# Cyber-Physical Production Systems to Monitor the Polishing Process of Cutlery Production

José Ferreira
*Centre of Technology and Systems, CTS UNINOVA*
Caparica, Portugal
japf@uninova.pt

Fábio Lopes
*Centre of Technology and Systems, CTS UNINOVA*
Caparica, Portugal
fl@uninova.pt

Sudeep Ghimire
*Centre of Technology and Systems, CTS UNINOVA*
Caparica, Portugal
sud@uninova.pt

Guy Doumeingts
*INTEROP-VLab*
Brussels, Belgium
guy.doumeingts@interop-vlab.eu

Carlos Agostinho
*Centre of Technology and Systems, CTS UNINOVA*
Caparica, Portugal
ca@uninova.pt

Ricardo Jardim-Goncalves
*Centre of Technology and Systems, CTS UNINOVA*
Caparica, Portugal
rg@uninova.pt

***Abstract*—As technology evolves, the contemporary technological paradigm of manufacturing SME's has been changing and gaining traction to accommodate their ever-growing needs for adaptation to modern industrial processes and standards. Their willingness to integrate Cyber-Physical Systems (CPS) modules in their manufacturing processes and to implement Cyber-Physical Production Systems (CPPS) is firmly based on the perception that value added services result from the technological evolution and, in the future, better tools are expected to guarantee process control, surveillance and maintenance. These Intelligent Systems involve transdisciplinary approaches to guarantee interaction and behavioral fluidity between hardware and software components which often leads to complexity in the coordination of these components and processes. The main objective of this work is to study these aspects and to contribute with data regarding the applicability of CPS components in the current SME's environment with the intent of improving the performance of their manufacturing processes. To accomplish this, an architecture is proposed, which is based in the process modelling and process simulation of the different stages of an existing SME factory production and allows real-time information, through IoT data collection, to feed different mechanisms of production improvement such as planning, scheduling and monitoring.**

***Keywords—Process Modelling, Process Simulation, Cyber-Physical Systems, Data Acquisition, Sensors, Interoperability***

## I. INTRODUCTION

The term cyber-physical systems (CPS) refers to a new generation of intelligent systems with integrated computational, networking and physical capabilities [1] that aims to fill the gap between the cyber world, where data is exchanged and transformed, and the physical world in which we live.

The ability to interact with, and expand the capabilities of, the physical world through computation, communication and control is a key enabler for future technology developments. The emerging CPS are set to enable a modern grand vision for society-level services that transcend space and time at scales never possible before [2]. Contemporary opportunities and research challenges include the design and development of next-generation airplanes and space vehicles, hybrid gas-electric vehicles, fully autonomous urban driving and prostheses that allow brain signals to control physical objects.

In CPS, embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice-versa [3]. This means many things happen at once. Physical processes are compositions of many things going on at once, unlike software processes, which are deeply rooted in sequential steps. In [4] computer science is described as "procedural epistemology," knowledge through procedure. In the physical world, by contrast, processes are rarely procedural. Physical processes are compositions of many parallel processes. Measuring and controlling the dynamics of these processes by orchestrating actions that influence the processes are the main tasks of existing embedded systems. Consequently, concurrency is intrinsic in CPS. Many of the technical challenges in designing and analyzing embedded software stem from the need to bridge an inherently sequential semantics with an intrinsically concurrent physical world [3].

In the architecture of the experiment presented in this work, which is more detailed in further chapters, the design time and runtime processes are distinctively separated. The design time includes the process modelling and process simulation, accounting for the full manufacturing processes that the system is set to achieve. The runtime includes the execution of such models to follow the shop floor production in real-time. This is possible by including IoT (Internet of Things) agents, responsible for the data acquisition and pre-processing, and the inclusion of all available data from the existing ERP systems and incoming production orders for the factory. This data is fed into a process orchestrator, which oversees the coordination of all runtime modules, which processes and forwards it to the data consumers, i.e. the company managers and the process

978-1-5386-7097-2/18/$31.00 ©2018 IEEE

improvement modules such as a scheduler and a capacity planner. The selection of these modules for the improvement of the manufacturing process is directly correlated with the needs of the use-case scenario, which is presented further in this work.

The mentioned use-case scenario and corresponding and implementation resulted from the participation in the CPMSinCPPS experiment, included in the BEinCPPS project.

### A. *CPMSinCPPS Vision*

The main objective of the BEinCPPS CPMSinCPPS experiment is to investigate the applicability of the BEinCPPS components in the SME's (Small and Medium-sized Enterprises) environment for improving the performance of their manufacturing processes. BEinCPPS envisages the full adoption of cyber-physical production systems and related business models by European manufacturing SME's by 2020 [5]. Hence, it promotes a series of business experiments that seek to integrand and experiment a Future Internet (FI) based machine-factory [6].

The industrial use-case (CRISTEMA) for this work, is a young SME focused on cutlery manufacturing that is seeking to move towards an operational CPS-based production process with the support of BEinCPPS modeling and simulation components for the design and execution of their processes.

SME's are the most active and common company profile in the northern part of Portugal, the location of CRISTEMA, and many other diversified regions mainly directed at manufacturing services. Their willingness to accommodate the integration of CPS modules in their manufacturing process and adhesion to CPPS is firmly based on their perception that value added services will result from the technological evolution and in the development of future better tools that are expected to guarantee better conditions.

For these purposes, which are set to accomplish the project requirements and also to provide significant scientific content to this experimentation, this work consists in the presentation of the advantages regarding the CPMSinCPPS solution and the involved BEinCPPS components, with focus on the different stages of process modelling and simulation (separating Design Time and Runtime) and on the analysis of the industry requirements and functionalities. It also consists in the presentation of the architecture that comprises the different modules of the CPMSinCPPS proposed solution and, finally, the detailed experimentation of this work in the use-case scenario.

## II. CPMSinCPPS Solution and BEinCPPS Components

The solution presented in this work is divided into two different stages, as can be seen in Fig. 1 (the architecture of the solution and explanation of each component is presented in the next chapter). The design time is where the user's knowledge gains shape and results in a representation of the factory's production processes. This knowledge is provided by the different types of users (of the different roles in the company) that have their own understanding of the factory and production stage that they're responsible for. It's when these different perceptions come together, that the modelling activity can bring real value and feedback to enable better strategic planning and proportionate a better integration, coordination and control during the overall process of the production execution.

The runtime stage is where the execution of the modelled processes occurs, enabling the monitorization of the production.

In this specific case, the presented process is a polishing process, relevant for the use-case scenario that is presented

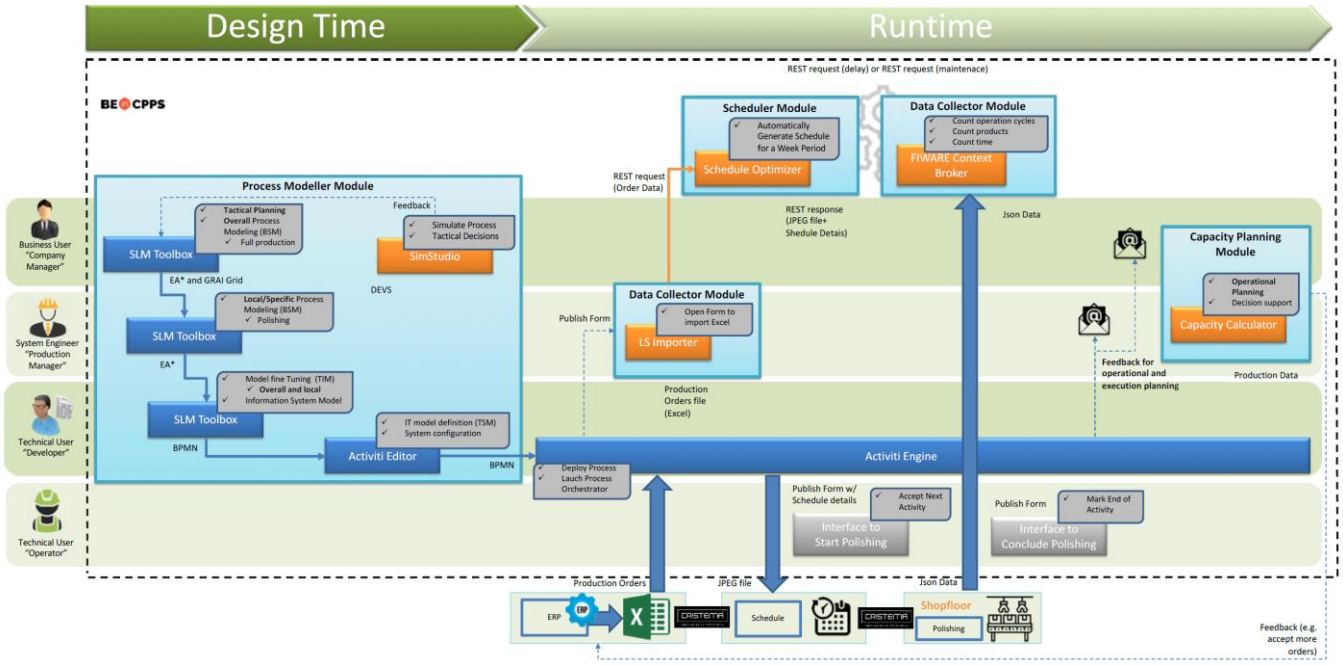


Fig. 1 - Design Time and Runtime Procedures in CPMSinCPPS.

further in this work. This machine is relevant for the company due to the lack of automatic scheduling (the previous schedules were made by hand), the need of long periods for maintenance or setup for accommodating different kinds of products and the fact that all the company's products must pass by this machine, creating a production bottleneck. The solution allows relevant feedback about the polishing activity and, based on the production orders (and the type of products to be produced) that must be completed, creates an optimized schedule for a time frame one week, which results in an optimization of the production.

### *A. Design Time*

This phase (illustrated on the left of Fig. 1) is the beginning of the project development. An initial analysis determines the business objectives of the company, that in specific case of the use-case scenario were:

1. Improve the performance of the production by a better planning and estimation of production time.
2. To propose corrective and preventive actions to gain agility in production.
3. Improve awareness on planners about the production status.

After defining these objectives for the process modelling, a model is created using the MSTB (Manufacturing System Tool Box) [7] [8], which is an intuitive enterprise modelling tool that can be easily used by end-users and does not require permanent support of a specialist. This tool is based on the concepts of the GRAI (Graph with Results and Activity Interrelated) model [9], developed by the GRAI group of the University of Bordeaux.

In this step, the chosen manufacturing system is decomposed in three domains: the physical system (PS), which produces the products, the decisional system, which controls the PS, and an information system which supports the storage and exchange of information internally and externally. The resulting model, considering these three domains, is created, based on the business requirements of the factory and the technical worker's information about the production process.

With the intent of creating a valid and viable proof of concept, the mentioned model results in a BPMN (Business Process Modeling Notation) diagram [10] that will serve the purpose of providing the different production steps to be modelled and simulated during the execution of the factory's production. The model in MTSB (Model Driven Services Enterprise Architecture), as can be seen in Fig. 2, is then exported to ACTIVITI, a tool for executing business processes included in the BEinCPPS project.

In the ACTIVITI editor, the exported simplified version of the BPMN process (Fig. 3) needs to be refined to allow it to be executed properly. Several fields, inside the tool, determinate some aspects of the its behavior and allow to insert some additional logic, through a JAR (Java Archive) file, to be carried out during the process, e.g. e-mail notifications about the status of a certain machine.

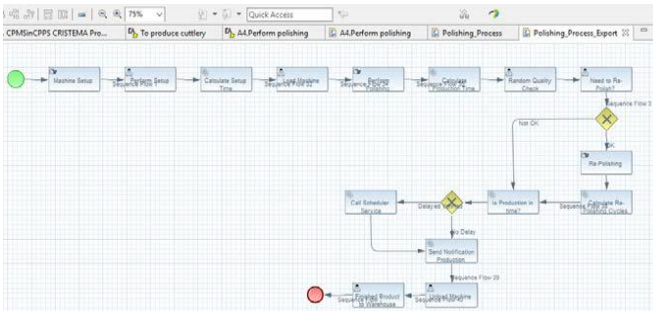

Fig. 2 - MSTB Diagram for Polishing Machine.

After the BPMN process is ready, the design time phase is finished. The modelled process can be executed along with the factory's production and the ACTIVITI tool engine functions as process orchestrator, coordinating the software and hardware components of the architecture. This is possible due to an IoT device deployed on the polishing machine. With this, the company administrator is able get information such as when the activities are completed, in real-time, the number of polished, re-polished and wasted cutlery and production/setup times for the machines.

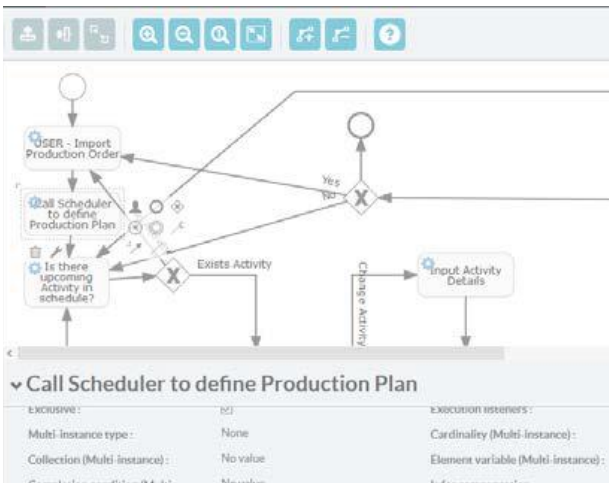


Fig. 3 - BPMN process in ACTIVITI.

### *B. Runtime*

The runtime phase consists, as said previously, in following the real-time production process. This is possible with the usage of the ACTIVITI engine and the data fed to it by the IoT node deployed in the polishing machine. This IoT device is integrated in the developed data collector module, which is detailed in the next chapter. To avoid any complication or waste of time in the production process, an addition tool was developed to be used by the factory workers. It consists in a very simple and intuitive user interface that allows the operator, of a certain machine, to carry out his work without significant inconveniences while providing relevant real-time information.

This interface is supported by an *Android* mobile device (tablet) place on the machine and allows the input of information such as choosing the next activity to be

performed by the machine or when a production step begins or ends (Fig. 4), or also check the production schedule for the machine, as illustrated in Fig. 5.


Figure 4 - User Interface screen for confirming the end of a production step *(Unloading the machine).*

With the information readily available in the ACTVITI engine, decisions can be made based on real-time data. This data is used by the Scheduler and Capacity Planner modules (explained in the next chapter) with the objective of optimizing the polishing process, as mentioned before.

With this brief description of the solution provided by this work, the next chapter is a more detailed explanation of the relevant components of the CPMSinCPPS architecture.


Fig. 5 - User Interface screen for checking the machine's production schedule *(allows confirming or changing the next activity to be performed).*

## III. CPMSinCPPS Architecture

To allow a better explanation of the solution in terms of the components that build the architecture of this work, Fig. 6 depicts the different layers, modules and components that integrate the presented solution. The following sub-chapters provide a focus on a functional description of some main architectural modules that aim to validate the process optimization provided by this experiment.

### A. *Process Orchestrator*

The purpose of the Process Orchestrator is to standardize, unify and automate the production processes of the factory, by aligning the various services available to perform the polishing process.

The Process Orchestrator runs on the ACTIVITI Engine and orchestrates the data received through Data Collector Module services, which are used to decide the next steps, and making the connection with the Scheduler Module and the Capacity Planner Module, enabling them to function properly.

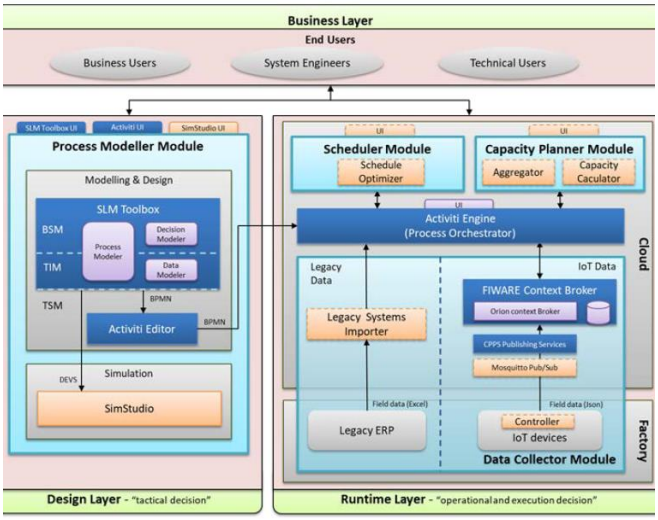


Fig. 6 – The CPMSinCPPS architecture.

### B. *Scheduler Module*

This module is a tool that addresses the typical scheduling problem, i.e. optimization problems that involve assigning resources to perform a set of tasks in specific times. In this solution, the scheduler module allows the automation of the creation of an optimized schedule for a specific machine, which helps the company's Production Responsible to set the order of production orders for the workweek. It is important to mention that in the scope of this use-case, resource optimization is not considered since only one polishing machine is available.

To better determine the sequence of activities, some rules were defined, prioritizing the urgent activities and considering the different types (or families) of products. The types, in this case, are related to the shape of the cutlery, and for each different type, it is necessary to make a specific re-configuration of the polishing machine. To optimize the timeline, the scheduling tool groups the maximum number of activities of the same type together, to avoid the machine preparation/re-configuration, which ultimately results in saving time.

With the integration with the process orchestrator, this tool maximizes the efficiency of the polishing process by swiftly responding (re-generating the schedule), to accommodate necessary readjustments, in face of delays or maintenance needs.

The implementation of the scheduler module can be divided into three sub-components: An optimizer, a front-end application and an API (Application Programming Interface).

The optimizer is the core engine of the scheduler and is responsible for generating the optimized schedule, based on the provided inputs and applicable constraints by using strategies as maximizing machine usage, minimizing setup time between orders and others.

The front-end application acts as the dashboard for the scheduler, where the end-user can view the entire schedule and status of the polishing machine, for each product (Fig. 7).

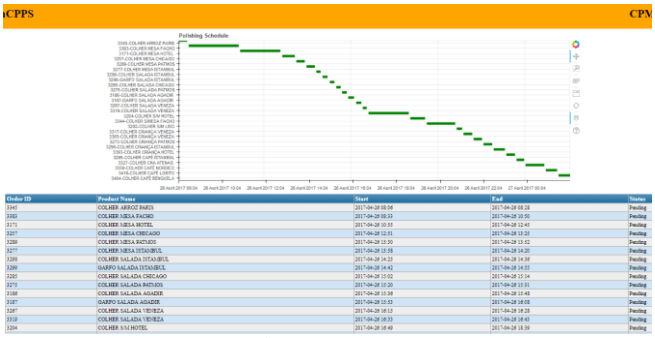


Fig. 7- Front-end application for the scheduler module.

The possible status for each product is Pending, Progress or Completed. The schedule, presented as a Gantt chart, provides the information about the start and end times for each polishing activity.

The API is a REST API for interaction with the orchestration engine, ACTIVITI engine. Provides the resulting schedule based on inputs such as an Excel file with new production orders, the completion of a activity (marks it as complete and responds with the next activity for the user-interface to display) or the need for maintenance (a maintenance is added to the schedule in the next immediate feasible timeframe).

### *C. Data Collector Module*

The data collector module integrates the FIWARE context broker and deploys a set of IoT nodes in the polishing machine. The objective is to increase the CPPS capabilities by collecting production data, providing it to modules such as the scheduler and the capacity planner through the process orchestrator, which then can relay feedback to the company and production managers.

The IoT is an emerging technological concept that aims to combine consumer products, industrial components and other everyday objects with Internet connectivity and powerful data analytic capabilities that can transform the way we work and live [11]. Combined with CPPS, the IoT objective is to optimize functionalities, as every extractable information becomes a mean of analyzing and computing the functioning process, to enhance its context functioning and to provide new purposes, that emerge when connecting a certain object to an intelligent network [12].

The data collection process concerns an important step for the establishment of an efficient and dynamic industry solution of an intelligent system, in the IoT contemporary environments [6]. Its main function is to fill the gap between devices and information systems, comprising the acquisition (heavily based on the low-layer devices from the network), the processing and management of raw field data retrieved from the physical worlds into viable information on which the applications can make decisions upon.

The objective of the data collection module, in this work, is to comply with the factory's business requirements by retrieving data from the factory shop floor and production systems, processing it and providing relevant information to the higher-level modules of the architecture, such as the capacity planner and the scheduler. This process of data collection uses the existing systems in the CRISTEMA's factory production process, performing basic ETL (Extract, Transform, Load), with additional middleware processes in the cloud and hardware, on-site sensors and nodes, for better data harmonization, filtering and extraction, into the internal data structure of the CPMSinCPPS architecture.

The CPMSinCPPS middleware technology is located in the cloud. This approach is denominated as cloud computing, where the information goes directly from the IoT devices and nodes into the cloud for processing. This is less expensive, suitable for the contemporary SME's, and allows more complex processes but requires the allocation of some processing time in the data servers/consumers.

As mentioned before, the data collector module uses field data, retrieved from the deployment environment (order status and polishing functioning) to provide structured planning and scheduling for the polishing machine. Following the architecture presented in Fig. 6, the field data retrieved from the IoT devices, which have Internet capability, is forwarded to the FIWARE Context Broker submodule using the lightweight Pub/Sub messaging services of Mosquitto (M2M open source software for the MQTT (www.mqtt.org) connectivity protocol), which itself is handled by the CPPS Publishing Services (IoT Agent) (which acts as a gateway for communication devices using the MQTT protocol with NGSI brokers) in order to respect the structure already defined in the Orion Context Broker and relevant NoSQL MongoDB databases holding the collected information (in the JSON format). The FIWARE Orion is responsible for the lifecycle of context information including storage, updates, queries, registrations and subscriptions of the CRISTEMA's devices. From this context information is possible to easily retrieve and infer about production data (e.g. setup and idle times, production cycles and maintenance requirements).

The information retrieved from the existing CRISTEMA's Excel spread sheets of the Legacy ERP about production orders also plays an important role in this data collection process by providing the necessary information for other CPMSinCPPS modules. This section of the data collector module is parallel to the devices/FIWARE process.

A Legacy Systems Importer provides the data structuring to feed the scheduler module, which among other fields needs to use the priority and product ID fields of the production order along with the hardware restrictions of the polishing machine to optimize the best scheduling of the production orders available. The general communication process for the legacy data is relatively simple where the legacy systems importer simply adapts the data for the scheduler. The process for the field data originated by the IoT devices has a more intricate communication

methodology and is presented in Fig. 10. In both, the data collector feeds data into the architecture process orchestrator (that uses the ACTIVITI engine). This communication process allows essential interoperability for the functioning of the data collector module, providing certain functionalities through the CPPS Publishing Services (IoT Agent) capabilities of managing devices, configuration exchange, information forwarding and data structure. Initially, the CPPS Publishing Services (IoT Agent) defines the structure for the data in the Orion Context Broker and then performs discovery of the existing sensors, subscribing to MQTT topics that allow the communication with the devices. The sensors, connected to their respective nodes, pre-configured for communication, subscribe to the CPPS Publishing Services (IoT Agent) topics and reply with their identification. After that, the CPPS Publishing Services (IoT Agent) sends the full configuration of data collection to the devices, which then begin to publish the acquired field data (following the parameters of the configuration). This data is processed in the CPPS Publishing Services (IoT Agent) and sent to the Orion Context Broker. In the Orion, the information is stored in MongoDB NoSQL databases for further querying by the process orchestrator (ACTIVITI engine).

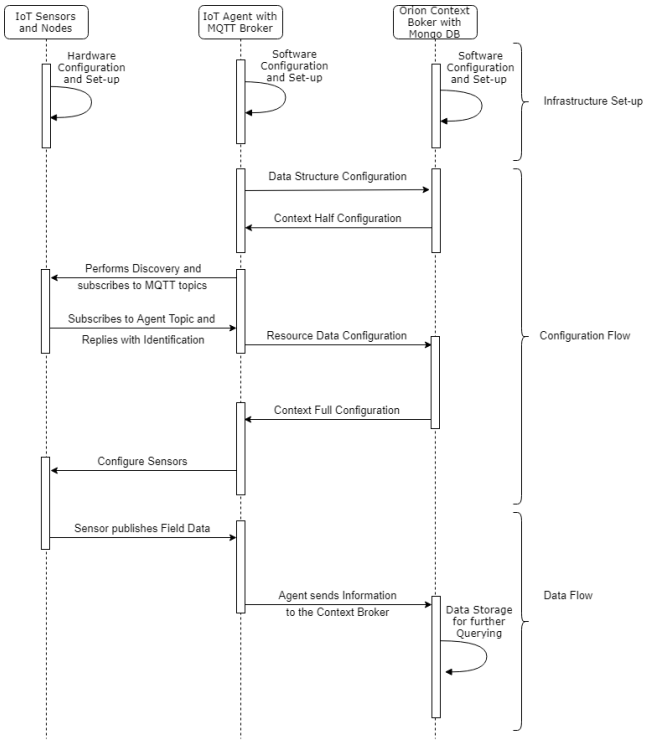


Fig. 7 - IoT Data Collection Communications Flow.

## IV. CPMSINCPPS USE-CASE

The site for the deployment and experimentation is the factory of CRISTEMA, in the north of Portugal. The production process of cutlery involves different processes, but the experiment prototype is only focused on the polishing process (Fig. 11).

### *A. Design Time*

Following the flow of actions represented in Fig. 1, the Company Manager and Production Manager need to start defining the overall process using SLM Toolbox and the Extending Actigram language [13], as described in section II.*A*. Then, as explained, the Developer takes up the modelling activity using that information and extending it, specifying two BPMN processes that will serve as the monitoring and control workflow for the runtime prototype: The polishing process, and the "master" process which coordinates the runtime loop.

Fig. 11 - CRISTEMA Polishing Machine.

The polishing machine process is first defined in the SLM Toolbox and later exported and parameterized in the ACTIVITI tool (Fig. 12). It explains the functioning of the polishing machine while allowing to gather information about the state of the production.

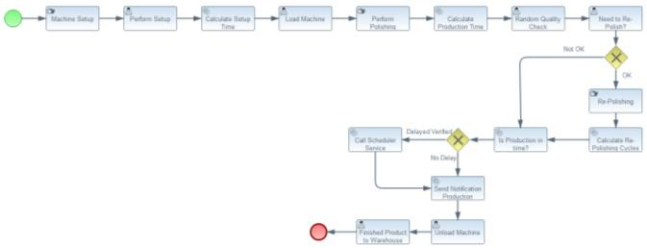

Fig. 12 - Process Model of Polishing.

There are four participants in this process, the Operator, the CPPS Publishing Services (IoT Agent), the Polishing Machine and the Production Controller (or Manager), as represented in Fig. 1. The operator only has user tasks, which require user feedback or approval when finished. These tasks are related to a quality check, transportation and machine setup and loading. The "polishing machine" participant represents the functioning processes that need to be mentioned but do not require the input information of a user (manual tasks), such as the polishing and re-polishing processes. The production manager is in charge of the production within the factory and receives notifications about the end of production orders. The CPPS Publishing Services (IoT Agent) represents the service tasks related with other module calls, calculation of times and cycles, usually done by calling a Java class.

The process description starts with the setup of the machine, first with the demarking of the setup initial time, then the machine setup performance and finally the calculation of the setup time. After that, the machine is loaded by an operator and the polishing process is performed.

During this task, the production counter is checked in order to determine if the production is finished. Each cycle corresponds to a number of products and the total number of products to polish is provided by the production order, before this process occurs. When the counter becomes equal or greater than the total number of products of the production order, the production duration time is calculated and the counter is reset.

After the production, the operator may perform a random quality check and if the piece of cutlery is not well polished, a re-polishing activity is performed. The number of cycles in this re-polishing process is stored in order to inform the production manager. After this, the current date is compared with the date predicted for the end of production in the schedule, to understand if there was a delay. If so, the scheduler is called in order to reorganize the schedule for the polishing machine and providing it.

A notification for the end of production is then sent as is followed by the user tasks of unloading the machine and transporting the products to the warehouse, which is assured by the operator.

The "master" process (Fig. 13) coordinates the runtime loop. Here, it all starts with importing the production order, which will provide information to the scheduler about the production orders available. After this is done, the scheduler API is called and provides the list of production orders to complete. The list provided is then checked and the next production order to perform (with production order ID, estimated time for the activity to end, number of products, etc.) is provided to the user. If there are no activities available, the process ends.

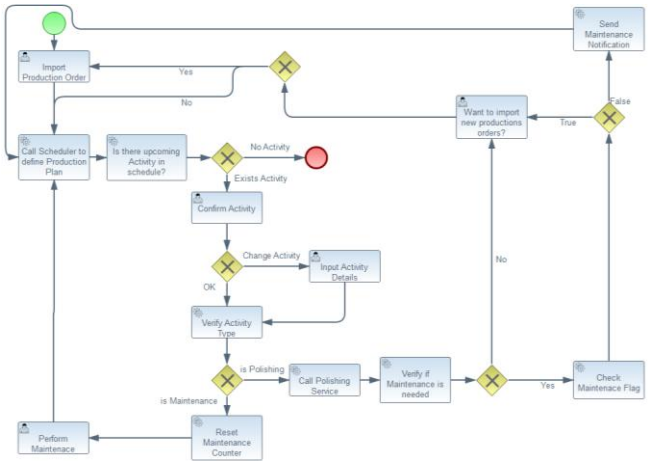


Fig. 13 - 'Master' Process Model.

The user confirms if the next activity should be performed right away (this was defined automatically by the rules of the scheduler but can be changed by the user if needed). If the user chooses to change the activity, it has to provide the details in order to send them back to the scheduler when the polishing ends, to acknowledge that particular task as finished and exclude it from further schedules.

The type of activity is verified next, to understand if it is a maintenance or a polishing one. If it is a maintenance, the counters and flags for the machine maintenance are reset, and the scheduler is called again (providing it with information about the activity performed). If it is a polishing activity, the BPMN service presented previously is called. When the polishing process ends, the maintenance counter is checked. If it is above 10000 cycles, the maintenance flag is verified. This prevents the BPMN process to be stuck sending the notification for maintenance needed when the maintenance is not made right away due to optimization or user demands. If the flag is false, that mentioned notification is sent to the scheduler and the flag is set to true (meaning the machine needs a maintenance soon, and it will only turn to false after that maintenance occurs). If the flag is true it means a maintenance was already flagged and acknowledged by the scheduler, and the process must continue.

After this, new production orders are checked. If any exists, the user must attach the correspondent spread sheet and import it to the scheduler. If no new production order exists, the process continues with another call for the scheduler in order to continue the production orders already scheduled for.

### *B. Runtime*

Now that the BPMN for the main process is explained and saved, the ACTIVITI generated app becomes be available following the deployment guidelines. From the initial menu in the ACTIVITI web interface, the CPMSinCPPS app should be visible.

To run it, the app should be selected and in the next menu, the "Start a new process and then track its progress" option should be selected. The following menu presents the option to "Start process" and that executes the BPMN process. The progress and interaction with the process tasks can be tracked on the ACTIVITI platform.

With the beginning of the polishing and the machine functioning, the Data Collector module starts receiving the IoT installation in place (Fig. 15).


Fig. 15 – CRISTEMA's IoT Installation.

The ACTIVITI, Mosquitto (MQTT Broker), CPPS Publishing Services (IoT Agent), FIWARE Orion and

related DB's, are all implemented and running on an accessible cloud server (multi-container Docker applications), within the network of UNINOVA.

## V. Conclusion

In this paper, it was presented the final prototype of the CPMSinCPPS experiment, that was selected in the second open call of the BEinCPPS project, to carry out innovative Modelling & Simulation (M&S) experiments in CPPS-based manufacturing systems, based on the components/artefacts of the project reference architecture.

The experiment was conducted on a part of CRISTEMA's production system related to the cutlery industry, with particular emphasis on the polishing process (bottleneck of the production due to high setup times and low processing capacity).

To reach the final prototype, the processes are organized in design-time and runtime phases, embedding the set of components developed to satisfy the requirements. The main outcomes can be summarized as follows:

- The final prototype of the CPMSinCPPS solution deployed and available for the use-case experimentation;
- Elaboration of the business process models;
- Preparation of production data needed for the prototype experimentation;
- Final CPMSinCPPS architecture available;
- Implementation of the Process Modeller module, customizing the SLM Toolbox to ensure proper interoperability with the remaining components;
- Implementation of the Data Collector module, integrating the FIWARE context broker and deploying a set of IoT nodes in the polishing machine;
- Implementation of the Scheduler module;

The Process Modeler, the Data Collector and the Scheduler have been the principal components involved in the intelligent system of this experiment. The team is still looking for a third-party solution that could provide the Capacity Planner full functionality. For the purposes of the prototype, it has been implemented as a set of Excel functions.

For future work, it is intended to finalize the Capacity Planner and provide a mobile application to facilitate the operator's work and be more user-friendly. For the Scheduler, the next steps are related with solving some bugs and to improve accuracy of the time constraint (improve capability to detect holidays, weekends, etc.). On the Process Orchestrator, the future work also starts with resolving identified bugs and improving the overall polishing process, to make it more efficient and to facilitate the operator, when identifying the current production step, allowing better production feedback. This can be achieved by adding more sensors in the production line, to avoid the manual input by the operator.

## Acknowledgement

The research leading to these results has received funding from the EC H2020 Program under grant agreement n°. BEinCPPS 680633 (http://www.beincpps.eu/) and n° BDO 732310 (http://www.bigdataocean.eu/).

## References

[1] R. Baheti and H. Gill, "Cyber-physical Systems," *Impact Control Technol.*, no. 1, pp. 161--166, 2011.

[2] L. Sha, S. Gopalakrishnan, X. Liu, and Q. Wang, "Cyber-Physical Systems: A New Frontier," *2008 IEEE Int. Conf. Sens. Networks, Ubiquitous, Trust. Comput. (sutc 2008)*, pp. 1–9, 2008.

[3] E. A. Lee and S. A. Seshia, *Introduction to Embedded Systems - A Cyber-Physical Systems Approach*. 2017.

[4] H. Abelson and G. J. Sussman, *Structure and interpretation of computer programs*. McGraw-Hill Book Company, 1996.

[5] BEinCPPS, "BEinCPPS," 2017. [Online]. Available: http://www.beincpps.eu/.

[6] O. Vermesan and P. Friess, "Digitising the Industry - Internet of Things Connecting the Physical, Digital and Virtual Worlds," pp. 153–183, 2016.

[7] C. Marques-Lucena, J. Ferreira, M. Sesana, K. Fischer, and C. Agostinho, "Process Modelling Approach for the Liquid-Sensing Enterprise," *I-ESA'16 - Interoperability for Enterprise Systems and Applications*. Guimarães, Portugal, 2016.

[8] S. Gusmeroli, C. Agostinho, C. Marques-Lucena, M. Sesana, A. Felic, and K. Fischer, "OSMOSE : A Paradigm for the Liquid-Sensing Enterprise." 2015.

[9] D. Chen and G. Doumeingts, "The GRAI-GIM reference model, architecture and methodology," *Archit. Enterp. Integr. IFIP Adv. Inf. Commun. Technol.*, 1996.

[10] INCOSE, "Systems Engineering Vision 2020," *Syst. Eng. Vis. 2020*, no. September, p. 32, 2007.

[11] K. Rose, S. Eldridge, and L. Chapin, "THE INTERNET OF THINGS: AN OVERVIEW. Understanding the Issues and Challenges of a More Connected World.," *Internet Soc.*, no. October, p. 80, 2015.

[12] F. Lopes, J. Ferreira, R. Jardim-Goncalves, and C. Agostinho, "Semantic maps for IoT network reorganization in face of sensor malfunctioning," *2017 IEEE Int. Conf. Syst. Man, Cybern. SMC 2017*, vol. 2017–Janua, pp. 1914–1919, 2017.

[13] Y. Ducq, C. Agostinho, D. Chen, G. Zacharewicz, R. Jardim-Goncalves, and G. Doumeingts, "Generic Methodology for Service Engineering based on Service Modelling and Model Transformation," in *Manufacturing Service Ecosystem: Achievements of the European 7th Framework*, 2014.