Emulation, 3DExperience, PLC, Simulation, Automation, Scania.

Nomenclature

2D	Two Dimensional
3D	Three Dimensional
CAD	Computer-Aided Design
DB	Data Block
FSA	Finite State Automata
HIL	Hardware In the Loop
HMI	Human-Machine-Interface
M Bit	Memory bit
OLE	Object Linking and Embedding
OPC	OLE for Process Control
PLC	Programmable Logic Controllers
PLM	Product Lifecycle Management
PROFIBUS	Process Field Bus
PROFINET	Process Field Net
R/W	Read/Write
RFID	Radio-Frequency Identification
RIL	Reality In the Loop
SIL	Software In the Loop
SPC	Statistical Process Control
TCP	Tool Center Point
TIA Portal	Totally Integrated Automation Portal

Acknowledgements

First I would like to thank Scania and Andreas Rosén, who's been my supervisor at Scania, for the time and interest he has taken into my project. Also a big thanks goes out to Franz Waker and everyone else in the "Digital Factory" department for taking the time to help me out when needed. I would also like to thank Robert Bergkvist for helping me out with PLC programming and Stefan You for the information about the production line. Simon Gräsberg and Robin Halldin from Sejfo group has helped me a lot with the robot programs and taken their time to thoroughly explain the function of the robot.

I also extend my gratitude towards Dassault Systèmes for supplying the needed software, especially towards André Jönsson, Bernhard Loske and Ioannis Basoukos for helping me out with various problems regarding the software.

From Chalmers I would like to thank Kristofer Bengtsson and finally I would like to thank my academical supervisor and examiner at Chalmers, Petter Falkman, for the ideas and support he's given.

Oscar Johansson, Gothenburg, April 2017

Contents

List of Figures

List of Tables

1

Introduction

1.1 Background

Scania is a global swedish truck and bus manufacturer owned by Volkswagen AG since 2014, however most of the development and manufacturing is still done in the original site in Södertälje. Today Scania is not performing commissioning of new production systems in-house, instead they hire subcontractors.

The automotive sector is a very competitive industry where the product variety is continuously increasing while the time for entering the market is decreasing. In order to stay ahead of the competition the set-up of new production systems has to be quick [1] and Scania is investigating the possibility of doing this themselves in the future.

Automated production systems today are usually controlled by PLCs that often run very advanced and complex code. When installing a new production line/-cell or robot, a big challenge in industry is to validate and verify this code efficiently. In the past this has often been done by manual debugging or by trials on the real equipment. The first method is very time consuming and the latter will often cause production stops due to long set-up times and in worst case damages to equipment and/or personnel.

To avoid these problems, simulation and emulation methods are becoming increasingly more popular to verify code before the actual physical equipment is installed. By the use of virtual commissioning, a lot of time and resources can be saved. The problem is to guarantee that the virtual model is an accurate replica of the real system.

1.2 Objective

The objective of the thesis work is to implement a virtual replica of an existing production cell at Scania producing cog wheels for use in gearboxes. The model of the cell will be implemented in Dassault Systèmes 3D Experience platform to verify that the software tool and the collectable data & information is enough to make a

virtual commissioning of the production system. The CAD model should then be controlled by a PLC that to largest possible extent is running the same code as the one controlling the actual plant. The virtual model should then mimic the actual systems behaviour.

1.3 Description of the Production Line

The Line containing the cell to be investigated produces cog wheels and is part of one of Scanias factories in Södertälje, dedicated to the manufacturing of gearboxes. The factory have several lines that are very similar with the difference that they can produce different parts. Each line is supervised by one operator that does both loading, unloading and inspection of parts. The input material to the production line is soft casted metal and the output is completely machined cog wheels ready for hardening.

The raw material for the cog wheels come casted with roughly the correct shape and is loaded onto a conveyor by the line operator. They first enter a lathe that machines the parts in two steps, starting with machining the center hole and one base side. The next step is finishing the lateral face and the last base side. After the lathe the part is picked up by a pick-and-place robot moving it to fixtures on another conveyor. These fixtures then enters the robot cell and are later used to transport the completed cog wheels to a quality inspection station at the end of the conveyor for packaging.

The production line is able to produce a variety of different cog wheels. This can however not be done without set-up time in the lathes that takes several hours.

1.3.1 Robot cell

The cell in question consists of one ABB IRB 4600 Robot loading and unloading five machines, two SPC hatches and a conveyor with material for cog wheels. Below follows a description of the material flow, a visualization can also be seen in Figure 1.1. The numbers in Figure 1.1 corresponds to the following:

1. Conveyor (Pick-up station)
2. Machine one - Hobbing machine
3. Machine two - Deburring machine
4. Centrifuge one
5. Centrifuge two
6. SPC hatch one
7. SPC hatch two
8. Marking machine

When the material enters on the fixtures they stop at a pick-up station for the robot and waits to be picked up. A tool with two grippers is attached to the robot enabling

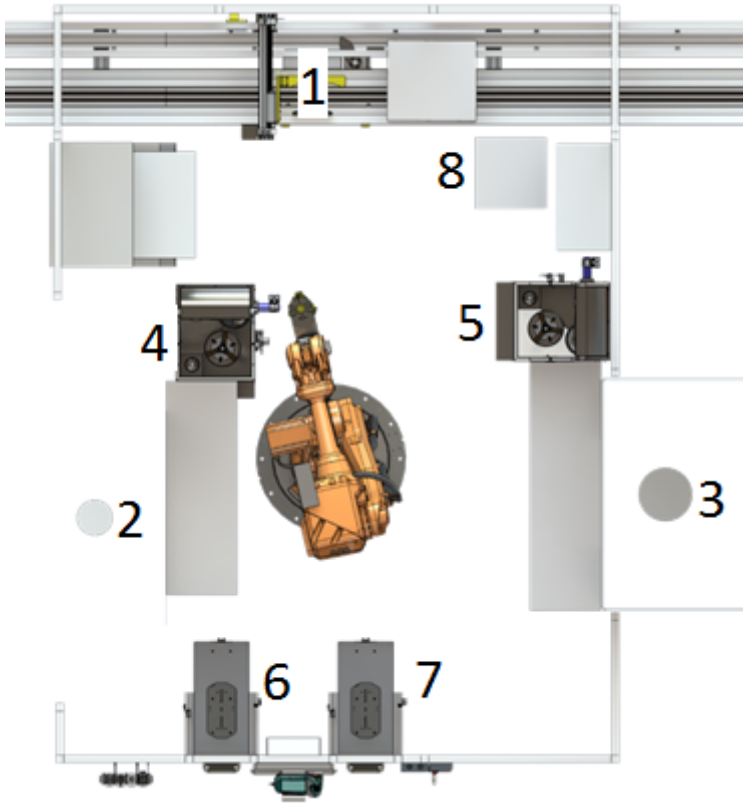


Figure 1.1: Cell layout. Picture: Sejfo Group

it to carry two parts at the same time, so it continuously exchanges parts. E.g when it picks up an incoming part it leaves the previously completed part to exit on the fixture that just came in.

The incoming parts is placed in the gear hobbing machine to have the cogs made, it is then moved to the first centrifuge in order to get rid of chips and oil residues. From the centrifuge it is moved to a machine for deburring of the cog edges. After this the processed part is once again centrifuged to remove any remaining chips and oil before it is moved to the marking machine for engraving of brand name and production time. The engraving is the final step before the part is moved to an empty fixture and exits on the conveyor.

Inspections of parts are done at a predetermined frequency decided for each product. This can be done in any step of the process and then the robot places the part at the corresponding SPC hatch. There are two hatches, numbered one and two. To avoid mistakes, parts coming to hatch number one has been processed by machine one (the hobbing machine) and parts arriving at the second hatch has been processed by machine two (the deburring machine).

In the case of an inspection, a light is turned on as an indication to the operator that a measurement of the part is required. If it is within tolerance the part is returned to the SPC hatch and a request to the robot is sent from the HMI to return the

product to the material flow. This operation has priority over the part coming from the previous operation as not to block the system when next inspection piece come. The operator can also at any time request a part for inspection via the HMI (from any step in the work flow), the robot will then pick the next part coming from the requested location in the flow and put it at the corresponding SPC hatch.

The system is capable of handling 18 different articles but the work flow will be the same no matter which type of cog wheel is currently being produced. The robot and other machines does not require any set-up time when switching between articles like the lathes and milling machines do, instead these read an RFID tag in the fixture delivering the part and switches to another program since the tools are the same.

1.4 Delimitations

The thesis has only covered a part of the line in question, the robot cell. To incorporate the whole production line would have been a too complex task for the scope of this project.

Only the PLC code running the cell has been included in the simulations, i.e the PLC controlling the conveyor and interfacing between the robot and the other machines. The machines have their own PLCs but these have been considered as black boxes in the simulation model.

The robot program showed to be far to extensive for Delmia to handle so only the robot positions and its speed & acceleration parameters have been imported from the real program. The robot program and movements was then reproduced in a more simple way to work properly in Delmia.

The robot program and PLC program uses the article numbers of the products to index a database with parameters, this is excluded in the model since this is not supported in Delmia. Due to the difficulties with storing different parameters only one type of product is considered.

2

Theory

In this chapter previous work, different concepts and required information regarding virtual commissioning is presented.

2.1 PLC

A PLC, or Programmable Logic Controller, is an industrial computer designed to operate in harsh environments for the control of manufacturing processes. It is usually rack mounted in a cabinet and constructed in such a way that is is easy to program, debug and exhibits a reliable behavior.

The basic function of a PLC is that it runs its program over and over in a cyclic manner. This cyclic program is done in three steps;

1. Read inputs and temporarily store a snapshot of them
2. Calculate all logic using this snapshot
3. Set outputs accordingly

By storing a snapshot of the inputs it guarantees that the calculations won't be unpredictable due to a change of inputs during the calculation. This is then repeated each sample interval.

PLCs can communicate and interface with other factory equipment in a number of ways. E.g a switch can be directly connected to an input pin and a HMI panel can be connected using ethernet or some vendor specific protocol.

The investigated cell is equiped with Siemens PLCs and make use of DP/DP couplers to communicate between the different machines (Networks). These work as an interface between two Profibus networks (Standard for fieldbus communication in industry) to enable communication and data transfer between their respective master. This means most of the communication and data storage between networks is done via data blocks.

2.2 Virtual Commissioning

Virtual commissioning is the concept of using a virtual/digital replica/copy of a mechanical system to reduce the time needed in the real commissioning phase. Commissioning is defined by Glas as *"all activities aiming at putting a completely assembled and mechanically reviewed production system into operation"* [2]. Furthermore Reinhart and Wünsch state that the commissioning phase *"ends with the production of the first work pieces that meet the specifications and the acceptance by the customer"* [2]. The aim with building a virtual model of the mechanical system (emulating the production system) is to test and verify production systems and its control system before the existence of the actual system [3]. This greatly helps shorten the time to fulfil Glas definition of commissioning by exposing problems in an early construction phase.

According to Makris et al. virtual commissioning can be included in the digital factory regime (the use of virtual reality and simulation to design and optimize processes [4]) and also says it is closely related to simulation but also includes the mechatronic behaviour in the model [5]. It is a natural inclusion in the digital factory since the computer models used in the design phase can be used to verify code and functionality and later even be used for improvement simulations.

2.3 Simulation vs Emulation

Shannon defines in [6] simulation as *"the process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behavior of the system and /or evaluating various strategies for the operation of the system"*. When using simulations in a production context it mostly refers to a discrete event simulation since these models have the capability to mimic the dynamics of the system. Ingalls uses this and in [7] adapts the definition of simulation as *"the process of designing a dynamic model of an actual dynamic system for the purpose either of understanding the behavior of the system or evaluating various strategies for the operation of the system"*.

When including dynamics, the model can suddenly be used for emulation and Zhang et al. defines emulation as *"the testing of control systems through the use of a simulation model"* [8]. Danielsson et al. extends this by saying that an emulator has to be connected to a simulation including all geometrical and physical properties of the contained objects [9].

Simulation can be used to quickly investigate throughput over time and detect deadlocks since the simulation time can be sped up. This however should not be done when running an emulation since the embedded control system for the plant is designed to run in real time. By speeding up this process unexpected behaviour can arise.

Thus the main differentiating factor is that emulation uses a simulation model to run embedded control systems to control a virtual version of the plant. Emulation should be done in real time and used for validation rather than testing different improvements in the production system.

2.4 Methods for validating industrial control logic

In [10] Auinger et al. presents four different ways of validating the commissioning of a production system:

1. Testing on real system - the tradition way
2. Soft commissioning / HIL - Real control system and a virtual plant
3. RIL - Real plant and simulated control system
4. Off-line simulation / SIL - Simulated control system and simulated plant

In the traditional way testing is done on the actual physical equipment, something that may be problematic since any errors will cause delays and extra costs. This is usually done by connecting the controller to a stand-alone version /replica of the production cell [11]. This adds cost since a cell has to be constructed just for testing purposes and it also neglects the rest of the factory equipment that most likely is connected to it [11]. Often this is not possible when testing conveyor systems since these usually are too large to be made off site and testing has to be done in the actual production facility.

Soft commissioning /HIL is what is also often referred to as emulation or virtual commissioning and can be done prior the existence of the physical plant. By doing this the software engineers can work in parallell with the mechanical engineers in the commissioning of a production system. It also enables better reproduction of the actual conditions within the plant [11]. Reinhart and Wünsch on the other hand says virtual commissioning can be both HIL and SIL, just two different approaches [2].

In the RIL approach the real production system is used which may result in dangerous situations and add extra cost as well as pose difficulties when the virtual control system are to be transformed to a real one [12].

In the SIL approach everything is done virtually which gives low cost for experiments but since the real system does not exist the functionality is not proven [12]. A lot of effort may be needed to go from simulation to reality.

2.5 Benefits of using Virtual Commissioning

With the correct use, virtual commissioning can reduce start-up times by up to 50 % [13] and help make simulation models useful in several steps of the project as well as getting products out on the market faster [14]. It can also reduce risk and cost in the start-up phase of a production line [13]. Virtual commissioning creates an environment suitable for code optimization since it does not add extra down time on the real system at the same time as it gives environmental benefits in the form of fewer wasted products. And with the possibility to reduce start up times, less travels are needed for personnel [13]. Potentially dangerous situations can be shown and evaluated and the staff can be trained in advance with no need for excessive downtime on the real system [15].

2.6 Difficulties with Virtual Commissioning

Oppelt et al. identifies that the major reasons virtual commissioning isn't used to greater extent are the major modelling effort and the need for plenty of knowledge in creating the virtual models [16]. Berger et al. also states that there may be difficulties when swithing to the real production system from the virtual one [12].

According to Lee and Park the two major hurdles with the implementation of virtual commissioning in industry is the lack of physical device modeling and logical device modeling [17]. These two are not normally included in the traditional development of production systems.

They elaborate by saying that the physical modeling consist of two parts:

- A geometric model
- A kinetic model

The geometric model is included in a lot of litterature but the kinetic modeling is mostly done by hand and not given much attention in litterature, making it a very time consuming task [17].

They point out the biggest problem with modeling the logical part is the need for in-depth knowledge of modelling and simulation [17]. This includes knowledge about Automata, Petri nets and set theory.

2.7 Previous work /case studies

Park et al. studied ten different PLC applications in industry and verified them using a five-step method [18]:

1. Create 3D graphic model
2. Create the virtual model (including dynamics)
3. Map inputs and outputs between the virtual world and the PLC
4. Run the virtual production system and the PLC simultaneously
5. Check the behaviour of the virtual system compared to the PLC

The first step involves drawing the production system using some CAD software. The second step is to model the system including states, dynamics and time. In this case the authors used FSA to model the system with states, transistions and events. When the number of states grow this approach will be quite difficult. The third step includes connecting the outputs of the PLC to corresponding operation in the virtual model and the inputs of the PLC to some sensor in the virtual model. The fourth and fifth step is running the model and check that the behaviour is the intended one.

Park et al. use a method where they split the virtual model into a generic model and a visual one. The generic model is what is describing the dynamics and is modeled with FSA. The visual one is drawn in some CAD software. This method makes it possible to use different 3D simulation softwares using the same FSA model. The authors found that PLC programs could easily be validated using their approach with the softwares ROBOWORKS and IGRIP. They also mention DELMIA Automation V5 as an alternative but says there have been difficulties in creating the virtual models as well as in changing them afterwards [18].

Makris et al. applied virtual commissioning on a cell with cooperating robots in [5]. In their experiments however, they used a SIL approach with two PCs where one ran the visual model and the second simulated the behaviour of the model. They used a similar methodology to Park et al. but with four steps:

1. Develop the simulation model (3D graphic model)
2. Define the inputs and outputs from the PLC
3. Define the material flow and connect the inputs and outputs
4. Define the human-machine interfaces, i.e control buttons etc

By comparing the two one can see that they are very similar. The only major difference is that Makris et al. puts more focus into the definition of parameters while Park et al. bundles it together into fewer steps. Thus the two cases is pointing towards the same work flow but with some differences in detail.

Makris et al. succeeded in validating their virtual model but do not explain in what way the dynamics are described within the model. The test was carried out using the software WINMOD for control simulation and INVISION software for the 3D simulation part. A drawback of using these softwares is that to carry out the same

experiment using the real control system the code needs to be translated to generic language, otherwise it is not supported by INVISION. Also the the coordinates of the robots needs to be transformed to a different Euler angle sequence [5]. This makes the software questionable for use with real equipment.

2.8 OPC

OPC is a platform independent standard for data exchange in industry [19] based on a client/server technology [20]. That is, one application acts as a server providing data to all other applications (clients). The OPC server is able to communicate data in real time between all the devices on the shop floor, e.g PLCs, HMI panels and desktop PCs.

The protocol was originally developed in 1995 by representatives from Fisher-Rosemount, Intellution, Opto 22 and Rockwell Software [21]. In 1996 the first OPC specification was released and the OPC foundation were created to develop and maintain the standard.

The main advantage of OPC is that it is an open standard, meaning all devices that are OPC enabled will communicate with each other no matter the brand. This makes it a cheap alternative for manufacturers and gives the user a lot more options when choosing equipment.

2.9 3D Experience Platform

3D Experience is Dassault Systèmes latest PLM platform, it includes software solutions for all parts of a company, from sales to engineering. Dassault Systèmes refer to it as a "Business experience" platform that will make it easy to create value and costumer experiences with an all-in-one software for all departments.

The software is built up around different apps and company roles. Depending on what roles you have, you gain access to different apps. The apps are divided into four areas;

- Social and Collaborative Apps
- 3D Modelling Apps
- Simulation Apps
- Information intelligence Apps

Figure 2.1 shows a screenshot from 3DExperience of roles and Simulation apps.

To carry out a virtual commissioning mainly "3D Modelling Apps'" and "Simulation Apps" are needed. For 3D modelling, *Catia* is used for part modelling and assembling of parts into products. *Delmia* is used for the simulation of robots and production flows as well as designing equipment and tooling from products. A big advantage of 3DExperience is that everything is done in the same software environment, when switching between apps only the available functions and buttons change. E.g when a product is constructed in Catia, just switch to a Delmia app and the product follows and a set of new functions appear.

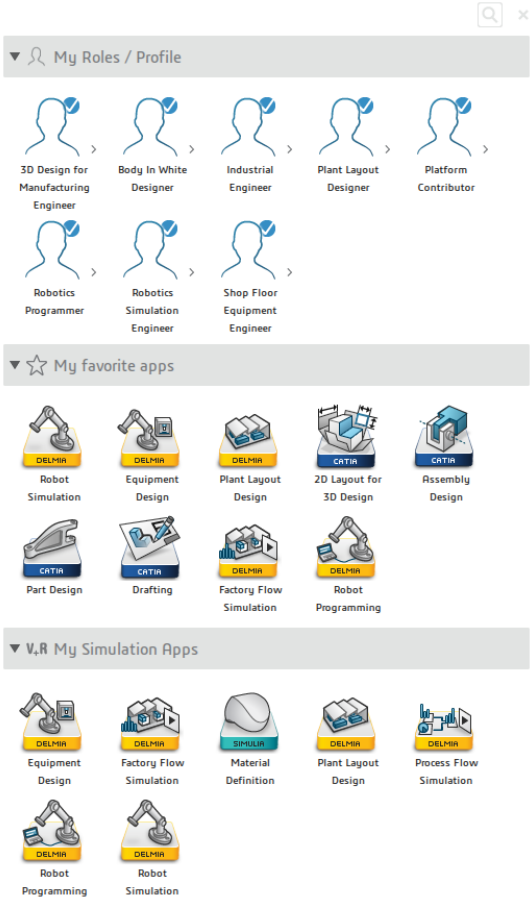


Figure 2.1: Roles and apps in 3DExperience

3

Modelling, Programs and Communication

The project has utilized an adapted method from the studied literature, a flow chart of the methodology can be seen in Figure 3.1. The data collection has mostly been done before the actual model building started but in the cases where there were data missing these two steps has been done in parallel.

3.1 Simulation Model Building

In order to create an accurate 3D model the first step is to learn the software. This has been done by taking courses supplied by Dassault Systèmes covering the different apps that were used and by reading the user assistance manual for 3D Experience.

Once familiar with the software the model building could start, existing layout schematics and CAD data were extracted from Scania servers and imported to 3D Experience. Delmia provides a library of the most common industrial robots on the market ready to import into the model as well as some standard tooling. The equipment not found as existing CAD data has been created using *Catia Part Design* and *Catia Assembly Design*, to add kinematics to the products *Delmia Equipment Design* were used.

In order to make equipment interact in simulation each machine/ tool meant to move needs a *motion controller*, these controllers were added in the *Delmia Robot Simulation* app. One controller is able to control several machines/ tools but to be able to run the machines in parallel it is necessary to define one motion controller for each machine.

The model has successively been expanded and tested in steps in order to make it easier to trouble-shoot. Once all the parts of the model existed the behaviour of the machines were created, since these were seen as black boxes only the time for movements and machining were added. The times for each machine was measured using a stop watch while observing the actual plant.

The verification was made by creating control logic in Delmia and running the model

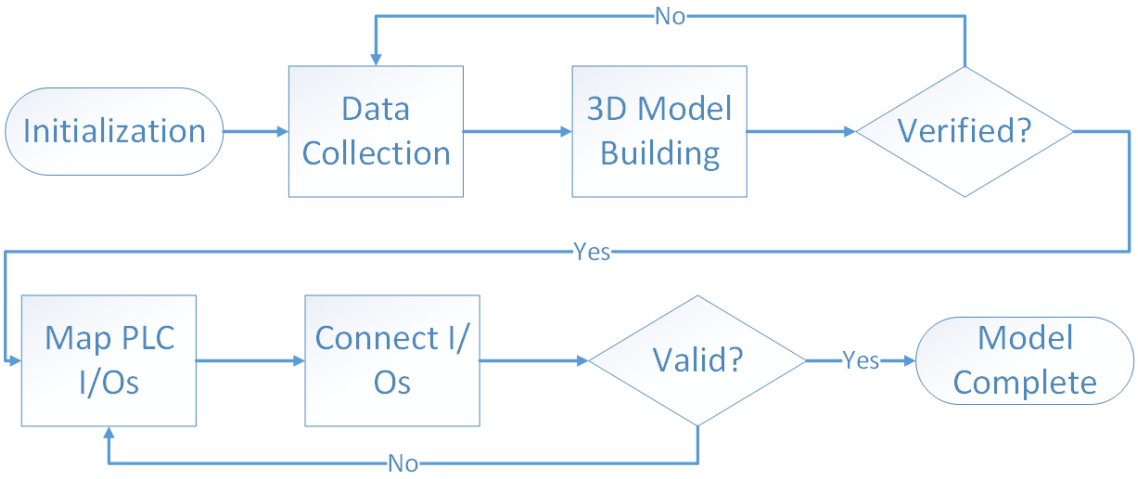


Figure 3.1: An illustration of the used methodology

as a pure factory flow simulation. By doing this the robot movements, machine behavior and task selection could be verified.

3.1.1 Modelling of Machines

Only the hobbing machine existed as an accurate 3D representation, however since this machine has two spindles and thus contains two parts in the flow some modifications were needed. The rotation of the center shaft holding the parts were added and kinematics as well. This imposed some trouble in modelling the logic of the machine since more than just a wait time was needed. The logic quickly got complex in order to handle the case when the cell is empty and also includes transforming the incoming material to the shape of a cog wheel for illustration purposes. A picture showing the modelled machine with movable shaft can be seen in Figure 3.3.

The deburring machine and the marking machine was modelled with simple square shapes that occupies the same volume as the real ones in order to shorten the time needed for modelling. The opening and closing of lid/ doors was included to better visualize the functionality.

3.1.2 Modelling of Centrifuges

The centrifuges existed as 3D shapes but in order to make them interact and move in simulation they had to be remade. Figure 3.2 shows the remade centrifuge where the lid is movable and controlled by a motion controller in simulation. The simulation logic for the centrifuges is simple and only includes setting the correct I/Os, opening and closing of the lid as well as the measured time for centrifuging of the part, implemented as a wait time.

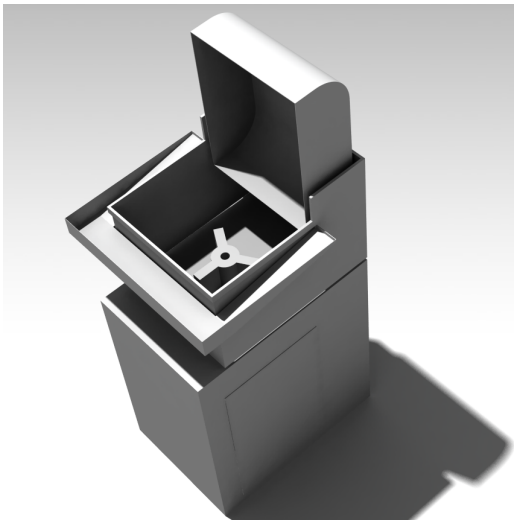


Figure 3.2: 3D Model of Centrifuge

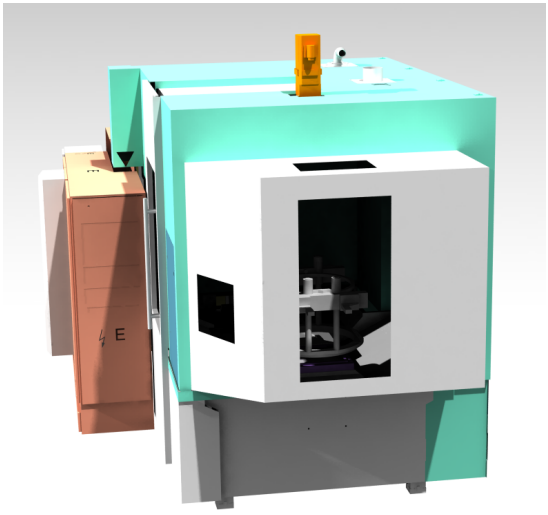


Figure 3.3: 3D Model of Machine one

3.1.3 Modelling of Robot Tool

The tool attached to the robot was not available as CAD drawing so it had to be reproduced. By contacting the suppliers, CAD models of the swivel and grippers were obtained. The remaining parts had to be manually created from 2D drawings found on Scania servers. Kinematics were added to the grippers in order to properly model the movements of the three shoes. The complete tool can be seen in Figure 3.4.

Tooling attached to robots can easily be controlled by the same motion controller as the robot in *Delmia*, but since this tool can be seen as two tools attached to the robot (the two grippers) it was much more convenient to define one motion controller for each gripper. By doing it this way the control logic got shorter since fewer variables were needed to control the grippers and attached parts. The robot still needs to be aware of the tool and its TCP in order to calculate the inverse kinematics. Thus the tool were attached to the robot in *Delmia* but defined not to be controlled by the robot controller. Two TCPs were defined in the robot representation, one for each gripper, so the robot can calculate the correct target depending on wich gripper is to be used. The operation of the grippers is then controlled by I/Os from the robot to the corresponding motion controller for the tool.

3.2 Robot Program

Delmia has the capability to import robot programs written in their native language. In this case the translator converts from ABB RAPID to the flowchart representation Delmia uses. This translator is however some what limited and only supports simple statements like if..else.

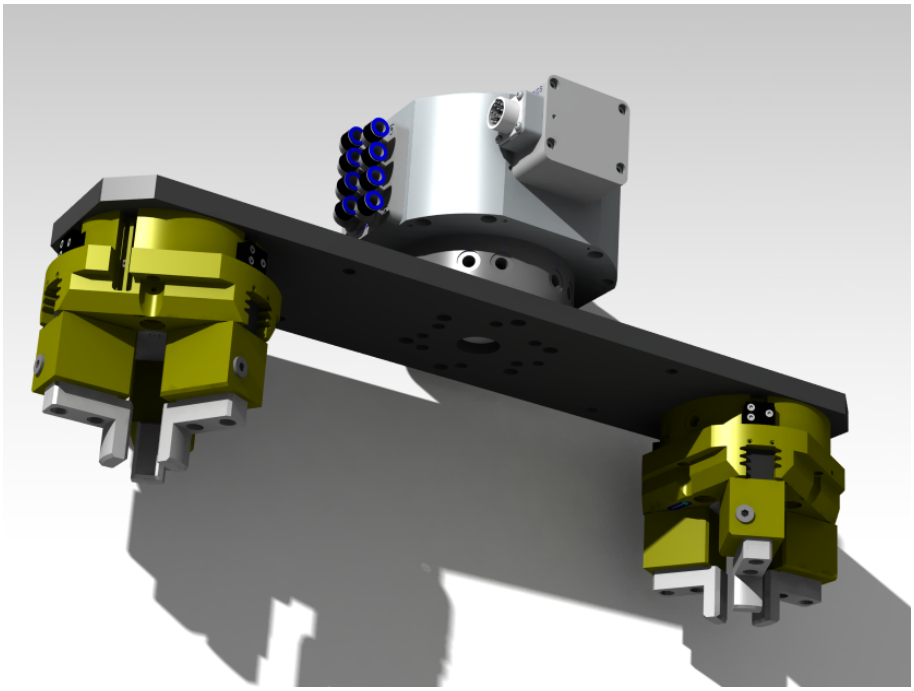


Figure 3.4: 3D Model of Robot Tool

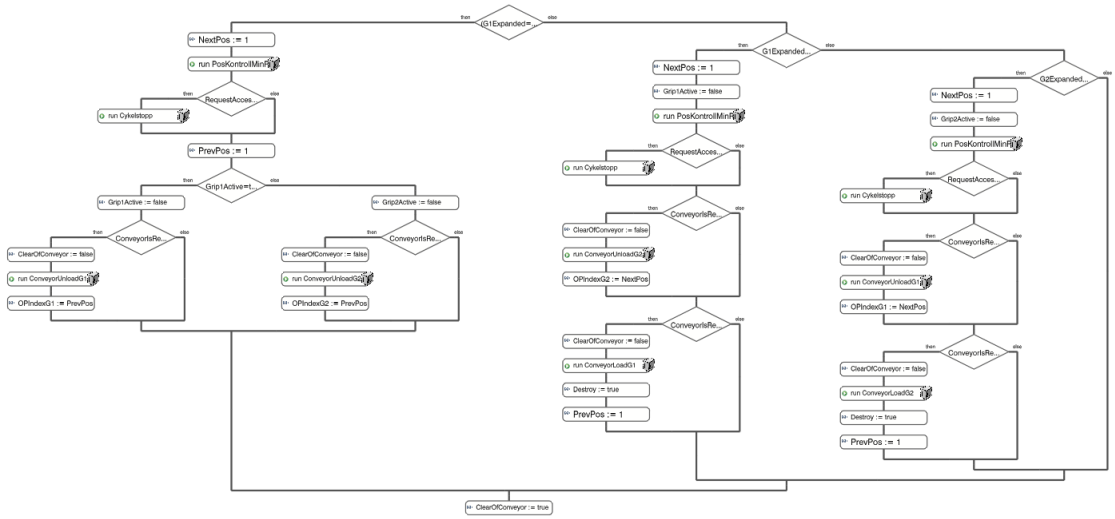


Figure 3.5: Robot routine for loading and unloading the conveyor

The robot program running in the physical cell showed to be several thousand lines long and very complex. The robot program has been made this sophisticated since the robot is the ”brain” in the cell and controls everything else. Because of this, very little of the program could be imported and only the robot targets where imported and used. All the movements and logic in the robot program had to be remade in Delmia as accurate as possible which made the logic for the robot very complex even in the Delmia environment. Since all tasks can be done with either gripper

the logic needs to keep track of that as well, this also increases the complexity of tasks. An example of a robot task can be seen in Figure 3.5, this routine handles the loading and unloading of the conveyor. Programs like this had to be created for each machine / station together with subroutines containing the robot movements for loading / unloading. Also comprehensive routines to handle what task to perform had to be made to be able to call the right robot routine.

In the real robot program the robot is responsible for ordering in and out the fixtures on which the parts travel on the conveyor. The robot thus orders in a fixture with raw material and then orders out the fixture once the ready made part is placed. It can also order in an empty fixture at any time in order to make room for parts coming from the SPCs. Since this routine also is central in keeping track of the different articles it has been excluded from the model in Delmia due to its high complexity. In the simulation a much more simple routine is created by only handling the *create* and *destroy* activities that represent an incoming and exiting part. This approach have excluded the conveyor from the simulation, this would have been an easy feature to include but due to a software bug the creation of conveyors were not possible.

3.3 I/Os and PLC connection

When the model was verified the mapping between the PLCs inputs and outputs could begin. The previously internal I/Os were connected to a *logic controller* interfacing with the OPC server. The OPC server tags is browsable in Delmia and can be chosen to act as I/Os for the logic controller block.

The KEPServerEX was chosen as OPC server since this was previously used by Scania as well as Dassault Systèmes. The server provides an intuitive user interface and also comes with *OPC Quick Client* for monitoring and writing of the defined OPC tags. The server was installed on the same computer as Delmia for simplicity but it could be installed on any computer.

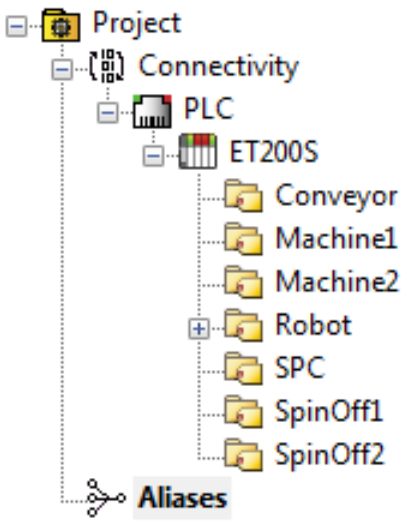


Figure 3.6: Structure of OPC server

The OPC server is structured as in Figure 3.6 where one first define *channels*, in this case only one called ”PLC”, and then adds *devices* to that channel. Below the device level there are groups of OPC tags defined depending on what they are used for. The tags are adressed to the corresponding PLC adress or memory area where the variable is located.

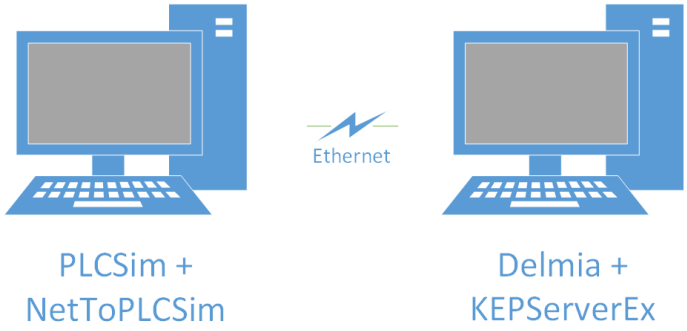


Figure 3.7: OPC connection

Table 3.1: OPC tag adressing

Machine1 Interface				
Name	**Adress**	**Type**	**R/W**	**M Bit**
Machine1 is ready to be loaded	DB161.DBX0.1	Bool	W	M9.0
Machine1 is ready to be unloaded	DB161.DBX0.3	Bool	W	M9.1
OK to enter machine1	DB161.DBX0.5	Bool	W	M9.2
Machine1 loading doors open	DB161.DBX3.1	Bool	W	M9.3
Machine1 is empty	DB161.DBX1.2	Bool	W	M9.4
Robot has loaded machine1	DB161.DBX12.1	Bool	R	-
Robot has unloaded machine 1	DB161.DBX12.3	Bool	R	-
Robot is clear of machine 1	DB161.DBX12.5	Bool	R	-

The connection has been made to a simulated PLC (SIL approach) running in Siemens PLCSim software. PLCSim doesn't support OPC connection per default but there exists a plug-in created by Thomas Wiens [22] that extends PLCSim with a TCP/IP network interface. By using this plug-in and configuring the OPC server channel to use Siemens TCP/IP Ethernet driver the two can communicate. An illustration of the setup can be seen in Figure 3.7.

When Delmia sets an I/O connected to a PLC tag the OPC protocol sends that signal to all clients only once. This means that the signals going to the PLC has to be sent to a memory area that doesn't update each loop of the program. Since the PLC is running in simulation it is possible to connect the OPC tag directly to a "physical" input on the PLC, an *I adress*, but this won't work since the PLC reads this value each loop and will thus only keep a high signal for one sample. Because of this all tags adressing a physical input or DB that are to be written outside the PLC program needs to be assigned to another area in the memory of the PLC and then moved to the correct adress.

To achieve this an extra block of code was added to the PLC program to update the inputs with the correct values. This was done by assigning tags to memory bits instead and then writing the values to the DBs using the PLC program itself. Memory bits keep the assigned value until they are reset which make them useful for this purpose, they do however need to be cross referenced as to not overwrite

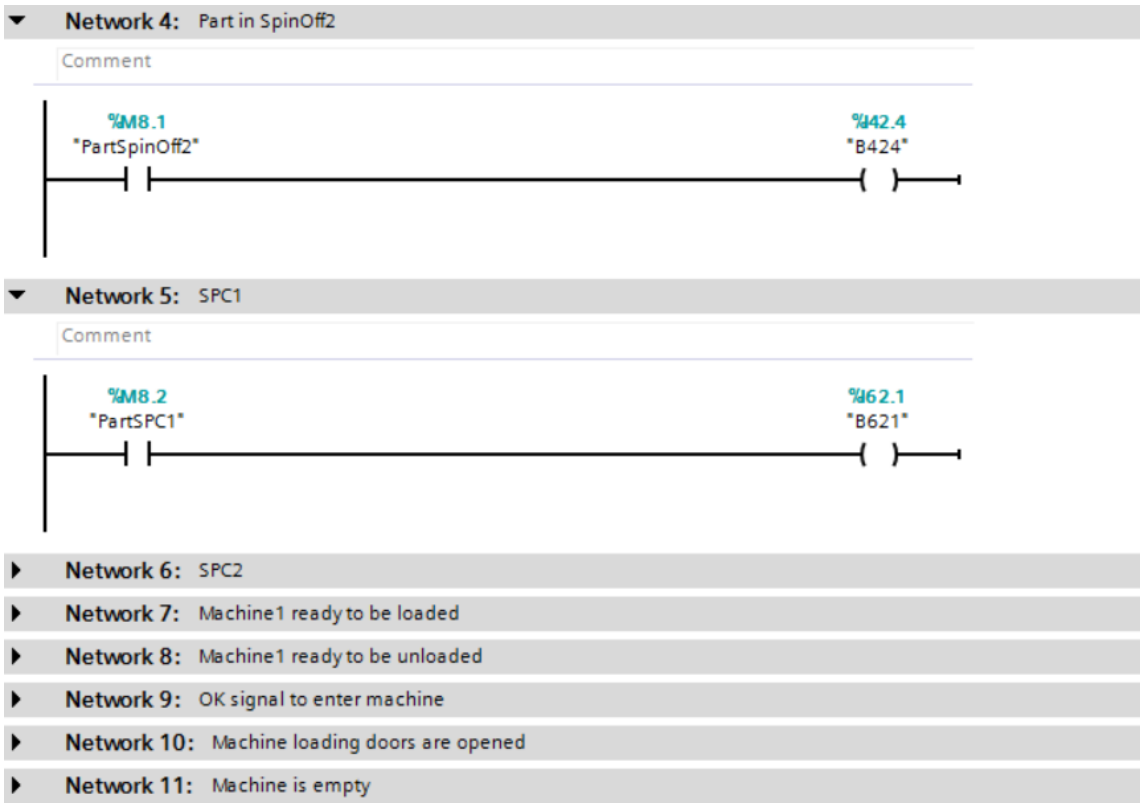


Figure 3.8: Added ladder logic

any other part of the program. All tags writing to the PLC were remapped to memorybits as can be seen in the example in Table 3.1 where the tags for the hobbing machine is written. These memory bits were then used in the added code to set the inputs as in Figure 3.8.

The HMI panel is made in such a way that most of the parameters in the cell can be altered; such as flow, articles to be produced and inspection frequency. The HMI panel and the PLC is connected via Profinet and the panel is also made by Siemens and programmed with WinCC Advanced software, it can thus be simulated in the TIA Portal software. The panel and the PLC require no communication over OPC since Siemens has integrated that communication in the TIA portal. By running both these simulations together with the 3D cell model one can control the simulation with the HMI panel in the same manner as the real system with the exception of articles. The main screen of the HMI panel is shown in Figure 3.9. Note that the cropped text in the Figure is caused by different screen resolution on the computer running the simulation compared to the real HMI panel.

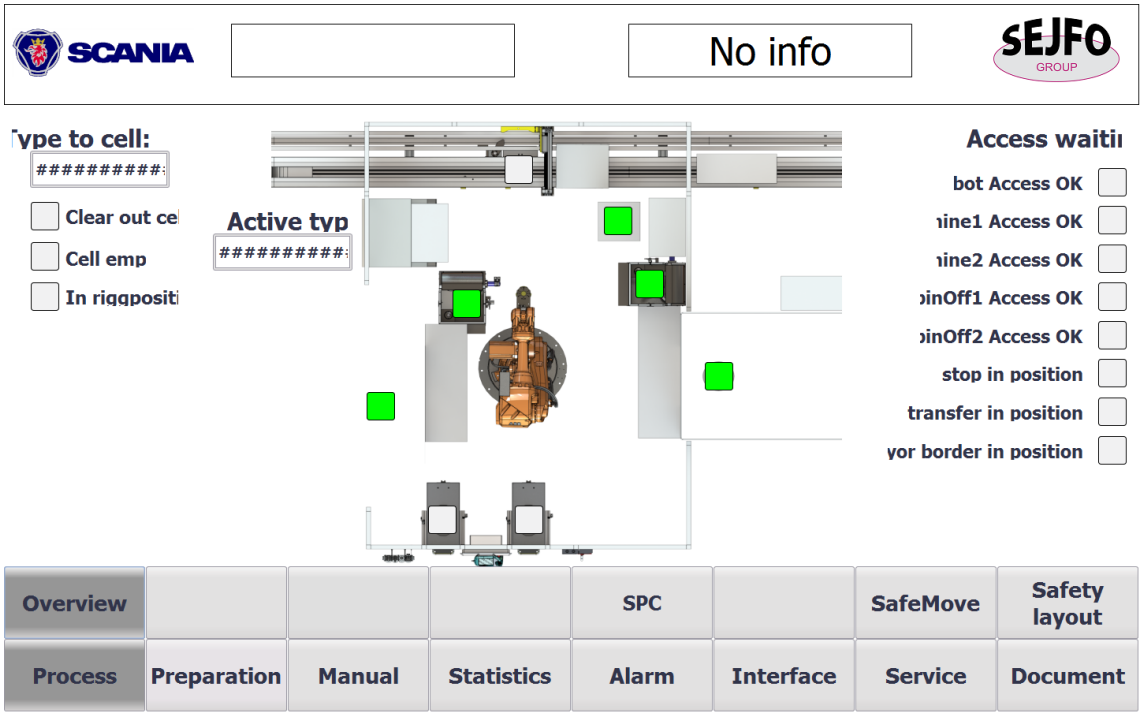


Figure 3.9: Main screen of HMI panel

4

Results and Discussion

The project has shown that is is possible to carry out a virtual commissioning using 3D Experience, but not without effort. It is also possible to control the model using a simulated PLC communicating via an OPC server. In order to implement this approach on a larger scale the design of production systems needs to be done with consideration to the limitations in this software in order to be as efficient as possible. The robot program had to be simplified and remade to a big extent in order to work in Delmia, making it questionable for validation purposes.

Scania is already modelling all their parts, products, tools and fixtures in Catia which is something that would make the 3D model building phase very quick since most of the pieces already exists for reuse. But for the control of the model a lot of consideration needs to be taken towards how to program the robot and PLCs, and what should be the ”smart” component. The Robot should be seen more as tooling and not include complex logic or product- and article management, i.e the programming needs to be as simple as possible since Delmia is limited in its support of advanced features in this area. By incorporating all the complex tasks in the PLC the robot can be controlled by simply calling pre-programmed movements, in this way the signal exchange between the robot and PLC will be simple and easy to incorporate in a simulation. In this particular case the offline programming and simulation when constructing the cell was done in ABB RobotStudio, which supports the full functionality of the robot. If the robot is to continue being the controlling part of the cell RobotStudio is recommended for use to perform a proper simulation useful for validation purposes.

Delmia is currently lacking some functionality that would be desirable for this kind of project. When constructing robot programs the flowchart approach gives a good overview and is fairly easy to debug, but it quickly gets very big. This is mostly due to the lack of functions like *switch..case* and the ability to pass data between tasks. This is possible to work around but the logic gets unnecessarily long. Apart from being very time consuming in construction, the complex logic in Delmia causes loading times for initialization of the simulation to be very long. Especially calling other tasks within the logic causes loading times to increases rapidly and special care needs to be taken to minimize the number of function calls in the logic. In the investigated case the loading times were up to 40 minutes but with reduction of function calls the time were decreased to four minutes. In this case the simulation

was executed on a PC with an Intel Core i7-3940XM 3.0 GHz processor, 32 GB of physical memory, SSD disk and a Nvidia Quadro K4000M graphics card. Overall there also exists some bugs in the software causing the program to crash now and again.

One key feature for performing this kind of simulation is also missing, the possibility to remove specific products in the production flow. Even though all parts/products created in a simulation gets an ID that can be extracted using a sensor, it's not possible to use that ID to remove/ transform that specific product [23]. If this were possible, activities like disposal of products not meeting quality standards could be properly incorporated. The lack of this feature makes the wrong products disappear from simulation when any end up on an SPC hatch since they then exits in a different order than created. One possible way to work around this would be to transform the product at each machine/station since there then only would exist one part of each type in the simulation. This has not been investigated further since it would be quite time consuming to implement.

The investigated cell was developed by a subcontractor and all the parts and data were not easily accessible. If Scania is to continue using subcontractors but still wants to perform simulation of their production systems more demands needs to be put on the deliveries. They need to include updated CAD and layout models of the lines and equipment as well as thorough function descriptions in order to minimize rework on Scanias side.

One big advantage with using 3DExperience in the whole organization is that all models and parts are accessible to all departments. This means that the rework is minimized since each department doesn't need to construct their own model, instead they can use a lot of existing models. Even though a 3D model needed for virtual commissioning is extensive it can be reused in several stages of the production system life span. It can first be used for commissioning and verification of PLC code, later the same model can be used for discrete event simulation, investigating changes and layout changes just to mention some things. This has the possibility to give big cost savings but it requires the whole process to be done in-house or good deliveries from subcontractors.

The learning phase in this kind of project is quite long, and with the acquired knowledge it would probably take a couple of weeks maximum to redo a similar project. This is important to keep in mind when deciding to introduce this method, the first projects will probably not be as beneficial as possible since the knowledge has to be acquired first. This is a problem that is emphasized in the literature as well. However there are some issues addressed in literature that wasn't visible in this project. There are no need for extensive modeling with automatas, generic- and kinematic models since this is taken care of within Delmia. By ignoring these time consuming tasks a lot of time is saved. The work flow can also be said to follow the literature and it is probably the most efficient way of doing it.

A recommendation for future work is to investigate the accuracy of a virtual commissioning carried out with 3DExperience on a cell where the complex tasks is located

in the PLC. Another question to answer is whether or not the code is completely fail-proof without creating automatas and analysing each reachable state. After the evaluation of these scenarios a conclusion can be drawn as to if this is the best tool to perform this kind of validation. For proper validation a real PLC is recommended, in this case it was not interesting since the robot programs already was modified to a large extent. This makes it impossible to pinpoint problems arising from the use of a simulated PLC.

Bibliography

[1] A.-L. Andersen, K. Nielsen, and T. D. Brunoe, "Prerequisites and Barriers for the Development of Reconfigurable Manufacturing Systems for High Speed Ramp-up," *Procedia {CIRP}*, vol. 51, pp. 7 – 12, 2016, 3rd {ICRM} 2016 International Conference on Ramp-Up Management. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S221282711630498X

[2] G. Reinhart and G. Wünsch, "Economic application of virtual commissioning to mechatronic production systems," *Production Engineering*, vol. 1, no. 4, pp. 371–379, 2007.

[3] P. Hoffmann, R. Schumann, T. M. Maksoud, and G. C. Premier, "Virtual Commissioning Of Manufacturing Systems A Review And New Approaches For Simplification." in *ECMS*, 2010, pp. 175–181.

[4] M. Gregor, S. Medvecky, J. Matuszek, and A. Stefanik, "Digital factory," *Journal of Automation Mobile Robotics and Intelligent Systems*, vol. Vol. 3, No. 3, pp. 123–132, 2009.

[5] G. M. S. Makris and G. Chryssolouris, "Virtual commissioning of an assembly cell with cooperating robots," *Advances in Decision Sciences*, pp. 1–11, 2012.

[6] R. E. Shannon, "Introduction to the Art and Science of Simulation," in *Proceedings of the 30th Conference on Winter Simulation*, ser. WSC '98. Los Alamitos, CA, USA: IEEE Computer Society Press, 1998, pp. 7–14. [Online]. Available: http://dl.acm.org/citation.cfm?id=293172.293175

[7] R. G. Ingalls, "Introduction to Simulation," in *Proceedings of the 40th Conference on Winter Simulation*, ser. WSC '08. Winter Simulation Conference, 2008, pp. 17–26. [Online]. Available: http://dl.acm.org/citation.cfm?id=1516744.1516754

[8] J. Zhang, V. Le, M. Johnston, S. Nahavandi, and D. Creighton, "Discrete Event Simulation Enabled High Level Emulation of a Distribution Centre," in *Computer Modelling and Simulation (UKSim), 2012 UKSim 14th International Conference on*, March 2012, pp. 470–475.

[9] F. Danielsson, P. Moore, and P. Eriksson, "Validation, off-line programming and optimisation of industrial control logic," *Mechatronics*, vol. 13, no. 6, pp.

571 – 585, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957415802000302

[10] F. Auinger, M. Vorderwinkler, and G. Buchtela, "Interface driven domain-independent modeling architecture for "soft-commissioning" and "reality in the loop"," in *Simulation Conference Proceedings, 1999 Winter*, vol. 1, 1999, pp. 798–805 vol.1.

[11] H. Schludermann, T. Kirchmair, and M. Vorderwinkler, "Soft-commissioning: Hardware-in-the-loop-based Verification of Controller Software," in *Proceedings of the 32Nd Conference on Winter Simulation*, ser. WSC '00. San Diego, CA, USA: Society for Computer Simulation International, 2000, pp. 893–899. [Online]. Available: http://dl.acm.org/citation.cfm?id=510378.510505

[12] T. Berger, D. Deneux, T. Bonte, E. Cocquebert, and D. Trentesaux, "Arezzo-flexible manufacturing system: A generic flexible manufacturing system shop floor emulator approach for high-level control virtual commissioning," *Concurrent Engineering*, vol. 23, no. 4, pp. 333–342, 2015. [Online]. Available: http://cer.sagepub.com/content/23/4/333.abstract

[13] R. Phillips and B. Montalvo, "Using emulation to debug control logic code," in *Simulation Conference (WSC), Proceedings of the 2010 Winter*, Dec 2010, pp. 1673–1677.

[14] C. Starner and M. Chessin, "Using emulation to enhance simulation," in *Simulation Conference (WSC), Proceedings of the 2010 Winter*, Dec 2010, pp. 1711–1715.

[15] 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*, ser. WSC '12. Winter Simulation Conference, 2012, pp. 252:1–252:12. [Online]. Available: http://dl.acm.org/citation.cfm?id=2429759.2430099

[16] M. Oppelt, G. Wolf, O. Drumm, B. Lutz, M. Stöß, and L. Urbas, "Automatic model generation for virtual commissioning based on plant engineering data," in *Proceedings of the 19th World Congress of the International Federation of Automatic Control*, 2014.

[17] 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. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2288430014500292

[18] C. M. Park, S. M. Bajimaya, S. C. Park, G. N. Wang, J. G. Kwak, and K. H. Han, "Development of Virtual Simulator for Visual Validation of PLC Program," in *2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, Nov 2006,

pp. 32–32.

[19] “What is OPC?” Online, 2017, accessed 2017-01-13. [Online]. Available: https://opcfoundation.org/about/what-is-opc/

[20] “OPC Interoperability: Open Connectivity Through Open Standards,” Online, 2017, accessed 2017-01-13. [Online]. Available: https://www.kepware.com/en-us/products/kepserverex/opc-interoperability/

[21] “History of OPC,” Online, 2017, accessed 2017-01-13. [Online]. Available: https://opcfoundation.org/about/opc-foundation/history/

[22] T. Wiens, “NetToPLCSim,” Online, 2015, accessed 2017-03-14. [Online]. Available: http://nettoplcsim.sourceforge.net/

[23] Vincent van Houtte, Personal Communication, 2017.