Early-Stage Analysis of Cyber-Physical Production Systems through Collaborative Modelling

Mihai NEGHINA[1], Constantin-Bala ZAMFIRESCU[1], Ken PIERCE[2]

[1] *Lucian Blaga Univ. of Sibiu, Faculty of Engineering, Department of Computer Science and Automatic Control, Bd. Victoriei nr.10, Sibiu, Zip code 550024, Romania*
{mihai.neghina,zbcnanu}@gmail.com

[2] *School of Computing, Urban Sciences Building, Newcastle University, 1 Science Square, Newcastle Helix, Newcastle upon Tyne, NE4 5TG, UK*
kenneth.pierce@newcastle.ac.uk

Abstract. This paper demonstrates the flexible methodology of modelling Cyber-Physical Systems (CPSs) using the INTO-CPS technology through co-simulation based on Functional Mock-up Units (FMUs). It explores a novel method with two main co-simulation phases: homogeneous and heterogeneous. In the first phase, high-level, abstract FMUs are produced for all subsystems using a single discrete-event formalism (the VDM-RT language and Overture tool). This approach permits early co-simulation of system-level behaviours and serves as a basis for dialogue between subsystem teams and agreement on interfaces. During the second phase, model refinements of subsystems are gradually introduced, using various simulation tools capable of exporting FMUs. This heterogeneous phase permits high-fidelity models of all subsystems to be produced in appropriate formalisms. This paper describes the use of this methodology to develop a USB stick production line, representing a smart system of systems. The experiments are performed under the assumption that the orders are received in a Gaussian or Uniform distribution. The focus is on the homogeneous co-simulation phase, for which the method demonstrates two important roles: first, the homogeneous phase identifies the right interaction protocols (signals) among the various subsystems, and second, the conceptual (system-level) parameters identified before the heterogeneous co-simulation phase reduces the huge size of the design space and creates stable constraints, later reflected in the physical implementation.

Keywords: Co-Simulation, Cyber-physical Production Systems, Homogeneous and Heterogeneous Modelling, Design Space Exploration

1. Introduction

In the development of truly complex Cyber-Physical Systems (CPS), a model-based approach can be an efficient way to master system complexity through iterative and incremental development. Such systems are often made in an ad hoc manner by combining different CPSs without explicit and comprehensive coordination among the design teams, whose expertise and engineering background are often limited to the domain of their respective subsystem. These domains include their own focus, terminology and modelling approaches. Engineers of physical systems often use continuous-time (CT) formalisms, realised as differential equations, to produce high-fidelity simulation of physical phenomena. Meanwhile, software engineers tend to adopt discrete-event (DE) formalisms, focusing on the logical behaviours of control systems. To a large extent, these CPSs emerge and evolve through iterative and incremental developments, from digital models through multiple, costly physical prototypes where teams are unable to truly collaborate and design

faults found at a late stage are extremely costly. This paper provides insights from the collaborative model-based design of a manufacturing system for assembling USB-OTG (USB On-The-Go) sticks, developed in the iPP4CPPS innovation project [1], as a representative example of a distributed and heterogeneous system in which the subsystems are all cyber-physical in nature [2].

In our previous work [3], we illustrated how the co-simulation technology [4] can be used to gradually increase the detail in a collaborative model (co-model) following a mixed "discrete-event first" and "interface first" methodology [5]. The combination of these two methodologies was needed to accommodate the specific co-simulation approaches with the general and well-established methodologies for engineering manufacturing systems, such as agent-oriented and component-based, used by the design teams.

The paper builds on that previous work [3], expanding significantly on the experience of the design teams. While [3] focused on the digital design and validation of the CPS-based manufacturing systems, giving a high-level view of the two main phases of development (the homogeneous co-simulation phase and the heterogeneous co-simulation phase), this experience report extends it by going into the details of the homogeneous co-simulation, underlining not only what could be decided and refined at that stage, but also how the co-simulation helped reduce the huge size of the design space by identifying optimal system-level parameters that translate into stable constraints, reflected in the physical implementation of the manufacturing system for the heterogeneous co-simulation.

The remaining part of this paper starts with a brief introduction of the methods for generating discrete-event first models as part of the INTO-CPS technology [6][7] in Section 2. This is followed in Section 3 by the CPS development approach, with emphasis on the necessity of having a homogeneous co-simulation phase, as well as underlining the specific challenges of the subsystems that will be analysed with simulation experiments. Sections 4 and 5 provide the experimental setup, the most important experimental data, and comparisons and interpretations of the experiment outcomes. Finally, Sections 6 and 7 discuss the implications and limitations of the approach from a methodological point of view, and offer concluding remarks related to the final, balanced system parameters.

2. Discrete-Event First Modelling with INTO-CPS

This section introduces the main technologies used for building the initial models of the homogeneous phase of development, including the INTO-CPS co-simulation technology and the use of Discrete-Event first (DE-first) approach of generating DE models in Overture [8]. This section also presents shortcomings in directly applying the workflow of INTO-CPS and the motivation for an alternative approach.

2.1 The INTO-CPS technology

The INTO-CPS co-simulation technology is a collection of tools and methods that have been linked to form a tool chain for model-based design of CPSs. The approach recognizes the need for software, control and mechatronics engineers to collaborate in the design of CPSs, while overcoming the fact they each domain has adopted its own formalisms and vocabularies. Rather than suppress this diversity by requiring all disciplines adopt the same general-purpose notation, INTO-CPS embraces this diversity by integrating the formalisms at a semantic level [9][10][11][12][13], allowing engineers to collaborate using familiar modelling techniques and methods.

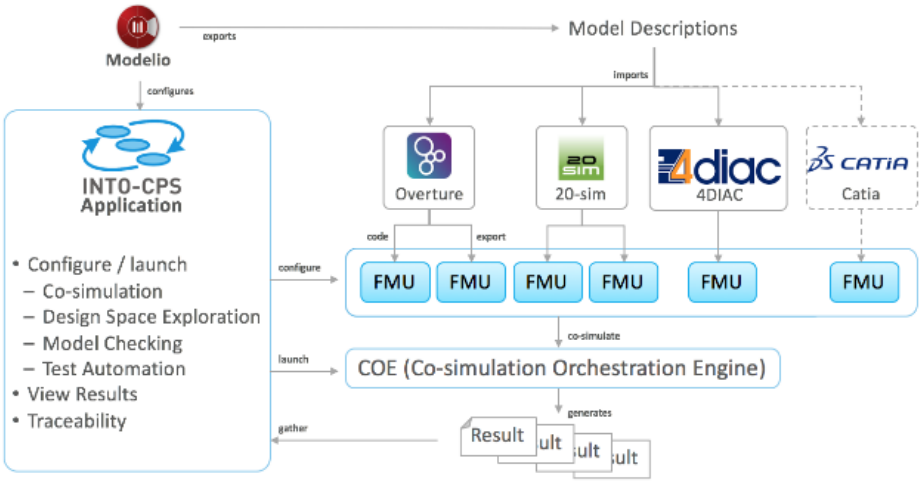


Figure 1. General workflow of the INTO-CPS tool chain used for model-based design on CPSs

The overall workflow and services from the INTO-CPS tool chain are illustrated in Figure 1. The INTO-CPS tool chain is centred around co-simulation of heterogeneous models using the Functional Mock-up Interface (FMI) standard [6]. The FMI standard allows models from different tools and formalisms to be packaged as Functional Mock-up Units (FMUs), which can be combined and analysed through co-simulation. Each FMU has a model description that describes its interface and can be provided as a black-box to protect Intellectual Property (IP) contained in the details of the model. INTO-CPS includes a co-simulation engine, called Maestro [14], that fully implements version 2.0 of the FMI standard and has been successfully tested with over thirty tools [6]. Around this co-simulation core, the INTO-CPS tool chain links additional tools to support model-based design throughout development. A Systems Modelling Language profile (also known as a SysML profile) is provided and supported by the Modelio tool [15], allowing FMU model descriptions to be captured and linked to requirements. The model descriptions can capture both physical and cyber parts of the system, can be exported for use in modelling tools [16]. The profile also allows these descriptions of FMUs to configure co-simulations and other forms of analysis supported by INTO-CPS.

Specific support for importing model descriptions and producing skeleton models is provided by tools in the INTO-CPS tool chain: Overture [8], supporting DE modelling; 20-sim [17] and OpenModelica [18], both supporting CT modelling. These tools also guarantee export of FMUs, which can then be used in a co-simulation with Maestro [14]. FMU export is included in an increasing number of industry tools. During the heterogenous co-simulation in the later stages of our case study we used 4DIAC [19], an open-source tool for development of industrial control systems, and CATIA [20], an industry-standard software suite for computer-aided design and manufacturing. Both are shown in their respective positions within the INTO-CPS tool chain in Figure 1.

2.2 Initial Models

Given the workflow and tool chain described above, a natural step might be to follow it directly: define a CPS model using the SysML profile; export model descriptions and import them into the appropriate modelling tools; model the respective components and behaviours in each; and finally generate FMUs and combine them for co-simulation.

This approach allows different teams to work on the constituent models separately, however it can be prone to failure. It requires FMUs from all teams to be available before integration testing through co-simulation can begin. If one or more teams are delayed, all other teams will be delayed. This can also lead to late discovery of problems, which collaborative modelling is designed to avoid. Similarly, if there is a single team producing all FMUs sequentially, again co-simulation is delayed until later in the project.

A potential strategy to mitigate these risks is to have each team produce quick, initial versions of FMUs as soon as they can and perform integration testing with these models. The initial models can then be updated in an iterative manner towards more detailed models, with each team able to move at its own pace, and with previous versions providing fall-backs in the case of problems and baselines for regression testing. This approach might be more difficult in some modelling paradigms however, where quick and simple models might not function sufficiently well for testing.

We adopt a variation on this strategy in this paper, a DE-first approach. This involves the same style of producing initial FMUs and replacing them as more detailed models become available, but rather than using each individual formalism, a single DE formalism is used for all initial models. In this way, a simple and abstract model of the whole system is made and analysed within a DE modelling environment such as Overture. The behaviour of the system and the interfaces and functions of the components can be sketched, and assumptions tested at the beginning of the process. A DE formalism is selected because these are designed to capture abstract and logical behaviours, often described in terms of interfaces, and therefore well-suited to this task. Such a DE-first approach is described in Fitzgerald et al. [21] (using two rather than many models).

2.3 Discrete-Event First with VDM-RT/Overture

In applying a DE-first approach to an FMI setting using Overture [8], the principles of the Vienna Development Method (VDM) [22][23], a set of modelling techniques successfully applied in both research and industrial application, have been followed. First, given a set of model descriptions, possibly generated from a SysML model, a single VDM Real-Time (VDM-RT) [24] project is created, containing: a class for each FMU, with port objects corresponding to the interface given in the model description for the FMU; a main (system) class that instantiates appropriate port objects and instances of each FMU class to which it passes the ports; a world class that provides a method as an entry point for simulation by starting the threads of the FMU objects and blocks until simulation is complete.

Figure 2 provides class and object diagrams and shows such a set up using two constituent models, FMU1 and FMU2. Such a model can be simulated within Overture to see how the FMUs behave and interact. Once sufficient confidence in these initial models is gained, they can be exported individually as FMUs and integrated in a co-simulation.

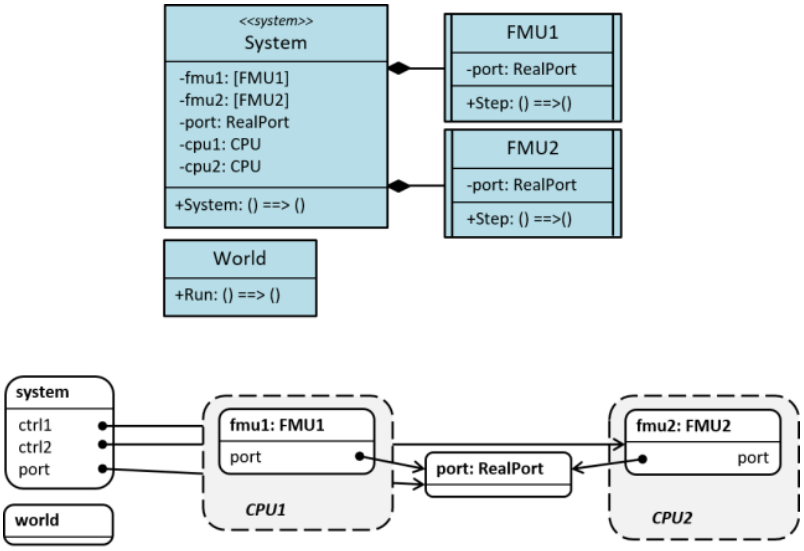


Figure 2. Class diagram showing two simplified FMU classes created within a single VDM-RT project, and an object diagram showing them being instantiated as a test.

The Overture FMI plug-in can then be used to export an FMU from each individual project unit, these can then be combined in a co-simulation. These FMUs can be revised if problems are found, then replaced with higher-fidelity models. The models could be retained for later use and as a fall-back in case of future problems in integration.

3. The CPS development approach

The case study was developed as part of iPP4CPPS innovation project [1]. The ambition of iPP4CPPS project was to contribute to the advancement of engineering methods and tools employed in manufacturing CPS-based production systems by:

- Demonstrating the proper methodological steps for achieving a working heterogeneous co-simulation (with units modelled in various dedicated tools) of a relatively complex system that requires diverse and multidisciplinary teamwork;
- Extending the libraries and functionalities of the employed tools to cope with the real industrial needs.

This section starts with a brief description of the production line case study. Then, the particularities of the most important subsystems are discussed, to allow the identification of the parameters analysed in the experiments. Finally, the general overview of the development approach is given, with emphasis on the homogeneous co-simulation phase that the paper is focused on.

3.1 Description of the case study

The case study concerned the manufacture of USB-OTG sticks (shown in Figure 3) as a tractable but representative production line example. This production line has the classical characteristics of a smart product, as defined by Mühlhäuser [25]:

- Situated: recognition of situational context, in terms of order identification, availability of parts and slots, awareness of perturbations (e.g. vibrations) and malfunctions, etc.;
- Personalized: personalization of USB-OTG sticks according to orders, as well as the capability of handling cancellations and order modifications during the production phase;
- Adaptive: adaptation of the line to the customer orders, for instance according to order urgency and the level of perturbations (vibrations);
- Pro-active: anticipation of the production line owner intentions by restricting functionality in certain conditions in order to minimize the risk of malfunctioning or extending the testing in uncertain conditions of luminosity;
- Business-awareness: energy-efficient behaviour unless receiving special urgency requests by the customers;
- Network capable: although not tested in this project, each production unit has intrinsic communication capabilities with external products (including similar production lines).

To capture the value-adding processes in Industry 4.0 [16][24][26][27], the case study includes distinct subsystems to reflect order-placing users, as well as the required infrastructure that enables the other CPSs to function properly.

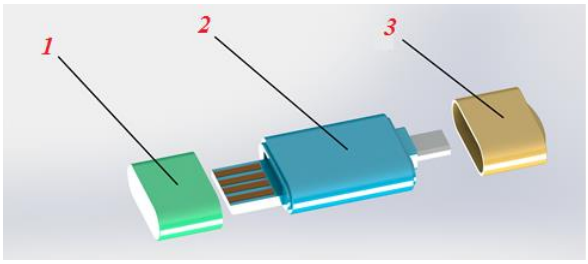


Figure 3. Example of USB-OTG unit consisting of three component parts:
1) left cap; 2) middle (body) of the stick; 3) right cap

The subsystems identified as necessary for the case study and represented in Figure 4 (along with the communication patterns between them) are:

- The Human-Machine-Interface (HMI), which is handling incoming orders, being responsible for interpreting and transmitting them correctly to the Part Tracker;
- The Part Tracker, which is the infrastructure unit capable of communicating with the HMI, capable of relaying the order information to the production system and of gathering data on the status of any order received;
- The Warehouse, which assembles the USB-OTG sticks from the stored parts;
- The Robotic Arm capable of moving parts or assembled USB-OTG sticks;
- The Wagons, which are the transportation units between subsystems of the production line;
- The Test Station, which is the processing station for checking the conformity to order requirements.

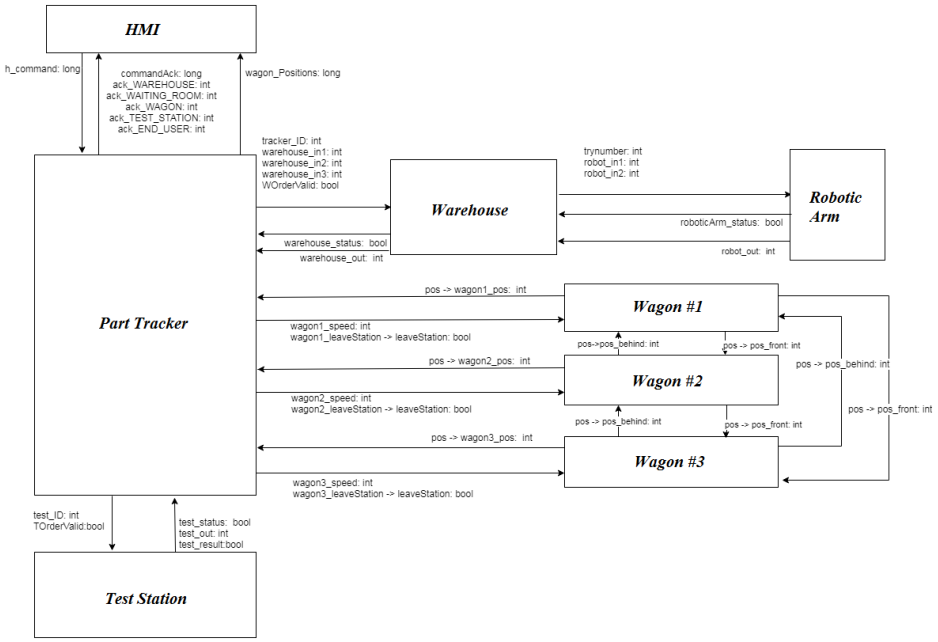
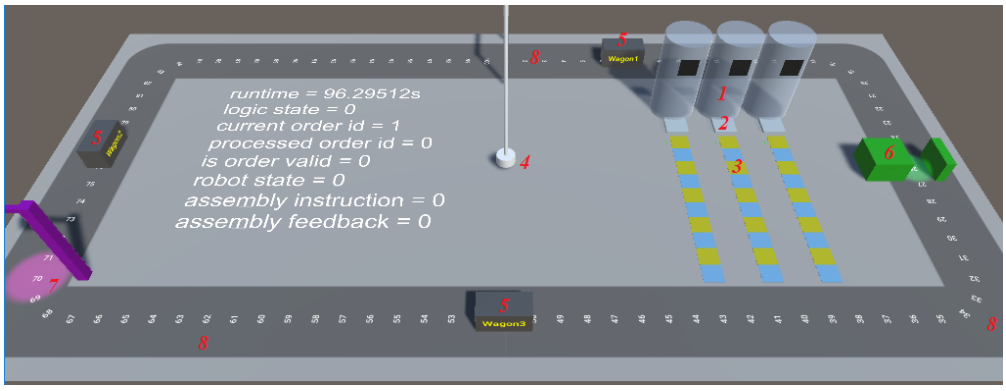


Figure 4. Connections between subsystem models

Additionally, the simulation includes a dynamic 3D graphics unit (created in Unity) for visualization purposes of the system's dynamics, in both the homogeneous and heterogeneous phase. Figure 5 shows the plant layout as depicted in the 3D rendering of the simulation, as well as the demo stand with the physical units realized during the project. The communication between units contains both simple (straightforward) messages, requesting the setting of a certain value or indicating the current value or state of a subsystem, as well as composed messages that need to be decoded and the information extracted from them before that information can become useful. The purpose of the composed messages is twofold: to ensure that certain bits of information arrive simultaneously (as opposed to them coming on different message lines that may become unsynchronised or for which further synchronisation logic might be needed) and to account for the possibility of coded messages (that might be interesting in applications with significant noise, where error correcting codes might become useful).

For instance, a request from the Part Tracker to the Wagons to assume a certain speed or the feedback of the Wagon positions to the Part Tracker is done with straightforward messages (the value requested) on dedicated lines (that only carry these types of messages and nothing else). On the other hand, the order requests from the HMI to the Part Tracker or their acknowledgement (feedback) contain multiple pieces of information in each.



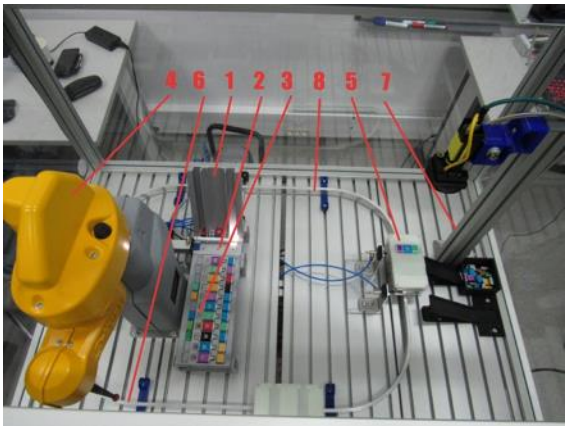


Figure 5. Layout of the simulated production line, as depicted in the Unity rendering (above), and the physical demo stand (left), containing:
1) the Warehouse stacks;
2) the Warehouse assembly box;
3) the Warehouse memory boxes;
4) the Robotic Arm;
5) the Wagons on the track;
6) the loading station;
7) the test station;
8) the circular track for the wagons.

3.2. The subsystems of the manufacturing plant

As the entire system and its constituent subsystems have been described in detail in [3], the focus in this section is on the specifics of the implementations relevant to the analyses and conclusions in later sections.

The Human-Machine-Interface (HMI) subsystem. The case study was provided with two complementary implementations of the HMI unit, although they cannot be used both at the same time. The 4DIAC+MQTT implementation is meant for gathering user heuristics and allowing for real-time placement of orders. However, more useful for the current analysis is the Overture (VDM-RT) implementation, which reads orders from a file. This approach is both flexible and powerful. The flexibility stems from the possibility of creating scenarios with various amounts of orders for covering statistical possibilities, while the power comes from the repeatability of experiments. Having the same input file, the co-simulation can be run with various parameters, ensuring that the same input orders come at the same time, thus generating a detailed picture of the behaviour of the system in controlled, repeatable experiments.

The Warehouse unit. The Warehouse assembles the USB-OTG sticks from the component parts. As shown in Figure 6, it contains stacks for each type of component part, an assembly box for the actual assembly of the items and memory boxes (with the same number of slots available for each part type) for storing components that do not fit the current order. The memory boxes may also be a source of component parts for new orders, if the requested colour is available.

To simplify the analysis initially, the simulated Warehouse is considered to have an unlimited number of parts in the stacks (there would be no need to re-fill the stacks at any time). Also, the memory boxes would have the same size (or number of slots) for all part types (left, middle and right). Upon receiving an assembly order, the Warehouse first looks in the memory boxes for available parts of requested colours. If these do not exist, either because the memory boxes are empty or filled with differently coloured parts, the Warehouse drops the parts from the stacks. However, the parts in the stacks are not arranged in any specific order. Unless it is a lucky hit, a dropped part has the wrong colour and would need to be stored in the memory boxes (if a slot is available) or discarded for recycling (if memory boxes are full). To keep a reasonable symmetry, all part types have the same range of colours for the users to choose from. Up to eight colours can be selected; if a colour is available for a component part, it is also available for all other component parts.

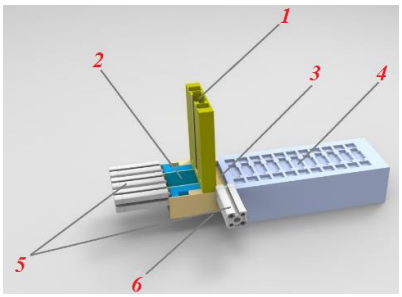


Figure 6. Mechanical design of the Warehouse, highlighting: 1) storage stacks; 2) actuators that push available parts from the stack into the assembly tray; 3) the assembly tray; 4) memory boxes with 11 slots per part type; 5) colour detection sensors; 6) pneumatic actuators for the assembly

The Wagons. The system has been restricted to having three transportation units (wagons), but each of them has the capacity of carrying one or two assembled sticks from the loading station (next to the Warehouse) to the Testing Station (close to the end user pick-up point). As described in [3], it is assumed that the simulated wagons cannot overtake each other and do not malfunction (e.g. fall off the tracks) in the current analysis. The mechanical design of a double-capacity wagon is shown in Figure 7.

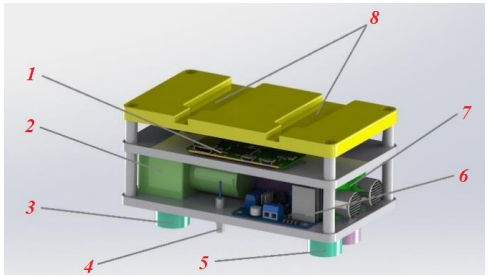


Figure 7. Mechanical design of a Wagon containing: 1) electronic board for control; 2) electronic motoreductor; 3) drive wheel; 4) sensor for station detection; 5) driven wheels; 6) motor driver; 7) ultrasonic distance detectors; 8) slots for transporting sticks.

3.3 Phases of development

There are two distinct phases of development, as envisioned in the methodology described in [3]: the digital model and the construction of the prototype and deployment (Table 1). For the first phase, the agent-oriented approach was best suited to provide the most adequate abstractions to design the conceptual model of the prototype by identifying its main subsystem types (i.e. production machines, order, and factory infrastructure) and define the interaction protocols among these subsystems. These types are well-established in agent-based manufacturing control system [28] and are now part of the more complex and abstract Reference Architecture Model Industry 4.0 [29]. While the first phase was intended to determine how to build the prototype, in the second phase the prototype was implemented in its final form. Therefore, each subsystem was developed in a specific language and tool, suitable to the domain and expertise of the team, by following the component-based approach to reach its concrete implementation. Table 2 shows the correspondence between the subsystems and the adequately suited tool for implementing a complete simulation, as well as the deployment devices considered.

Table 1. The general overview of development approach

Dev. phase	Stages	Goals
Digital model	*The requirements model*: a preliminary mechanical model of the production demo (domain description). It includes all the mechanical components to store, transport and assemble the USB-OTG sticks. The identification of the composite simulations from the production system, together with their roles and tasks, reflects the most important component types of the Reference Architecture Model Industry 4.0: production machines, order, and factory infrastructure.	• to identify the compositional structure of the targeted co-simulation and the best-suited model/simulation tool for each component from the production demo; • to facilitate a shared understanding among the specialized teams engaged in implementing the specific simulations.
	The homogenous co-simulation model: the high-level abstraction for the behaviour of each simulation, and the interactions among the composite simulations as described in the previous section. It includes distinct simulations for each component type: production (e.g. warehouse station, robot, transporting wagons, and testing station), orders (e.g. placed via mobile devices), and factory infrastructure (e.g. part tracker).	• to validate the interaction protocols among the composite simulations; • to have an early working co-simulation where the specific simulations may be gradually added, tested and validated; • to lessen the dependency among the dispersed teams involved in modelling the specific simulations; • to cover the left-over parts of the co-simulation that are not needed to be modelled at a high-level of details (e.g. the test station); • to identify conceptual (system-level) parameters that can be used at a later stage as stable constraints in the design space exploration for fine tuning.
Construction and deployment	*The heterogeneous co-simulation implementation model:* the detailed model of each simulation. It includes both continuous-time (CT) and discrete-event (DE) models in various simulation tools (Table 2, "technology" column).	• to simulate, test and validate from a holistic perspective and with an increased level of accuracy an entire system • to generate code from the specialized simulation tools of the different subsystems for specific hardware implementations
	The deployment model: the units modelled and tested by the heterogeneous co-simulation have been deployed in the demo stand for fine tuning under real-life conditions (Table 2, "deployment" column).	• to extend the libraries (e.g. 20-sim and 4DIAC with specific sensors and communication protocols) and functionalities (e.g. INTO-CPS, Overture with visualisation and code-generation capabilities) of the employed tools to cope with the real industrial needs

Table 2. The simulation technology and deployment infrastructure for each component

Type	Unit	Simulation Technology	Deployment Device
Orders	HMI	4DIAC + MQTT Overture (VDM-RT)	Smartphones and tablets
Infrastructure	Part Tracker	Overture (VDM-RT)	NVIDIA Tegra Jetson
Production	Warehouse	20-sim	Raspberry Pi with UniPi Expansion Board + Stäubli robot
Production	Wagons	4DIAC	Raspberry Pi controlling DC motors, position sensors and anti-collision ultrasonic sensors.
Production	Test Station	4DIAC	Camera for image processing and actuators connected to a Raspberry Pi
Overview	Unity	Unity animation	PC

The experimental results reported in this paper are related to the first phase of the development (the digital model), more specifically the homogenous co-simulation. It is important to emphasize that the VDM-RT models are not meant to be accurate in the physical (mechanical / electrical) implementation sense. The VDM-RT models do not need to have complete functionality for the homogeneous co-simulation, only the bare minimum from which the communication lines between units and the system-level parameters can be validated. The incompleteness of the VDM-RT models is related to details of the inner workings of the components, not necessarily respecting all the constraints of reality. Examples of incompleteness include randomly generating colours for USB-OTG parts in the warehouse or randomly considering the test successful or unsuccessful in the Test Station. Another example is breaking continuity: the physical Warehouse sequentially drops coloured USB-OTG parts from stacks into the assembly box and instructs the Robotic Arm to remove the parts if they do not fit the requested colour. When a cancellation occurs, the physical Warehouse must instruct the Robotic Arm to remove all parts from the assembly box (if they do not fit the next order) and then it can start dropping new parts from the stacks. The simulated Warehouse generates random colours for the USB-OTG parts and will immediately start generating new colours for the next order without instructing the Robotic Arm to remove existing parts from the assembly box in case they do not fit. Such aspects of the functionality are minor details with respect to the DE

modelling used for the abstract validation. The internal states however are all well-established, along with the communication patterns and lines between modules, such that the behaviour of the refined modules does not diverge substantially from the behaviour of the abstract models. Once established, the communication lines and the types of data they carry become hard constraints of the simulation that cannot be easily changed, but new lines of communication could be added if necessary. For instance, new communication lines have been added later in the development of the project, for transmitting (to the Part Tracker) the level of vibration recorded by each subsystem.

Another advantage of having only VDM-RT models as the first stage of development is the possibility of determining conceptual (system-level) parameters before the design space exploration at the heterogeneous co-simulation stage. The identified parameters of the production line can be grouped into two categories:

- Conceptual (system-level) parameters, which are independent of (or do not pose any significant problems for) the physical implementation, such as the number of colour choices available to the customers, the number of memory box slots in the Warehouse or the carrying capacity of the wagons. These parameters are suited for analysis during the homogeneous co-simulation stage.
- Physical (subsystem-level) parameters, which define the physical properties of the components inside subsystems, such as the parameters of the pneumatic piston used in the assembly tray of the Warehouse, mechanical and electrical parameters for the Robotic Arm and Wagons, sensitivity of the sensors used in various parts of the production line, etc. These parameters are suited for analysis during the heterogeneous co-simulation stage.

The conceptual parameters can be determined sufficiently accurately even with the incomplete simulation model. The number of colour choices available to the customers may influence the stocks of parts required for a good functioning of the assembly line but is not influencing the physical design in any way. The number of memory box slots and the carrying capacity of the wagons influence the design of those units, but without increasing the complexity of the mechanical design. Adding a few slots or a second indentation in the upper surface of the wagon (for a second USB-OTG stick), although becoming hard constraints for the heterogeneous simulation and implementation, poses no additional mechanical difficulty compared to not adding them and therefore does not impact the future stages of development. With these parameters determined beforehand, the design space exploration at the heterogeneous stage is constrained and focused on identifying the physical properties of the components that go into the mechanical, electrical and other designs of the subsystems.

4. Experimental setup

Having a system covering all the important stages of a production line, from orders to delivery, enabled us to set up two types of analysis scenarios:

- Analysis of the behaviour of the system when varying parameters external to the system (exogenous parameters)
- Analysis and optimization of the system when varying parameters internal to the system (endogenous parameters)

4.1 System setup

All simulated experiments have been run for 45000 time units, which correspond to 45000 seconds or 12.5 hours for the physical systems (although the simulation itself took about 6 hours for each experiment). The orders can be placed at any time within the initial 12 hours, while the remaining 30 minutes were added to allow for the completion of the late or postponed orders.

The track length is 100 units and the wagons report their positions every 10 seconds. There are three speed options available to the customer: low (corresponding to the default cruising speed of the wagons, 0.3 units of length per second), medium (0.7 units of length per second) and high (1.1 units of length per second). On the considered track, the Warehouse Waiting Room (loading station) is at position 10, and the Test Station is at position 80 of the total length of 100 units. When loaded, the wagons attempt to keep the requested speed as long as crashes are avoided, but on the return track (from the Test Station back to the Loading Station), they revert to the default speed, considered most energy-efficient.

The exogenous parameters refer to the expected number and distribution of orders. Using the Overture (VDM-RT) HMI allowed for a good control of both parameters. The orders have been generated from the Uniform and Normal (Gaussian) distributions, shown in Figure 8 (a). The Gaussian distribution is justified by the Central Limit Theorem when the demand comes from many different independent or weakly dependent customers, whereas the uniform distribution could be the result of orders coming from independent customers at various time zones (or evenly populated meridians).

The number of orders over a 12-hour period was selected between 200 and 300 orders, about half of which would be cancelled if the order does not reach the end user within 900 seconds (15 min) from the request. For the uniform distribution, the orders are expected to come every 144s (for 300 orders) to 216s (for 200 orders) on average. The actual sets of orders and their distribution, as generated with the Matlab random variable functions, are shown in Figure 8.

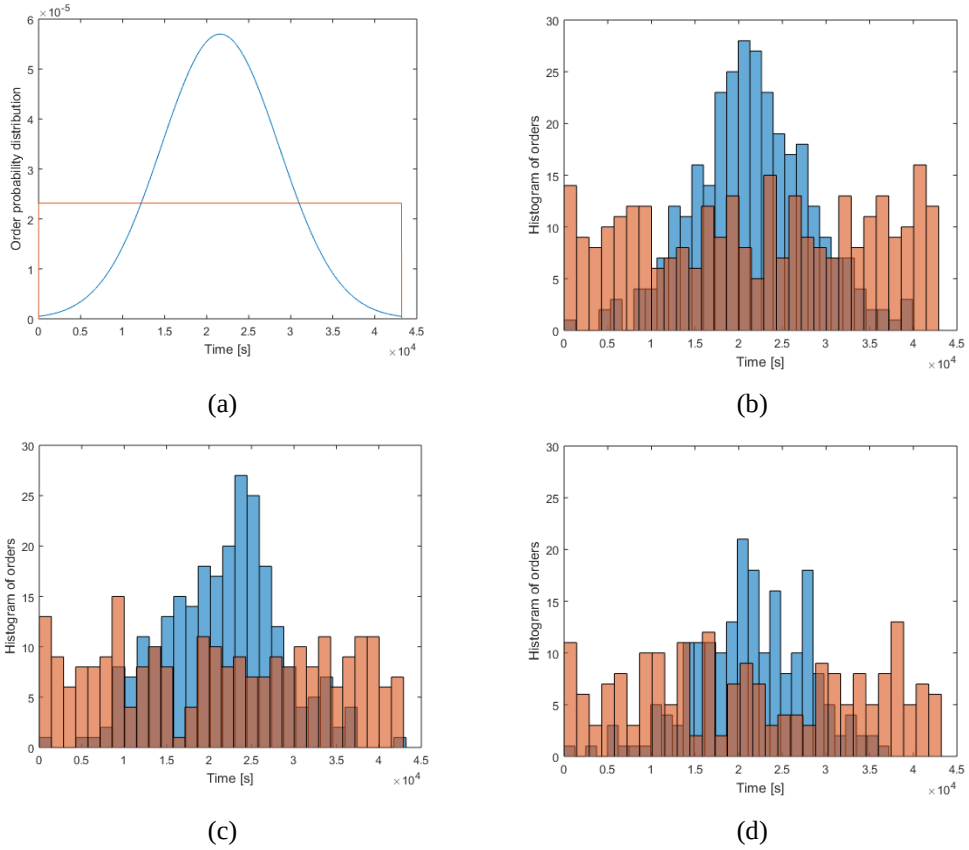


Figure 8. Distribution probability functions (a) and histogram of orders for (b) 300, (c) 250 and (d) 200 orders from the Gaussian (blue) and Uniform (orange) distributions.

Additionally, one order (an all-black stick with requested speed "low" and no cancellation) is inserted in the list at the start, to initialise the system. These sets of orders are used to test the load and overload of the production line. The other parameters of the orders are fixed:

- the colour choices are random from the discrete uniform distribution
- the speed options are also uniformly distributed (about a third of the orders contain each option)

The endogenous parameters considered in the current analysis are:

- the number of available colours
- the size of the Warehouse memory box
- the transport capability of wagons

The description and reasonable experimental range of these parameters for the homogeneous co-simulation is shown in Table 3. The best endogenous values chosen according to the experiments detailed in the next section are marked in bold font.

Table 3. Parameters analysed in the homogenous co-simulation design space exploration

Parameter Name	**Type**	**Description**	**Experimental values**
Number of orders	Exogenous	Number of orders received in an interval of 12 hours	201, 251, 301
Distribution of orders	Exogenous	The distribution of order times within the interval of 12 hours	Uniform, Gaussian
Colour choices	Endogenous	The number of colours available for the users to choose from, for each part of the USB-OTG stick	4, **6**, 8
Memory box size	Endogenous	The number of memory box slots in the Warehouse for each part type	10, 25, **40**, 50, 60
Wagons capacity	Endogenous	The number of slots available on the wagons for transporting assembled USB-OTG sticks to the Test Station	1, **2**

Available colours have been named generically based on the RGB interpretation of the binary representation of the colour numbers, as shown in Table 4. The number of available colours has been chosen between 4, 6 and 8, considering that fewer than 4 colours is not an interesting test, and that more than 8 colours is impossible without re-defining the concrete message encoding between subsystems. Having a maximum of 8 available colours (limited due to the structure of the messages between HMI and Part Tracker, as well as between Part Tracker and various other units such as the Warehouse and the Test Station), the lowering of the colour set may have a positive impact on the time an order spends in the Warehouse before being assembled. The size of the Warehouse memory box could also be adjusted between 10 slots for each part type up to 60 slots for each part type. Unlike the colours, the memory box size is internal to the Warehouse and can be adjusted freely. However, large memory boxes may be impractical (may use a lot of space) or undesirable by the producer.

Table 4. Generic colour names based on the binary representation of their number

Colour Number	**Binary representation**	**Generic Colour Name**
0	0 0 0	Black
1	0 0 1	Blue
2	0 1 0	Green
3	0 1 1	Cyan
4	1 0 0	Red
5	1 0 1	Magenta
6	1 1 0	Yellow
7	1 1 1	White

The transport capability of the wagons refers to the number of units that can be loaded on a wagon. The capacity of each wagon can be maximum two sticks at a time. Of course, wagons can still carry a single stick, if a second one is not readily available in the Waiting Room next to the Warehouse. The testing time of the sticks in the Test Station is fixed to 16s per unit and the test result has a theoretical 2% probability of rejection, in which case the order returns to the Warehouse for re-assembly.

4.2 Cost functions

The improvements of the setups have been quantified by the following cost functions which would ideally all be at a minimum at the same time.

- CF0: Percentage of cancelled orders
- CF1: Average Lead Time,
- CF2: Number of discarded parts

The percentage of cancelled orders is the most important cost function from a practical point of view in a real production line, since cancelled orders become unsold items. Only about half of the orders have cancellation policies (if the order is not received in 900s from submitting the request). Lead time is considered the length of time between the order placement and the moment the system delivers the requested unit to the end user. The states that a stick goes through before reaching the user are *<ORDERED>*, *<WAREHOUSE>*, *<WAITING_ROOM>*, *<WAGON>*, *<TEST_STATION>*, *<END_USER>*, as shown in Figure 9. Besides the lead time, the log files allow the computation of times spent in each state. If the test station rejects a unit, the order returns in *<ORDERED>* state and the lead time takes into account both assembly runs.

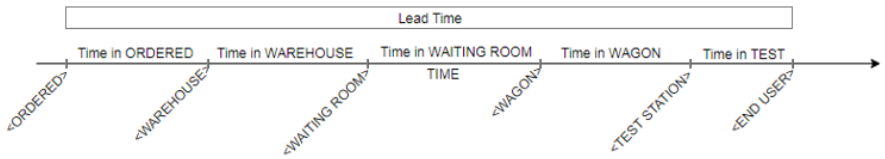


Figure 9. Lead time and its constituent intervals based on the time of reaching each stage

The number of parts discarded for recycling is an important indicator of both the Warehouse lead time and the memory box overload. While these measures have been computed separately, keeping the number of discarded parts low improves both.

Considering the transportation from the Warehouse to the test station, the user has the option of requesting the speed of the wagons (low, medium or high), which the system does its best to comply to. The average speed difference is computed as in equation 1, where the mediation is performed over all transported units (excluding cancelled orders that did not reach the wagons); v_{real} is the speed of the transport between the loading and testing station, computed based on the track length and arrival times; v_{req} is the requested speed:

$$v_{av} = mean(v_{real} - v_{req}) \quad (1)$$

4.3 Data extracted from the log files

Each experiment is identified by the five parameters of Table 3. The important data extracted from each experiment falls under one of the following categories:

- General experiment data, including the experiment inputs, number of cancelled and successfully delivered orders, distribution of order parameters, etc.
- Time-related data, including minimum, maximum and average lead time and time spent in each state, both in numerical and graphical form.
- Warehouse data, including the Warehouse memory box capacity and average load, number of discarded pieces (overall, per colour, per side, etc.), lucky hits, both in numerical and graphical form.
- Wagon data, including the capacity, average speed difference (compared to the requested speed), both in numerical and graphical form.

All this data will be shown comparatively between experiments to highlight some behaviours or improvements of the system.

5. Experimental Results

Even with the small number of system level parameters and choices for each parameter as described in Table 8 (two exogenous: number and distribution of order, and three endogenous: number of colour choices, size of memory boxes and wagon capacity), the number of possible combinations is large (180). Therefore, a series of experiments has been devised to arrive at the optimal parameters without running all combinations.

The first experiments focused on varying the exogenous parameters to identify when the system is overloaded (or close to being overloaded), while the second set of experiments aimed to improve the cost functions by sequentially improving the number of memory box slots, wagon carrying capacity and number of available colours.

5.1 Analysis of the behaviour of the system when varying exogenous parameters

For the first round of experiments, the size of the memory box (10/25/40 slots) and the external parameters (order distribution: uniform/Gaussian, and number of orders: 201/251/301) are varied to generate nine experiments per distribution. During these experiments, the focus is on the external parameters, thus the memory box size is not varied to the full extent. The uniform distribution experiments do not produce cancelled orders, and the reason is revealed when analysing the histogram of order times in Figure 8. Even with the highest number of orders tried, the Uniform distribution rarely goes over 14 orders in 1500s (25 minutes), whereas the Gaussian distributions concentrate at least 20 orders in the most loaded 25-minute timeslot, preceded and followed by other crowded timeslots.

The number of orders over the whole 12-hour experiment is therefore less indicative of the number of cancellations than the load in the peak timeslots. The system can handle constant flows of 10-15 orders per 25-minute timeslot, but when one timeslot goes over 20 orders, the system is overloaded, and the delays pile up and negatively affect following orders. Figure 10 shows the relation between the number of orders and the percentage of cancelled orders in the case of the Gaussian distribution. The size of the Warehouse memory box seems to have limited effect on this cost function.

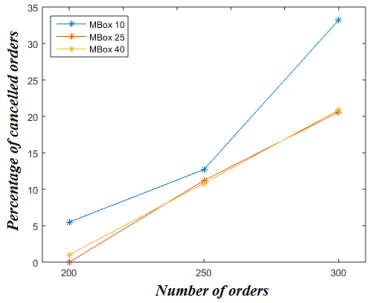
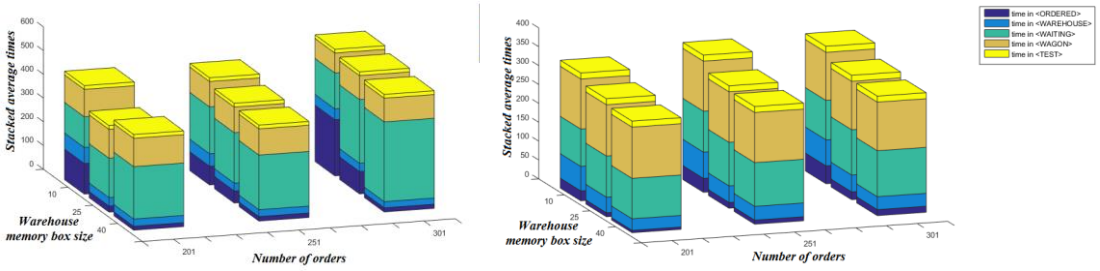


Figure 10. Percentage of cancelled orders in the Gaussian distribution experiments

Referring to the lead time cost function, all experiments seem to have similar minimum lead times (around 100s) stemming from the first orders received with high speed request, very close to the theoretical minimum. Figure 11 show the average time the components spend at each stage in graphical form. Memory box sizes have a limited influence on the average time spent in the <*ORDERED*> and <*WAREHOUSE*> states.

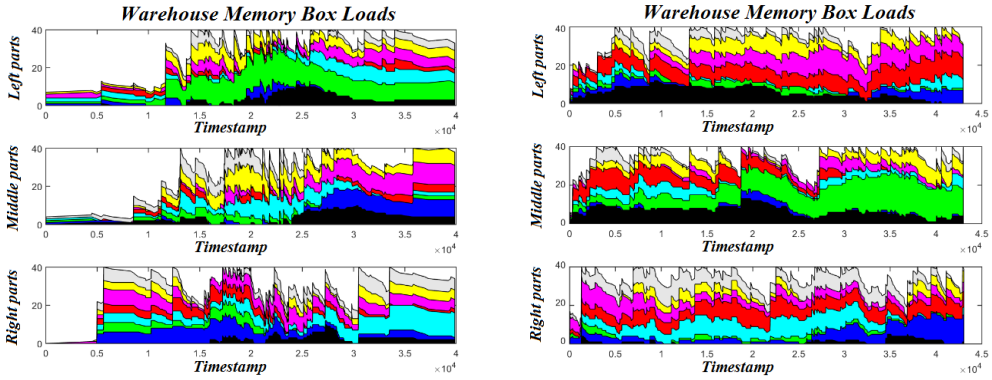


(a) Gaussian distribution experiments (b) Uniform distribution experiments

Figure 11. Stacked average times spent by sticks at each stage of production

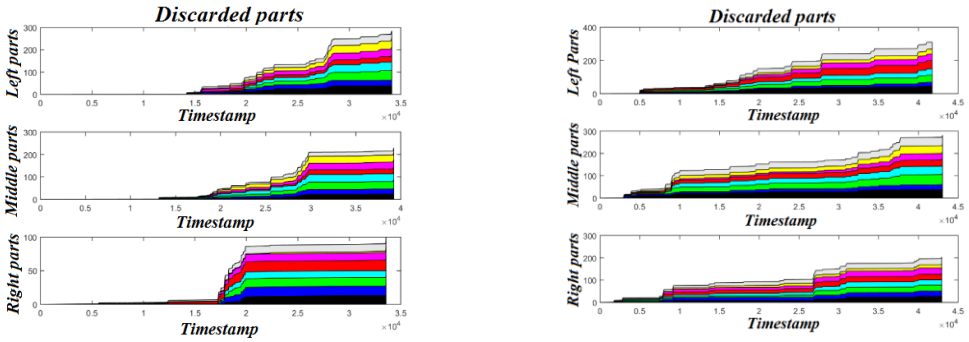
A more detailed insight into what is happening in the Warehouse can be gained by looking at the memory box load and the number of parts discarded for recycling because of a lack of memory space. Although all experiments have this information, Figure 12 shows the evolution of the memory loads over time and discarded parts only for experiments with Uniform and Gaussian distribution of 301 orders, eight available colours, 40 memory slots and single carrying capacity of the wagons as illustrative examples for the conclusions. The colours mark actual coloured parts, to express not only the load itself, but also the distribution of colours for parts in the memory boxes at all times. Figure 13 shows the evolution of discarded parts by colour in the respective colours, as well as the overall evolution as stacked area. Even at 40 memory slots, the Warehouse box is filled rather

quickly once enough orders have been received. In the case of the Uniform distribution, this happens earlier than for the Gaussian distribution. The uniform distribution generates a more linear graph for the discarded parts because of the more evenly received orders, whereas the Gaussian distribution exhibits a sharper increase in discarded parts once more orders start coming in the middle of the time interval. The number of discarded parts is comparable between the two distributions and is related only to the overall number of orders and memory box size, as shown in Table 5.



(a) Gaussian distribution experiment (b) Uniform distribution experiment

Figure 12. Left / Middle / Right memory box loads



(a) Gaussian distribution experiment (b) Uniform distribution experiment

Figure 13. Evolution of left / middle / right parts discarded for recycling

Table 5. Number of discarded parts and average speed difference

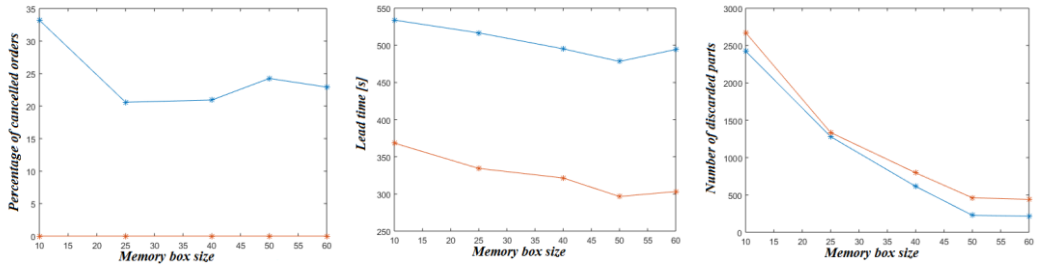
Experiment Parameters		**Number of discarded parts**		**Average speed difference**	
Number Orders	*Memory Slots*	*Uniform Distribution*	*Gaussian Distribution*	*Uniform Distribution*	*Gaussian Distribution*
201	10	1601	1834	-0.0677	-0.0736
201	25	772	569	-0.0732	-0.0679
201	40	362	396	-0.0740	-0.0664
251	10	2222	1781	-0.0701	-0.0623
251	25	1107	945	-0.0692	-0.0530
251	40	720	560	-0.0667	-0.0669
301	10	2674	2426	-0.0670	-0.0576
301	25	1336	1281	-0.0666	-0.0770
301	40	797	615	-0.0706	-0.0714

The average speed difference is computed between the real and requested speed of the wagons on the part of the track between the loading station (next to the Warehouse) and the test station. Not surprisingly, as shown in Table 5, the speed difference is quite close to zero (the Wagon runs with an average speed close to the requested speed, albeit somewhat slower) and is not varying significantly either with the distribution or with the Warehouse parameters. Another observation is that the number of pieces transported by each wagon is close to uniformity in all experiments. For instance, the uniform distribution experiment has wagon A transport 34.5% of the sticks, wagon B transport 31.3% of the sticks and wagon C transport 34.2% of the sticks and the Gaussian distribution experiment has wagon A transport 33.7% of the sticks, wagon B transport 33.3% of the sticks and wagon C transport 33.0% of the sticks. Wagon usage is therefore balanced. Since all wagons work equally hard, no imbalance has yet been created due to the number of wagons. Although not simulated, using three equally hard-working wagons certainly improves the lead time over one or two wagons, which suggests that the minimum number of wagons chosen for the production line should be three.

The conclusion of these experiments can be summarized as follows: the number of orders is less important than the peak load, the size of the memory box has limited effect on the cancellation percentage, the number of discarded parts is related only to the overall number of orders and memory box size, and wagon usage is balanced. This demonstrates the benefit of the homogeneous co-simulation. The next set of experiments is focused on investigating the effect of the number of memory box slots on the discarded parts.

5.2 Analysis of the system when varying the number of memory box slots

The adjustment of the number of memory box slots makes more sense for the experiments when the system is close to be overloaded by orders. The experiments shown in Figure 14 focus on the case that 301 orders are made, by varying the size of memory box and observing the number of cancelled orders, the lead time and the number of discarded parts.



(a) (b) (c)

Figure 14. Evolution of (a) the percentage of cancelled orders, (b) the lead time in seconds and (c) the number of discarded parts as a function of the memory box size, for 301 orders from the Gaussian (blue) and Uniform (orange) distributions

The most important effect of increasing the memory box size is the reduction of discarded parts (due to the memory box being large enough to usually hold enough samples of all colours). However, even this effect is significantly diminishing for memory boxes with more than 50 slots. The lead time is decreasing due to the lower time spent in or before the Warehouse. But because the Warehouse time is less significant than the waiting time after this stage (due to wagons being elsewhere on the track), the decrease in lead time is rather mild. The number of cancelled orders is not affected as much by this setting, as all memory box sizes over 10 reduce the cancelled order percentage by about 10%.
The conclusion for this set of experiments is therefore that the only significant cost function decrease is the number of discarded parts, but even this improvement flattens for memory box sizes larger than 50.

5.3 Analysis of the system when doubling the load capacity of the wagons

As seen in Figure 11, the most consuming time stage is the waiting for the wagons in the Waiting Room next to the Warehouse. Table 6 compares the average lead time and the average time spent in waiting for some experiments with 8 colours available and a memory box size of 40 or 50, as well as the percentage of cancelled orders. In the table, *Single* indicates that wagons can carry a single stick at a time, while *Double* indicates the doubling of the capacity.

Table 6. Comparison of Single and Double wagon capacity results

Wagon capacity:	*Single*	*Double*	*Single*	*Double*	*Single*	*Double*	*Single*	*Double*
	Percentage of cancellations		**Average lead time [s]**		**Average waiting before Warehouse**		**Average Waiting Room time [s]**	
Gaussian; 40 slots	20.9%	0.7 %	495.26	379.07	17.42	18.05	335.77	96.70
Gaussian; 50 slots	24.3%	0.0 %	478.31	330.07	6.45	14.60	343.39	116.31
Uniform; 40 slots	0.0 %	0.0 %	321.46	286.49	15.87	9.24	121.70	84.75
Uniform; 50 slots	0.0 %	0.0 %	296.57	288.56	7.75	10.21	113.38	88.25

The doubling of the wagon capacity reduces the lead time primarily by reducing the waiting time after the Warehouse (to approximately a third) and therefore the average lead time is reduced significantly enough for eliminating almost all cancellations. Doubling the carrying capacity has no drawback and it can allow for lower sizes of the memory box to achieve a low number of discarded parts, as will be tested in the next set of experiments.

5.4 Analysis of the system when varying the number of colours available to the user

Reducing the number of colours should have a beneficial effect on the lead time by increasing the probability that a requested part is available in the memory boxes. The effect of reducing the number of colours will be analysed in conjunction with the dimension of the memory box, since both operations aim to improve the Warehouse time and reduce the number of discarded parts. The point of interest for this set of analyses is the balance between the two parameters, in the sense of identifying the amount by which the memory box can be downsized when reducing the number of available colours.

Table 7 shows the lead time when both parameters (number of colours available to the user and size of the memory box) are varied. The experiments use 301 orders, being the most challenging attempts in their respective distributions, and double wagon capacity, since it improves the cost functions with no drawback.

Table 7. Average lead time [s] and number of discarded parts for various combinations of the number of available colours and the size of the memory box

Uniform Distribution		**Size of the memory box**			
		50 slots	*40 slots*	*25 slots*	*10 slots*
Number of available colours	*8*	288.56s 691 parts	286.49s 625 parts	310.75s 1450 parts	360.56s 2696 parts
	6	274.35s 216 parts	273.75s 325 parts	284.04s 758 parts	300.60s 1536 parts
	4	260.31s 102 parts	266.11s 103 parts	263.60s 149 parts	289.67s 600 parts

Gaussian Distribution		**Size of the memory box**			
		50 slots	*40 slots*	*25 slots*	*10 slots*
Number of available colours	*8*	330.07s 416 parts	379.07s 664 parts	387.12s 889 parts	450.46s 2065 parts
	6	315.38s 171 parts	356.74s 180 parts	313.03s 544 parts	417.53s 1340 parts
	4	293.20s 20 parts	332.17s 129 parts	309.54s 230 parts	319.22s 475 parts

For almost all experiments (except Gaussian distribution with a memory box of 10 slots), at most 2 orders were not fulfilled, making the percentage of cancellations reasonable (under 1%). Regarding the lead time, it is not significantly decreasing with a memory box of 25 slots or more, regardless of the number of colours. The number of discarded parts does not decrease as sharply starting with 40 slots in the memory box, while still being higher for larger numbers of available colours. Therefore, a good balance is achieved when using 6 colours and 40 slots. The average time the order spends at each stage for these experiments is shown in Figure 15.

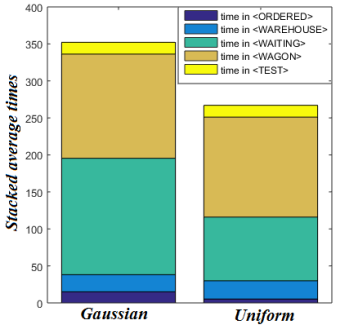


Figure 15. Stacked average times at each stage of production for the considered set of parameters

Table 8. Additional information about the experiments for the considered set of parameters

	General Information about the experiment	Cost functions
Uniform distribution, 301 orders, 6 colours, 40 memory slots, double carrying capacity	Requested colours for the Left part: Black 47 (15.6%), Blue 59 (19.6%), Green 62 (20.6%), Cyan 48 (15.9%), Red 40 (13.3%), Magenta 45 (15.0%), Yellow 0 (0.0%), White 0 (0.0%) Requested colours for the Middle part: Black 57 (18.9%), Blue 54 (17.9%), Green 45 (15.0%), Cyan 39 (13.0%), Red 52 (17.3%), Magenta 54 (17.9%), Yellow 0 (0.0%), White 0 (0.0%) Requested colours for the Right part: Black 58 (19.3%), Blue 43 (14.3%), Green 55 (18.3%), Cyan 51 (16.9%), Red 47 (15.6%), Magenta 47 (15.6%), Yellow 0 (0.0%), White 0 (0.0%) Requested speeds: Low 90 (29.9%), Medium 114 (37.9%), High 97 (32.2%) Number of pieces rejected at test station: 7 (2.33%) Warehouse Memory Box average load: 73.4% L 75.3%, M 80.7%, R 64.1% Warehouse Number of discarded pieces: 325 L 177, M 124, R 24 Warehouse Number lucky hit pieces: 241 L 89, M 86, R 66	CF0: cancelled orders 0 (0.0%) CF1: lead time Average: 273.75 s Minimum: 102.00 s Maximum: 834.00 s CF2: discarded pieces 325 pieces
Gaussian distrib., 301 orders, 6 colours, 40 memory slots, double carrying capacity	Requested colours for the Left part: Black 44 (14.6%), Blue 66 (21.9%), Green 45 (15.0%), Cyan 41 (13.6%), Red 54 (17.9%), Magenta 51 (16.9%), Yellow 0 (0.0%), White 0 (0.0%) Requested colours for the Middle part: Black 49 (16.3%), Blue 52 (17.3%), Green 49 (16.3%), Cyan 50 (16.6%), Red 54 (17.9%), Magenta 47 (15.6%), Yellow 0 (0.0%), White 0 (0.0%) Requested colours for the Right part: Black 52 (17.3%), Blue 56 (18.6%), Green 47 (15.6%), Cyan 43 (14.3%), Red 50 (16.6%), Magenta 53 (17.6%), Yellow 0 (0.0%), White 0 (0.0%) Requested speeds: Low 100 (33.2%), Medium 94 (31.2%), High 107 (35.5%) Number of pieces rejected at test station: 7 (2.33%) Warehouse Memory Box average load: 64.6% L 55.5%, M 72.9%, R 65.5% Warehouse Number of discarded pieces: 180 L 17, M 97, R 66 Warehouse Number lucky hit pieces: 199 L 68, M 58, R 73	CF0: cancelled orders 0 (0.0%) CF1: lead time Average: 356.74 s Minimum: 104.00 s Maximum: 830.00 s CF2: discarded pieces 180 pieces

Additional information given in Table 8 includes:

- the number and percentage of requested colours, almost uniformly distributed;
- requested speeds, about a third of the orders requesting each available speed;
- the number and percentage of pieces rejected at the test station;
- memory box average load, as well as the average loads for the boxes of each part;
- number of discarded pieces and lucky hits (totals as well as per part type);
- number and percentage of cancelled orders (0 for both experiments);
- minimum, maximum and average lead time.

Such information has been recorded for each experiment and the relevant data has been presented directly or indirectly in graphical form throughout Section 5. However, Table 8 provides full statistics for the experiments with 301 orders (with Gaussian and uniform distribution over time) and the chosen production line parameters (6 available colours, 40 memory slots, wagons with double carrying capacity).

6. Discussion

From the methodological standpoint the case study provides several insights from employing the model-based approaches to build complex CPS-based manufacturing systems. Complex co-simulations for production systems requires to rely on more mature methodologies with relevant track-records in the domain, such as agent-oriented and component-based. Even if both methodologies (agent-oriented and component-based) promote the classical features of agile software development (iterative, incremental, lightweight and collaborative) they use different abstractions that make them suitable to describe either the conceptual or the implementation model. While the agent-oriented approach provides the most adequate abstractions to design the conceptual model of the co-simulation, some of its specifics make it difficult to implement in the available co-simulation technology which is more related to component-based approaches. In our case study we have used the agent-oriented approach to identify the subsystems and to structure the messages among the specific co-simulation units, whilst its implementation is following the component-based approach. This was needed for at least two reasons:

- The components' meta-model does not have abstractions for the goals and components can only use task delegation. In manufacturing systems there are multiple conflicting objectives at both: system level (e.g. throughput, lead time, work-in-progress, etc.) and individual level (e.g. energy consumption, degree of utilization, slack time, etc.). The right balance among all these conflicting objectives are unknown/underspecified and are usually discovered during the design space exploration phase. Any predefined way of achieving a goal (task delegation) may inhibit optimizations. While not tested in our case study, the possibility to allow internal optimization is available in subsystems, for instance in the warehouse or the test station. The warehouse receives information about the requested colours of a USB-OTG stick, but it is not forced to look for the necessary parts in any specific order. The VDM-RT implementation looks for parts in the order Left Cap – Middle Part – Right Cap, sequentially, because of the bottleneck of having a single Robotic Arm capable of moving parts to and from the memory boxes. The usage of two Robotic Arms, for example, would introduce an optimization problem where the Warehouse would have to decide which side to start from. For the test station as well, in the case of the double carrying capacity of the wagons, the messages from the Part Tracker indicate the two orders available for testing at that instance, but not the order of testing them. Consequently, goal delegation may be implemented in the current co-simulation technology.

- For agent-oriented approaches, the environment is a first-class abstraction that is a structural part of an agent's meta-model, while it is not part of a component's meta-model. In manufacturing the infrastructure is part of any referenced architecture and therefore a special attention has been given to model and simulate the Part Tracker. The interaction with the environment is different, agents can measure the environment, while components can react to an event only by defining a relation with it. Due to the initial requirements in our case study we have not tested the possibility to react to state changes from the Part Tracker, but it would be feasible to implement with the current co-simulation technology. For example, an extension/improvement would be, in the case of multiple Warehouse units, to allow the wagons to take the decisions themselves whether to wait longer in the loading station of a particular Warehouse (even if there are no sticks currently in the corresponding waiting room) or to leave the loading station, for another loading station (for instance in case another wagon is close enough to the current loading station). In this scenario, the wagon would acquire by itself the information about the existence (or not) of ready USB-OTG sticks in the waiting rooms (e.g. by enquiring the Part Tracker) and would have to be aware of the loading process, would have to decide on which slot to place the sticks and would know when the loading is complete (and the sticks are firmly placed in the wagon slots).

There are some recognizable benefits in using the two-phase methodology of the development as described in this paper, compared to the current state of technology. The most straightforward benefit is the possibility to simulate, test and validate (from a holistic perspective and with an increased level of accuracy) an entire production system that needs cross-functional expertise; in the past, these issues were tackled by experiential learning from multiple sequential implementations of the automation component; in other words, *the* co-simulation facilitates the adoption of agile software development principles for factory automation. For example, the initial development of a homogeneous co-simulation in VDM-RT for the iPP4CPPS prototype was particularly useful in driving cooperation and clarifying the assumptions of the teams involved in modelling specific components. Once the VDM-RT co-simulation was running, the independent developments of units were integrated, validated and deployed when ready, since the methodology allowed for their development in any order. Another benefit is the ability to handle unpredictable events (to a certain extent). The use of co-simulations when designing an automated production system avoids the build-up inertia of subsequent design constraints, facilitating the low and late commitment for these decisions, like the specific microcontrollers or PLCs, as well as allowing the change of modelling tools for any subsystem at any time during the project.

7. Concluding Remarks

The paper presented an experience report on employing the collaborative co-simulation to design a complex CPS-based production system, focusing on the early-stage homogeneous co-simulation. Each subsystem was first modelled abstractly in VDM-RT, using the Overture tool. The goal of this homogeneous co-simulation was twofold: to identify the right interaction protocols (signals) among the various components (stations) of the prototype and to identify conceptual (system-level) parameters before the design space exploration at the heterogeneous co-simulation step.

The experiments were performed under the assumptions that the orders are received in a Gaussian or Uniform distribution over an interval of 12.5 hours, with balanced distributions of colours for each side of the stick and balanced requested wagons speeds. Two types of analysis have been performed:

- Analysis of the behaviour of the system when varying exogenous parameters, to identify the maximum load that the system is capable of handling.
- Analysis and optimization of the system when varying endogenous parameters, to find a balanced set of parameters.

The analysis on exogenous parameters revealed that, for eight available colours, reasonably sized Warehouse memory boxes and single-unit capacity of the wagons, the system can respond adequately for up to 20 orders per interval of 25 minutes. Doubling the wagon carrying capacity (i.e. having two slots instead of one, for transporting sticks) improves the system response also for intervals with a little over 25 orders per interval of 25 minutes.

The analysis on internal parameters allowed the balanced decision of making only six colours available to the users and having a Warehouse memory box with 40 slots per side, thus decreasing the number of parts discarded to recycling while at the same time keeping a good memory box load level. These parameters, when taken into account, reduce by a considerable amount the design space exploration at the heterogeneous stage, in effect becoming system constraints for the heterogeneous simulation and the physical model.

Acknowledgements

This work is supported through the DiFiCIL project (contract no. 69/08.09.2016, ID P_37_771, web: http://dificil.grants.ulbsibiu.ro) co-funded by ERDF through the Competitiveness Operational Programme 2014-2020, iPP4CPPS project (Horizon 2020, grant agreement no.644400, experiment no. 16-UK-GERS-01) and Lucian Blaga University of Sibiu research grants LBUS-IRG-2018-04.

A special thanks goes to the other members of the iPP4CPPS project who have advanced the production line beyond the homogeneous co-simulation, including P.G. Larsen, K. Lausdahl, C. Thule (Aarhus University); V. Ruxandu, O. Savencu, R. Simedru, M. Neamtiu (Continental Automotive Systems Sibiu); C. Kleijn (Controllab); J. Cabral, H. Pfeifer (Fortiss GmbH.); J. Fitzgerald, C. Gamble (Newcastle University); B. Pirvu, A. Butean, S. Puscasu, R. Voju, D. Halati (Lucian Blaga University of Sibiu).

We would also like to express our gratitude to the anonymous reviewers for their positive comments, as well as for their constructive criticism based on which we have endeavoured to improve the writing and clarity of the paper.

REFERENCES

1. iPP4CPPS, Integrated product-production co-simulation for cyber-physical production system, accessed on September 2018 at http://centers.ulbsibiu.ro/incon/index.php/ipp4cpps/

2. Nof S.Y. (Ed.). **Springer Handbook of Automation**. Springer-Verlag, 2009. ISBN: 978-3-540-78831-7

3. Neghina M., Zamfirescu C.B., Larsen P.G., Lausdahl K., Pierce K., **Multi-Paradigm Discrete-Event Modelling and Co-simulation of Cyber-Physical Systems**, *Studies in Informatics and Control*, ISSN 1220-1766, vol. 27(1), pp. 33-42, 2018.

4. Gomes C., Thule C., Broman D., Larsen PG., Vangheluwe H.: **Co-simulation: A Survey**, In Sahni S. (Ed.): *ACM Computing Surveys (CSUR)*, vol.51(3), art. no. 49, 2018

5. Pierce K., Wolff S., Verhoef M., **Methods for Creating Co-models of Embedded Systems,** In *Collaborative Design for Embedded Systems*, Fitzgerald, J., Larsen, P.G., Verhoef, M. (Eds.), p. 153-183 (2014), Springer Verlag, ISBN 978-3-642-54118-6

6. Blochwitz T.; Otter M.; Åkesson J.; Arnold M.; Clauss C.; Elmqvist H.; Friedrich M.; Junghanns A.; Mauss J., Neumerkel D.; Olsson H.; Viel A: **Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models**, In: 9th International Modelica Conference, p.173-184, Munchen (2012)

7. Larsen P.G., Fitzgerald J., Woodcock J., Gamble C., Payne R., Pierce K.: **Features of Integrated Model-based Co-modelling and Co-simulation Technology**, *CoSim-CPS workshop organised in connection with the SEFM*, Trento, Italy, September 2017.

8. Larsen P.G., Battle N., Ferreira M., Fitzgerald J., Lausdahl K., Verhoef M.: **The Overture Initiative – Integrating Tools for VDM**. *SIGSOFT Softw. Eng. Notes* 35(1), 1–6 (2010)

9. Fitzgerald, J., Gamble, C., Larsen, P.G., Pierce, K., Woodcock, J.: **Cyber-Physical Systems design: Formal Foundations, Methods and Integrated Tool Chains**. In: *FormaliSE: FME Workshop on Formal Methods in Software Engineering*. ICSE 2015, Florence, Italy (May 2015)

10. Larsen P.G., Fitzgerald J., Woodcock J., Fritzson P., Brauer J., Kleijn C., Lecomte T., Pfeil M., Green O., Basagiannis S., Sadovykh A.: **Integrated Tool Chain for Model-based Design of Cyber-Physical Systems: The INTO-CPS Project**. In: *CPS Data Workshop*. Vienna, (2016)

11. Larsen P.G., Fitzgerald J., Woodcock J., Lecomte T.: **Trustworthy Cyber-Physical Systems Engineering, Chapter 8: Collaborative Modelling and Simulation for Cyber-Physical Systems**. Chapman and Hall/CRC (September 2016), iSBN 9781498742450

12. Larsen P.G., Fitzgerald J., Woodcock J., Nilsson R., Gamble C., Foster S.: **Towards Semantically Integrated Models and Tools for Cyber-Physical Systems Design**, pp. 171–186. *Springer International Publishing*, Cham (2016)

13. Fitzgerald J., Gamble C., Payne R., Larsen P.G., Basagiannis S., Mady A.E.D.: **Collaborative Model-based Systems Engineering for Cyber-Physical Systems – a Case Study in Building Automation**. In: *INCOSE 2016*. Edinburgh, Scotland (2016)

14. INTO-CPS Maestro, accessed on September 2018, available at https://github.com/INTO-CPS-Association/maestro

15. Modelio, accessed on September 2018, available at https://www.modelio.org/

16. Quadri I., Bagnato A., Brosse E., Sadovykh A.: **Modeling Methodologies for CyberPhysical Systems: Research Field Study on Inherent and Future Challenges**. *Ada User Journal* 36(4), 246–253 (2015)

17. Kleijn C.: **Modelling and Simulation of Fluid Power Systems with 20-sim**. *Intl. Journal of Fluid Power* 7(3) (2006)

18. OpenModelica, accessed on September 2018, available at https://openmodelica.org/

19. Strasser T., Rooker M., Ebenhofer G., Zoitl A., Sunder C., Valentini A., Martel A.: **Framework for distributed industrial automation and control (4diac)**. In: 2008 *6th IEEE International Conference on Industrial Informatics*. pp. 283–288 (2008)

20. Catia, accessed on September 2018, available at https://www.3ds.com/products-services/catia/

21. Fitzgerald J., Larsen PG., Verhoef M. (Eds.): **Collaborative Design for Embedded Systems –** *Co-modelling and Co-simulation*. Springer (2014) ISBN 978-3-642-54117-9

22. Bjørner D.; Cliff B.J.: **The Vienna Development Method: The Meta-Language**, Lecture Notes in CS 61. Berlin, Heidelberg, New York: Springer. (1978) ISBN 978-0-387-08766-5.

23. Fitzgerald J., Larsen P.G.: **Modelling Systems: Practical Tools and Techniques in Software Engineering**. Cambridge University Press, (1998) ISBN 0-521-62348-0

24. Verhoef M., Larsen P.G., Hooman J.: **Modeling and Validating Distributed Embedded RealTime Systems with VDM++**. In: Misra, J., Nipkow, T., Sekerinski, E. (eds.) *FM 2006: Formal Methods*. pp. 147–162. Lecture Notes in CS 4085, Springer-Verlag (2006)

25. Mühlhäuser M.: **Smart Products: An Introduction**, in *Constructing Ambient Intelligence*, Mühlhäuser M., Ferscha A., Aitenbich E. (Eds.) Springer Berlin Heidelberg, pp. 158-164 (2008)

26. Zamfirescu, C.B., Parvu, B.C., Schlick, J., Zühlke, D.: **Preliminary Insides for an Anthropocentric Cyber-physical Reference Architecture of the Smart Factory,** *Studies in Informatics and Control*, vol. 22 (3), pp. 269-278, 2013

27. Hermann M., Pentek T., Otto B.: **Design Principles for Industrie 4.0 Scenarios**. In: 2016 *49th Hawaii International Conference on System Sciences* (HICSS). pp. 3928–3937

28. Leitão P., Karnouskos S. (Eds.).: **Industrial Agents**. ***Emerging Applications of Software Agents in Industry***. *Elsevier, 2015*

29. **Referenzarchitekturmodell Industrie 4.0** (RAMI4.0), DIN SPEC 91345:2016-04, available at https://www.plattform-i40.de/I40/Redaktion/DE/Downloads/Publikation/din-spec-rami40.html