CHALMERS
UNIVERSITY OF TECHNOLOGY

Integrated virtual commissioning of a ROS2-based collaborative and intelligent automation system

Citation for the original published paper (version of record):
Erös, E., Dahl, M., Hanna, A. et al (2019). Integrated virtual commissioning of a ROS2-based collaborative and intelligent automation system. IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, 2019-September: 407-413.
http://dx.doi.org/10.1109/ETFA.2019.8869444

(article starts on next page)

Integrated virtual commissioning of a ROS2-based collaborative and intelligent automation system

Endre Erős
Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden
endree@chalmers.se

Martin Dahl
Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden
martin.dahl@chalmers.se

Atieh Hanna
Research & Technology Development
Volvo Group Trucks Operation
Gothenburg, Sweden
atieh.hanna@volvo.com

Anton Albo
Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden
anton.albo@chalmers.se

Petter Falkman
Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden
petter.falkman@chalmers.se

Kristofer Bengtsson
Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden
kristofer.bengtsson@chalmers.se

***Abstract*—In order to adapt to stricter system delivery and integration requirements, virtual commissioning (VC) has become a well adopted practice in industry. VC is getting increasingly integrated into the overall engineering process, where the control software is continuously tested with the virtual plant model. At the same time, collaborative and intelligent automation systems are becoming an important part of modern industries. In these complex systems, humans perform operations together with collaborative robots, intelligent machines and smart tools. However, performing VC of such complex, distributed and heterogeneous systems demands new ways of interfacing different hardware and software components. This paper discusses the requirements, process and results of integrated virtual commissioning of an industrial collaborative and intelligent automation system use-case. Moreover, this industrial use-case illustrates challenges and exemplifies the need to use the next generation Robot Operating System (ROS2) due to its robust communication layer as well as easy integration with smart devices and algorithms.**

***Index Terms*—virtual commissioning, robot operating system, factory automation, control and communication architecture, intelligent automation, collaborative robot systems, human-in-the-loop.**

I. Introduction

Global demand for high quality products with short life-cycles, that are reasonably priced, is growing. Meanwhile, the variability of such products is rapidly increasing to meet demands of a wider range of customers. In order to stay competitive, companies are decreasing innovation cycles and product delivery times [1]. These changes cause a paradigm shift in companies towards more flexibility and reconfigurability to respond quickly and efficiently to changing production requirements and market demands [2].

Volvo, being a global truck manufacturer, experiences these changes [3]. The vision for a Volvo factory of the future is that is must be more flexible in order to handle the challenges imposed by this paradigm shift [3].

* Supported by UNIFICATION, Vinnova, Produktion 2030.

One of the adaptation strategies is the inclusion of collaborative [4] and intelligent automation systems in production. The ergonomic environment in such production systems enables operators to interface the system through a range of smart human-machine interfaces (HMIs). While collaborative robots and machines support operators at workstations, safety is ensured by dedicated camera and sensor systems and on-line path planning [5]. An example implementation of a safety system using a Microsoft Kinect V2 camera is shown in Fig. 1, [6]. Moreover, humans and robots use common smart tools and are interchangeable when it comes to standard tasks [7].

Another strategy is to perform virtual commissioning (VC) [8] using simulated production systems. The purpose of VC is to enable the control software, which controls and coordinates different devices in a production station, to be tested and validated before physical commissioning.

However, a knowledge gap exists between simulation and control system development since they are traditionally decoupled activities in industry [9].

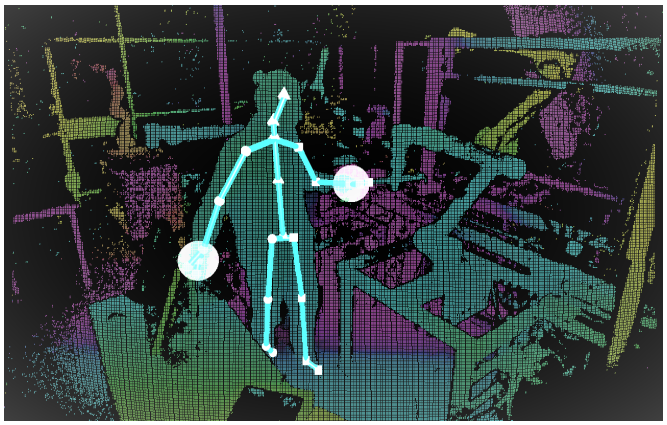

Fig. 1. Example of a dedicated operator safety camera feed.

A concept of integrated virtual commissioning (IVC) emerges, which aims to integrate VC into the standard engineering workflow as a continuous support for automation engineers [10]. Additionally, virtual preparation joins this workflow, extending IVC into integrated virtual preparation and commissioning (IVPC) [11].

However, performing IVPC of such complex heterogeneous systems demands a common platform that supports integration of different smart software and hardware components.

In order to ease integration and development of different types of online algorithms for sensing, planning and hardware control, various platforms have emerged as middle-ware solutions. One platform that stands out is the Robot Operating System (ROS) and its success can be measured by having over 16 million downloads in 2018 [12].

The next generation ROS2 is currently developed, where the developers implemented the Object Management Group standard DDS as the communication middleware considering it to be scalable, robust and well-proven in mission-critical systems. As a result, DDS enables large scale distributed control architectures and implementation of ROS in real-world industrial automation systems.

In the industrial use-case presented in this paper, the forementioned gap in the engineering workflow is bridged using ROS2 as a communication middle-ware and a platform for continuous integration and testing of simulation, visualization and control.

This paper explains the tools and methods used to perform IVPC of a collaborative and intelligent automation system. Section 2 briefly explains the use-case and goes through the components showing the overall complexity of the system. A quick overview of traditional VC is extended with newer concepts of VC in Section 3. Section 4 touches upon humans as part of the loop in VC and discusses different methods of including humans in VC. Section 5 goes through the components of the control system and Section 6 shows implementation of some of those components. The paper is concluded in Section 7 with a short overview of the paper and the achieved results.

II. The collaborative and intelligent automation system use-case

It is challenging to develop collaborative and intelligent automation systems [7] since they involve smart algorithms, advanced sensors, human operators and collaborative robots. Because of this, it is almost impossible to develop all aspects of the system without IVPC.

In this paper, this is highlighted in a collaborative robot assembly station located in a Volvo Trucks engine manufacturing facility in Skövde, Sweden (shown in Figure 2).

This industrial use-case demonstrates the design of a workstation where both humans and robots work in a collaborative and co-active fashion. Particularly in this use-case, a collaborative robot and a human operator perform assembly operations on a diesel truck engine.

In the use-case station, an Automated Guided Vehicle (AGV) that carries a diesel truck engine and an autonomous mobile platform (MiR100) that carries the kitted material to be mounted on the engine, enter the collaborative robot assembly station. An engine ladder frame, three oil filters and several oil transport pipes are to be mounted on the engine in this station.

Fig. 2. The collaborative robot assembly station

Before the collaborative execution of these operations can begin, an authorized operator has to be verified with a RFID reader. After verification, the operator is greeted by the station and instructions for tasks that are to be performed are shown on a screen. If no operator is verified, some operations can still be executed independently by the robot, however, violation of safety zones around the robot trigger a safeguard stop.

In the case of collaborative assembly, a dedicated camera system keeps track of the human assuring safe coexistence. During positioning of the AGV and the MiR100 in the assembly station, a Universal Robots (UR10) robot attaches to a special end-effector needed for manipulation of the ladder frame. Since the ladder frame is quite heavy, transporting it from the kitted MiR100 to the engine is done together by the robot and the human.

After placing the ladder frame on the engine, the operator informs the control system with a button press on a smart watch after which the UR10 leaves the current end-effector and attaches itself to a smart tool used for tightening bolts. During this tool change and as it is instructed on the screen, the operator is placing twelve pairs of bolts needed for assembling the ladder frame on the engine and indicates completion of this task through the smart watch. Now, the control system knows that bolts are in place and the UR10 starts the tightening operation with the smart nutrunner.

Since tightening of these bolts takes some time, the operator can mount three oil filters during that time. If the robot finishes the tightening operation first, it leaves the smart tool in a floating position above the engine and waits for the operator.

The operator uses the smart watch again to let the system know that the oil filters are in place. This event makes the robot attach to a third end-effector and start performing the oil filter tightening operations. During the same time, the operator attaches two oil transport pipes on the engine, and uses the same smart tool that the robot has left floating to tighten plates that hold the pipes to the engine.

After executing these operations, the AGV with the assembled engine and the empty MiR100 leave the collaborative robot assembly station. A video showing the described use-case in action can be found here: https://youtu.be/YLZzBfY7pbA

III. Traditional virtual commissioning

Simulation of production systems is a well adopted practice in modern industry. Industry leading software tools like Process Simulate and Delmia include support for VC, which exposes the simulation model to a real control system. Using these tools for designing control systems and performing VC has become a standard in industry.

Over the years, the VC community specified several commissioning configurations [13], [8] that resulted in terms like hardware-in-the-loop, reality-in-the-loop and constructive commissioning. As shown in Table I, these specifications are defined by combinations of components being real or virtual.

Plant	Controller	Commissioning type
Real	Real	Real (physical) commissioning
Real	Virtual	Reality-in-the-loop commissioning
Virtual	Real	Hardware-in-the-loop commissioning
Virtual	Virtual	Constructive commissioning

TABLE I
Traditional commissioning classification

Testing and integrating the physical production system with the real control system has been traditionally referred to as physical commissioning. However, in order to reduce the amount of on site man-hours during physical commissioning [14], the real control system is coupled with a simulation model of the production system creating a hardware-in-the-loop setup. This configuration is commonly known as VC [8].

Performing the inverse of VC can be beneficial in situations when debugging the control system is needed. In this case, the physical production system is controlled by a simulated controller in a setup known as reality-in-the-loop.

When designing a new control system, the natural way is to start with offline programming where all components are simulated. This configuration is also known as constructive commissioning [8].

Performing VC can reduce testing and integration time, as well as help detect undesired behavior before physical commissioning. However, it is usually the case that creating simulation models requires extensive modelling effort. Because of the cost associated with this effort, it is crucial that the created models provide as much additional value as possible.

This value can be increased by embedding VC into the engineering workflow. Instead of using VC as the last step before physical commissioning, IVPC enables simulation supported production preparation and automation engineering by simultaneous development of the control system and the virtual plant [11]. Figure 3 shows an early stage simulation that serves as continuous support for development of the control system

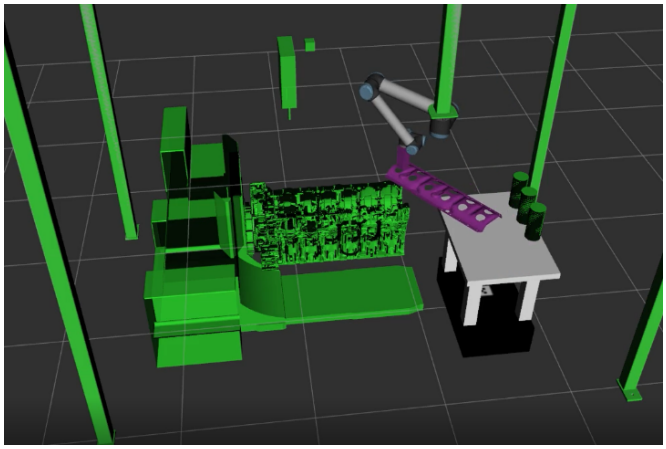

Fig. 3. Early simulation work is beneficial for control system development

for the collaborative and intelligent automation system use-case.

The traditional commissioning classification does not really apply in complex systems that rely on intelligent control. Containing algorithms beyond standard executing sequences as described in Section 5, the control system is the component that needs commissioning.

Since the start of it's use, VC focused on highly automated systems where human interaction and interference has been limited. However, automation is today introduced in traditionally operator intensive areas such as assembly stations. This puts new requirements on VC.

IV. Human-in-the-loop commissioning

Considering the role of humans in VC, especially in collaborative systems, human-in-the-loop commissioning has only recently become a topic of conversation [15].

In this chapter, three classes of the term human-in-the-loop are proposed, i.e. real, virtual and immersed. Always considering the use of the real controller, Table II shows the proposed human-in-the-loop (HITL) commissioning configurations.

Virtual-human-in-the-loop commissioning considers inclusion of a simulated human mannequin in the virtual plant environment. This can be beneficial for ergonomics verification [16], [17] as well as for safety testing using simulated safety equipment.

Plant	Human	Commissioning type
Real	Real	Real (physical) commissioning
Virtual	Virtual	Virtual-HITL commissioning
Virtual	Immersed	Immersed-HITL commissioning

TABLE II
Human-in-the-loop commissioning classification

Means to detect simulated mannequins in a virtual plant involve simulated camera systems, light curtains, laser proximity sensors and other, where disturbances can also be allowed in sensor simulations. Moreover, behavior generation algorithms

can ensure that the simulated mannequin covers large sets of different realistic kinematic behaviors [18].

Various HMIs can be emulated during IVPC in order to verify plant behavior during human interaction.

Humans can also be included in the commissioning loop by ”immersion”, which represents a real-time mapping from a real to a virtual human. Immersed-human-in-the-loop commissioning utilizes technologies like motion-tracking to ”project” actual human kinematic behavior into the virtual plant environment. As it is shown in Fig. 4, [19], an Intelligently Moving Mannequin (IPS IMMA) [16], [17] maps actual human behavior that is inferred from a human tracking system into the virtual environment [6].

Interactable emulated and real HMIs can include buttons, screens, smart watches, eye movement and hand gesture tracking cameras, safety sensors, voice recognition and others to enable the immersed human to interface the system.

An immersion method that is worth mentioning is virtual reality (VR) [20] which can be combined with motion tracking to provide a ”true immersion” experience. Additionally, platforms exist that support creation of industrial VR workspaces suitable for immersion of several humans [21].

A large amount of data can be collected during immersed commissioning which can later be used to drive behavior generation algorithms [22].

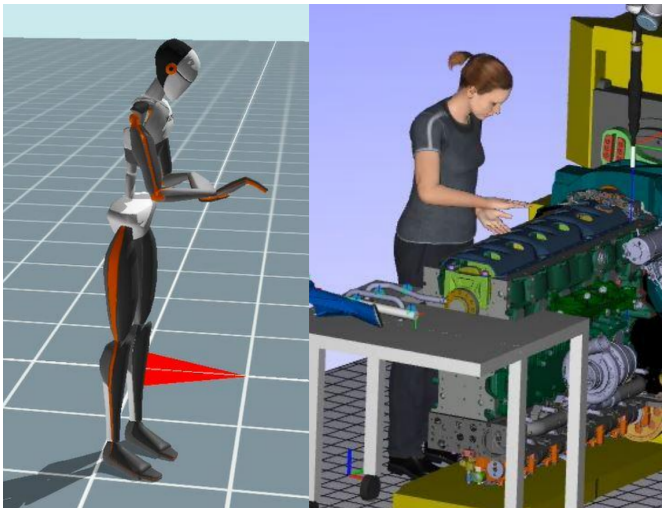

Fig. 4. Motion capture immersion with IPS IMMA where motions can be translated from the same safety camera system feed shown in Fig. 1

V. The control in collaborative and intelligent automation systems

From the challenges and requirements described so far, it is evident that there is a need for a control system that is able to step out of bounds of traditional automation methods and that can be commissioned using IVPC. In this paper, Sequence Planner (SP) [23] has been used as a tool for modeling, analyzing and control of large scale intelligent automation systems. SP includes supporting algorithms for a variety of use cases related to modeling, synthesizing control logic [24], formal verification, optimization, automated planning and visualizing complex operation sequences in different projections.

The core modeling aspects in SP are resources, abilities and operations. A resource is defined by a set of state variables that represents the current state of the resource. Some variables can be directly measured using sensors, others are used to represent the commands and some variables are used to estimate state that is not measured in the real system.

The resources have a set of abilities that they are able to perform. These abilities execute based on the current state of the resource and they update the command and estimated state variables. Abilities control everything in the system and are also used in the deliberation and execution loop of SP that is based on automated planning [25], which generates a plan deliberating what abilities to execute to reach a specific goal. These goals are defined by a number of high-level planning operations that continuously refine the current goal of the execution.

Most models in SP are based on transitions and variables [26], which are defined using extended finite automata (EFA) [27]. EFAs are finite automata extended with variables that are used in guards and actions associated with transitions. A transition in an EFA is enabled if and only if its corresponding guard formula (predicate) is true. After the transition is taken, a set of variables is updated by action functions. A transition table containing the behavior of an EFA models how an ability modifies the state of a resource in the system, Table III.

A. Resource

To exemplify abilities, let us consider the engine ladder frame tightening operation using the nutrunner described in Section 2. The nutrunner used in the industrial scenario described in Section 2 is quite complex. However, in order to exemplify the control of this system, a simplified version of the nutrunner is used from this point. The simplified nutrunner resource state variables can be defined as:

$$r_{nr} = \{run^c, run^m, tqr^m, fail^m\} \tag{1}$$

These Boolean variables make up the current state of the nutrunner, where run^c represents the command from SP to run the nutrunner forward and run^m is the signal from the nutrunner to SP showing if the toll is running forward or not.

After receiving the command to start running forward, the nutrunner engages tightening which will eventually result in reaching a specified torque tqr^m or failing $fail^m$.

B. Ability

The ability *run nutrunner forward* of the *nutrunner* resource models the task of tightening by starting to run the motors forward until a pre-programmed torque (tqr^m) has been reached or the tightening operation has failed $fail^m$. Table III shows the transitions of the *run nutrunner forward* ability, where each line makes up one possible transition of the ability.

For the purpose of this example, several lines are filtered out from ROS2 messages used for communication between SP and the nutrunner. As shown in Listing 1, a *command* message

predicate name	predicate (guard)	control actions	effects
enabled	$\neg run^c \wedge \neg run^m$	$run^c = T$	-
starting	$run^c \wedge \neg run^m$	-	$run^m = T$
executing	$run^c \wedge run^m \wedge \neg tqr^m$	-	$tqr^m = T \vee fail^m = T$
finished	$run^c \wedge run^m \wedge tqr^m$	$run^c = F$	-
failed	$run^c \wedge run^m \wedge fail^m$	$run^c = F$	-
resetting	$\neg run^c \wedge run^m$	-	$run^m = F$, $tqr^m = F$, $fail^m = F$

TABLE III
THE RUN NUTRUNNER FORWARD ABILITY

is sent out from SP as a state based command to the nutrunner while a *state* message provides SP with the current state of the tool.

```
# /nutrunner/command
bool run_c              # run_tool_forward

# /nutrunner/state
bool run_c              # got_run_tool_forward
bool run_m              # tool_running_forward
bool tqr_m              # programmed_torque_reached
bool fail_m             # tool_failed
```

Listing 1: Messages to and from the nutrunner.

Listing 1 shows only the measured and the command variables. Since the desired result of running the *run nutrunner forward* ability is to tighten a pair of bolts and there are no sensors for keeping track of bolt states, an *estimated* state variable $\hat{b} \in$ *(empty, placed, tightened)* can be introduced in order to keep track of the states of the bolts and enable the on-line planner to generate plans to tighten all bolts.

In order to maintain the simplicity of the example, the tightening of only one pair of bolts is considered. Now we can avoid the estimated bolt state and move on with the example.

The *enabled* predicate of the *run nutrunner forward* ability is declared as:

$$\neg run^c \wedge \neg run^m \tag{2}$$

However, this does not mean that the ability will start executing immediately after the guard is fulfilled by updating the variables with the corresponding control actions. This is due to the fact that the on-line planner deliberates if the control action should be executed or not.

C. Effect

To use the abilities in formal planning algorithms, measured variables must be updated so that the planner knows what to expect from the real system. In order to emulate an ability without its actual device or simulation, one or more starting and executing effects per ability are defined.

Effects model how measured state variables behave during execution of an ability. For example, issuing the command to start the nutrunner, the command variable run^c is set to true. Before the nutrunner actually responds that the tool has started to run forward, the *starting* predicate is now true. The effect mapped to this predicate models that the nutrunner will eventually be running forward.

After the tool has responded that it is running forward, the *executing* predicate is true. Now the effects model that bolts will eventually be either tightened or that the tightening operation will fail. These effects are extensively used during IVPC since they are used to emulate device behavior. This is explained in the following section.

VI. MODEL-BASED ROS2 COMPONENT GENERATION

Based on abilities modeling the behavior of resources, SPs component generator module generates a set of ROS2 nodes during compile-time. Five nodes per resource are generated: *interfacer*, *emulator*, *simulator*, *driver* and *test*. Moreover, message types shown in Listing 1 are also generated based on the model in SP as well as all other necessary ROS2 package components.

The *interfacer* node serves as the standardized interface node between SP and the nodes of the resource. *Emulator* nodes are completely auto-generated based on the abilities defined in SP and are used during IVPC.

Simulator and *driver* nodes implement the actual simulation and device control. *Test* nodes are generated based on the SP model and they implement automatic testing of *simulator* and *driver* nodes based on the SP model.

A. Emulator

The internal state of an emulation does not have to reflect the internal state of the target which it is emulating, it only has to mimic the observable behavior to match an existing target. Considering this, the internal structure of ROS2 *emulator* nodes is quite simple.

In order to emulate real device behavior during IVPC, effects happen outside of SP in dedicated ROS2 *emulator* nodes. These nodes emulate devices by executing effects specified in the SP model. It has to be noted that at a certain point in time, only one of the *emulator*, *simulator* or *driver* nodes per resource is active.

In order to exemplify the *emulator* nodes, let us continue with the nutrunner example. After generating the nodes based on the model in SP, the *emulator* node now mimics the observable behavior of the modeled resource based on the effects specified in the model. This means that the node listens to commands from SP and publishes its state like a *simulator* or a *driver* node.

The *emulator* node contains the model of the resource, evaluating all predicates based on the messages received

from SP. If some predicates are evaluated to be true, their corresponding effects are appended to a list of effects to be executed. After a command message has been received from SP and all predicates have been evaluated, the effects from the list of effects are executed.

This updates the *measured* variables which are then published from the node in the state message. This way, the model in SP can be continuously tested and improved using *emulator* nodes since the complete behavior of the resource based on the model in SP is achieved.

If the model changes during the development process, the *emulator* nodes are re-generated, capturing the change from the SP model. This shows that model-based code generation no only supports IVPC but is an essential part of it.

Moreover, since humans are resources in SP which have abilities that they can perform, human effects represent the expected human behavior after receiving an instruction. These human effects are contained in auto-generated human *emulator* nodes which emulate the human behavior based on the effects in the model.

B. Simulation

Contrasting to emulation, simulation involves modeling the underlying state of the target. Since collaborative and intelligent automation systems are composed of many different components, there is no single simulation environment that can support modeling of all aspects of the system.

ROS2 is a valid middleware to bridge different simulation environments with other ROS-based as well as non ROS-based components. This is supported by DDS being the robust communication layer in ROS2. There is a number of available implementations of DDS, and since DDS is standardized, ROS2 users can choose an implementation from a vendor that suits their needs. This is done through a ROS Middleware Interface (RMW), which also exposes Quality of Service (QoS) policies.

QoS policies allow ROS2 users to adjust data transfer in order to meet the desired communication requirements. This feature is crucial, especially in distributed and heterogeneous systems, since setups are usually unique. Some of these QoS policies include setting message history and reliability parameters. These features are beneficial in a real system as well as when performing VC, since robust communication between simulation environments and other components can be challenging to achieve.

These simulation environments are interfaced with *simulator* nodes that are partially generated based on the model in SP. Initial SP interfacing templates are generated, however, interfacing details to the actual simulation environments have to be manually specified. This can be done via an import to the node since that way the common parts of the node can be re-generated if the model changes, without influencing the added interfacing part.

The virtual plant is implemented using ROS2, virtual machines (VMs) and Docker containers (DCs). VMs and DCs are used to segment the virtual model and mimic the distributed nature of the system. Since ROS2 does not rely on a rosmaster [28], physical segmentation of the virtual plant model is easily achieved in order to distribute the computational load between machines.

A software used during IVPC of the described setup that successfully implements the potential of the virtual world is Industrial Path Solutions (IPS) [5]. IPS is a mathematics based software tool for automatic verification of assembly feasibility, design of flexible components, motion planning and optimization of multi-robot stations and simulation of key surface treatment processes [5].

Figure 5 shows a virtual scene in IPS generated from the point cloud of the real assembly station. IPS is used as a tool to automatically generate collision free paths of the UR10 in the station on-line as well as off-line. In both cases, ROS2 serves as a middleware for IPS to communicate with the UR10, either with the real hardware or the simulator.

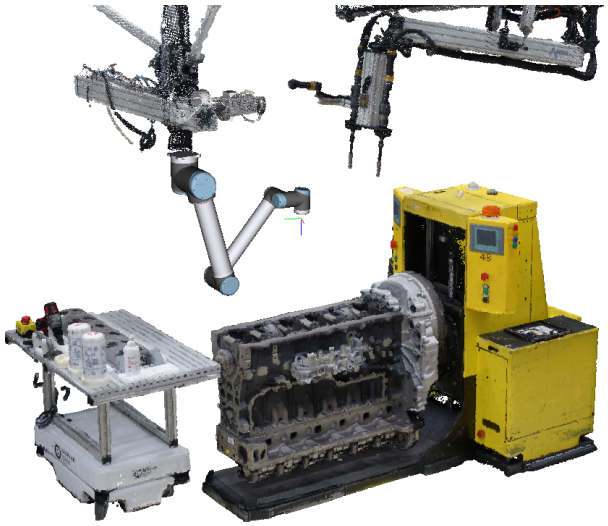

Fig. 5. The collaborative and intelligent automation system use-case in IPS

Moreover, having model based ROS2 component generation makes it straightforward to include a virtual or immersed human via the *simulator* or *driver* nodes to achieve human-in-the-loop IVPC.

C. Targeted IVPC and VM contained simulations

To extend the usability of code generation, multiple launch files are generated for various types of testing and commissioning purposes.

The first step in the engineering workflow would be testing the model with emulator nodes, iterating the model until the desired behavior is reached. Afterwards, a simulation or real hardware can be interfaced for one of the resources, while the others remain emulated. This way, real, simulated and emulated components can be combined in a mix-and-match fashion to achieve targeted model property testing.

This testing is supported using the *test* node that emulates SP by publishing model-based generated test commands, trying to break the behavior of the *simulator* node. After a failed

test, the model is improved and the tests run again until they don't fail anymore.

Moreover, simulation environments and their accompanying *simulator* nodes can be contained in VM's and DC's to mimic the distributed nature of a real industrial setup.

VII. Conclusion

This paper discusses IVPC of an engine assembly station where intelligent control algorithms are commissioned in a ROS2-based setup. To support IVPC, auto-generated ROS2 components are used to continuously test and improve the model in SP.

Moreover, due to ROS2's robust underlying communication layer DDS, it is seamless to mix real, simulated and emulated components during IVPC. As essential components of the intelligent control system, abilities and effects are briefly explained utilizing the description of the use-case provided in Section 2.

The concept of human-in-the-loop commissioning is discussed and a classification proposed where two different means of including humans in VC are isolated. A supporting literature review provides sufficient background to recognize the difference between virtual and immersed human-in-the-loop commissioning and a need to propose the mentioned classification. ROS2-based human-in-the-loop IVPC using auto-generated SP model-based emulator and simulator nodes and its benefits are discussed.

References

[1] P. Hoffmann, R. Schumann, T. Maksoud, and G. Premier, "Virtual commissioning of manufacturing systems - a review and new approaches for simplification," 06 2010, pp. 175–181.

[2] H. ElMaraghy, T. AlGeddawy, A. Azab, and W. ElMaraghy, "Change in manufacturing – research and industrial challenges," in *Enabling Manufacturing Competitiveness and Economic Sustainability*, H. A. ElMaraghy, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 2–9.

[3] "Volvo GTO Vision," https://www.engineering.com/PLMERP/ArticleID/18868/Vision-and-Practice-at-Volvo-Group-GTO-Industry-40-and-PLM-in-Global-Truck-Manufacturing.aspx, [Online; accessed 14-Apr-2019].

[4] R. Bloss, "Collaborative robots are rapidly providing major improvements in productivity, safety, programing ease, portability and cost while addressing many new applications," *Industrial Robot: the international journal of robotics research and application*, vol. 43, no. 5, pp. 463–468, 2016. [Online]. Available: https://doi.org/10.1108/IR-05-2016-0148

[5] "IPS," http://www.fcc.chalmers.se/software/ips/, [Online; accessed 14-Apr-2019].

[6] "Kinect ROS2," https://github.com/EresDavid/ROS-Kinect, [Online; accessed 22-Apr-2019].

[7] A. Hanna, P.-L. Götvall, M. Ekström, and K. Bengtsson, "Requirements for designing and controlling autonomous collaborative robots system-an industrial case," *Advances in Transdisciplinary Engineering*, pp. 139–144, 2018.

[8] 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

[9] S. T. Mortensen and O. Madsen, "A virtual commissioning learning platform," *Procedia Manufacturing*, vol. 23, pp. 93 – 98, 2018, Advanced Engineering Education and Training for Manufacturing Innovation8th CIRP Sponsored Conference on Learning Factories (CLF 2018). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2351978918304712

[10] M. Oppelt and L. Urbas, "Integrated virtual commissioning an essential activity in the automation engineering process: From virtual commissioning to simulation supported engineering," *Proceedings, IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*, pp. 2564–2570, 02 2015.

[11] 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, 8th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2016.

[12] D. Lu, "The 2018 ROS Metrics Report," https://discourse.ros.org/t/the-2018-ros-metrics-report/6216/2, 2018, [Online; accessed 25-Feb-2019].

[13] F. Auinger, M. Vorderwinkler, and G. Buchtela, "Interface driven domain-independent modeling architecture for "soft-commissioning" and "reality in the loop"," in *WSC'99. 1999 Winter Simulation Conference Proceedings. 'Simulation - A Bridge to the Future' (Cat. No.99CH37038)*, vol. 1, Dec 1999, pp. 798–805 vol.1.

[14] N. Shahim and C. Mller, "Economic justification of virtual commissioning in automation industry," in *2016 Winter Simulation Conference (WSC)*, Dec 2016, pp. 2430–2441.

[15] M. Metzner, J. Bnig, A. Blank, and J. Franke, "human-in-the-loop-virtual commissioning of human-robot collaboration systems," 04 2018.

[16] L. Hanson, D. Högberg, J. Carlson, R. Bohlin, E. Brolin, N. Delfs, P. Mrdberg, S. Gustafsson, A. Keyvani, and I.-M. Rhen, "Imma intelligently moving manikins in automotive applications," 05 2014.

[17] D. Högberg, P. Castro, P. Mårdberg, N. Delfs, P. Nurbo, P. Fragoso, L. Andersson, E. Brolin, and L. Hanson, *DHM Based Test Procedure Concept for Proactive Ergonomics Assessments in the Vehicle Interior Design Process: Volume V: Human Simulation and Virtual Environments, Work With Computing Systems (WWCS), Process Control*, 01 2019, pp. 314–323.

[18] C. Esteves, G. Arechavaleta, J. Pettre, and J.-P. Laumond, "Animation planning for virtual mannequins cooperation," *ACM Transactions on Graphics - TOG*, 01 2004.

[19] P. Castro, D. Högberg, H. Ramsen, J. Bjursten, and L. Hanson, *Virtual Simulation of Human-Robot Collaboration Workstations: Volume V: Human Simulation and Virtual Environments, Work With Computing Systems (WWCS), Process Control*, 01 2019, pp. 250–261.

[20] M. Dahl, A. Albo, J. Eriksson, J. Pettersson, and P. Falkman, "Virtual reality commissioning in production systems preparation," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2017, pp. 1–7.

[21] P. Galambos, Á. Csapó, P. Zentay, I. M. Fülöp, T. Haidegger, P. Baranyi, and I. J. Rudas, "Design, programming and orchestration of heterogeneous manufacturing systems through vr-powered remote collaboration," *Robotics and Computer-Integrated Manufacturing*, vol. 33, pp. 68 – 77, 2015, special Issue on Knowledge Driven Robotics and Manufacturing. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0736584514000738

[22] M. Manns, S. Mengel, and M. Mauer, "Experimental effort of data driven human motion simulation in automotive assembly," *Procedia CIRP*, vol. 44, pp. 114 – 119, 2016, 6th CIRP Conference on Assembly Technologies and Systems (CATS). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2212827116003140

[23] M. Dahl, K. Bengtsson, P. Bergagrd, M. Fabian, and P. Falkman, "Sequence planner: Supporting integrated virtual preparation and commissioning," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 5818 – 5823, 2017, 20th IFAC World Congress. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896317309047

[24] P. Ramadge and W. Wonham, "Wonham, w.m.: The control of discrete event systems. proc. ieee 77(1), 81-98," *Proceedings of the IEEE*, vol. 77, pp. 81 – 98, 02 1989.

[25] M. Ghallab, D. Nau, and P. Traverso, *Automated Planning and Acting*, 1st ed. New York, NY, USA: Cambridge University Press, 2016.

[26] M. Dahl, E. Erős, A. Hanna, K. Bengtsson, and P. Falkman, "Sequence planner - automated planning and control for ros2-based collaborative and intelligent automation systems," https://arxiv.org/abs/1903.05850, 2019.

[27] M. Skoldstam, K. Akesson, and M. Fabian, "Modeling of discrete event systems using finite automata with variables," in *2007 46th IEEE Conference on Decision and Control*, Dec 2007, pp. 3387–3392.

[28] M. Quigley, B. Gerkey, and W. D. Smart, *Programming Robots with ROS: A Practical Introduction to the Robot Operating System*, 1st ed. O'Reilly Media, Inc., 2015.