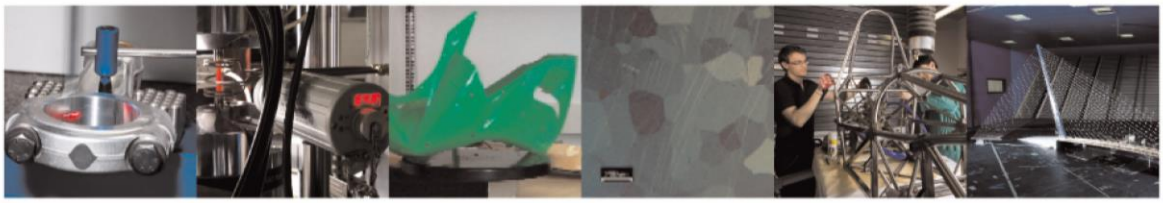

Formal modelling of release control policies as a plug-in for performance evaluation of manufacturing systems

Urgo M, Terkaj W

This is a post-peer-review, pre-copyedit version of an article published in CIRP Annals. The final authenticated version is available online at: http://dx.doi.org/10.1016/j.cirp.2020.04.007

Formal Modelling of Release Control Policies as a plug-in for Performance Evaluation of Manufacturing Systems

Marcello Urgo*, Walter Terkaj[2]

1 Politecnico di Milano, Department of Mechanical Engineering, Milan, Italy.
2 STIIMA-CNR, Institute of Intelligent Industrial Technologies and Systems for Advanced Manufacturing, National Research Council, Milan, Italy.

Abstract
Control policies significantly affect the performance of manufacturing systems, driving the need to assess their impact during both the design and operational phases. Performance evaluation tools can provide a relevant support, but their full exploitation is hindered by the difficulty of considering the huge variety of control decisions that are interwoven with manufacturing system configurations. Herein, a formal modelling approach is presented to jointly describe a manufacturing system and its release control policies, thus enabling the definition of performance evaluation models in terms of different policies. An application case is provided for the automatic generation of discrete event simulation models to assess the viability of the approach for assembly lines.

Control Policies, Performance Evaluation, Ontology Models

1. Introduction and Problem Statement

Release control policies play a key role to control the flow of jobs in a manufacturing system, i.e., deciding when a part is allowed to enter a workstation or a manufacturing area (e.g., a shop, a manufacturing cell, a subset of machines, etc.). Release policies take decisions whether a set of production resources should be working or remain idle, by releasing a part waiting to be processed on them. This has impacts on a wide range of key performance indicators related to lead time [1], level of the work-in-progress (WIP) and the associated stock-holding costs [2], the respect of delivery dates and the throughput rate [3]. In general, release control policies aim at finding a trade-off between keeping the WIP as low as possible and guaranteeing a high customer service level [4]. However, the choice of the best release control policy depends on several factors such as the characteristics and objectives of the company, the production system configuration and production planning strategies [5]. Thus, the optimized design of manufacturing systems and their management asks for methods able to explicitly take in consideration their configuration in terms of equipment, as well as release control policies.
Nevertheless, modelling and implementing release control policies in performance evaluation models is a rather complex task, both in case of analytical methods or discrete event simulation (DES). Indeed, analytical models [6][7] typically impose relevant constraints regarding the control policies that can be modelled in manufacturing systems because of the underlying mathematical hypotheses. On the other hand, DES models offer the capability to evaluate the performance of systems without imposing constraining hypotheses [8]. Nevertheless, the modelling of control mechanisms strongly depend on the specific commercial-off-the-shelf simulation package (CSP). As the control policy to be tested changes, small modifications (e.g., refactoring a rule or the associated variables) or, in many cases, more invasive revisions of the DES model are needed. The rigidity of performance evaluation approaches with respect to modelling control policies hinders the optimized design of manufacturing systems, considering that the huge variety of control decisions to be taken in a manufacturing system depend also on the actual system configuration.
With the aim of overcoming such limitations, this paper addresses the use of formal modelling approaches to support a modular representation of release control policies and their relations with production system configurations and production plans. Such a unified framework is a key enabler for the generation and fast reconfiguration of performance evaluation models with different control options in a manufacturing system, while guaranteeing an explicit definition of control policies that are not depending only on non-transparent assumptions at the implementation level. Modelling and simulation approaches can be extensively used to support and speed-up the decision process in the design stage of a manufacturing system, as well as for the definition of management decisions [9][10][11].
This paper is organized as follows. Section 2 describes the formal modelling approach with a focus on release control policies. Section 3 describes of how this model can support the definition of performance evaluation models; Section 4 presents an application case and, finally, conclusions are drawn in Section 5.

2. Formal Modelling of Release Control Policies

The proposed approach is centred on the concept of *Controller* that determines how a production system and its resources are managed during manufacturing execution. The *Controller* is directly related to the following concepts (Figure 1):

- *Production Resources* (e.g. workstations, buffers) whose behaviour is managed by the controller. A controller can have impact on one or more production resources.
- *Control Policy*, i.e. specific rules and algorithms adopted by the *Controller*. A policy typically makes use of observed variables (e.g. buffer level) related to production resources.

- *Production Plan* that consists in scheduled activities assigned to *Production Resources* within a given planning horizon. The execution of the plan is enforced by the chosen *Controller*.

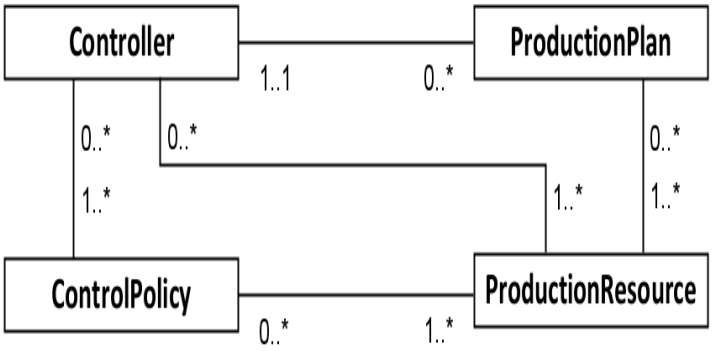

Figure 1. Conceptual model as an UML class diagram for controller, control policy, production resource, and production plan.

The *Controller*, modelled after the IEC 61499 standard, can be seen as a *function block* that employs an execution control chart (ECC) to control the execution of its algorithms. The ECC is usually modelled as a finite state machine [12], where the transition from one state to another can be either authorized by a verified condition or triggered by an event signal.
The conceptual model in Figure 1 has driven the enhancement of a reference factory data model [13] providing a common and extensible representation of factory objects related to production systems, resources, processes, and products. This reference factory data model is available as a modular OWL ontology based also on existing technical standards (Industry Foundation Classes, W3C SSN/SOSA, UML Statecharts), as listed in Table 1. Herein, the novel ontology module named *controlSystem* was developed and integrated into the existing architecture,, where *factory* (prefix *fa*) is the root module.

Table 1 List of ontology modules with prefix names. All modules are available online at the same address, except *fsm* that can be found at http://people.cs.aau.dk/~dolog/fsm/fsm.owl.

Prefix	Prefix IRI of ontology module
cs	http://www.ontoeng.com/controlSystem#
expr	https://w3id.org/express#
ex	http://www.ontoeng.com/expression#
fa	http://www.ontoeng.com/factory#
fsm	http://www.learninglab.de/~dolog/fsm/fsm.owl#
ifc	http://ifcowl.openbimstandards.org/IFC4_ADD1#
ifcext	http://www.ontoeng.com/IFC4_ADD1_extension#
list	https://w3id.org/list#
osph	http://www.ontoeng.com/osph#
sosa	http://www.w3.org/ns/sosa/
ssn	http://www.w3.org/ns/ssn/
stat	http://www.ontoeng.com/statistics#

An excerpt of the ontology is shown in Figure 2. Although the proposed scheme can be generalized to a wide range of control approaches, herein the focus is on release control policies, whose relevance has been motivated in Section 1.
Several release controllers can be defined in a model, but each production plan (*fa:ProductionPlan*) can be associated with a single release controller (*cs:ReleaseController*) that is in charge of controlling the system while producing the requested amount of products. The release controller is characterized by a state machine (*fsm:StateMachine*) that can be decomposed into several orthogonal regions (*fsm:Region*).
Each region models the control behaviour of a specific production resource (e.g. a machine tool) by containing two states: a release state (*cs:RControllerReleaseState*) that allows parts to be released to the controlled production resource and an idle state (*cs:RControllerIdleState*) that inhibits the release of parts.

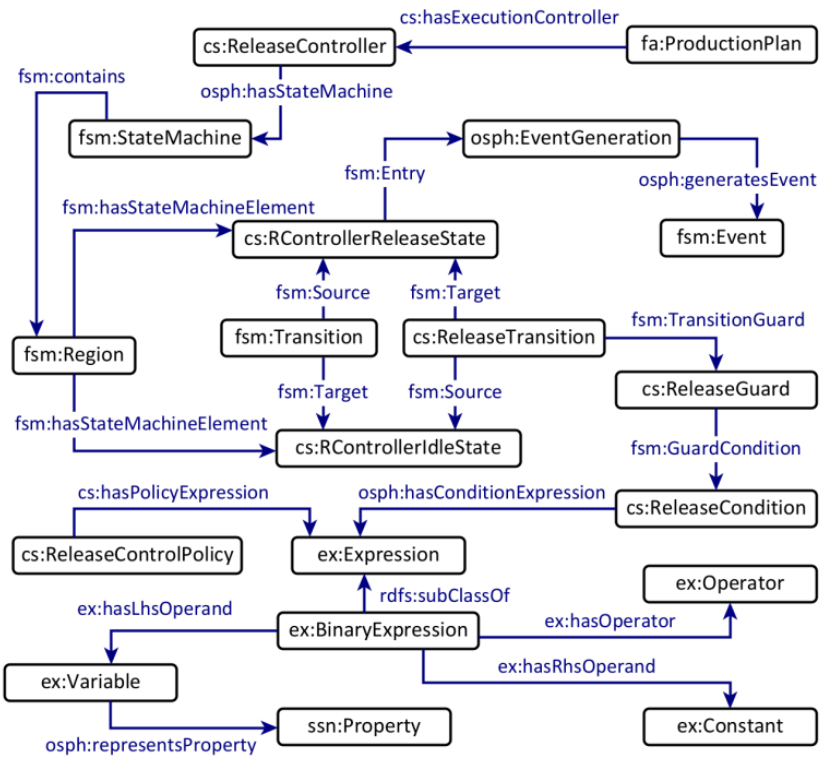

Figure 2. Core ontology to model the release controller. Boxes represent classes, whereas arcs represent object properties in OWL language. Prefixes defined in Table 1 are placed before class and property names.

A control policy (*cs:ReleaseControlPolicy*) is characterized by an expression (*ex:Expression*) that defines the guard condition (*cs:ReleaseCondition*) to be satisfied to trigger the release transition (*cs:ReleaseTransition*). The formulation of the expression can be arbitrarily complex; as an example, Figure 2 shows a binary expression (ex:BinaryExpression) that depends on a constant value (*ex:Constant*) and an observable variable (*ex:Variable*) representing the property (*ssn:Property*) of a production resource. If the observed production resource is a buffer, then its property could be the buffer level (*fa:BufferLevel*).

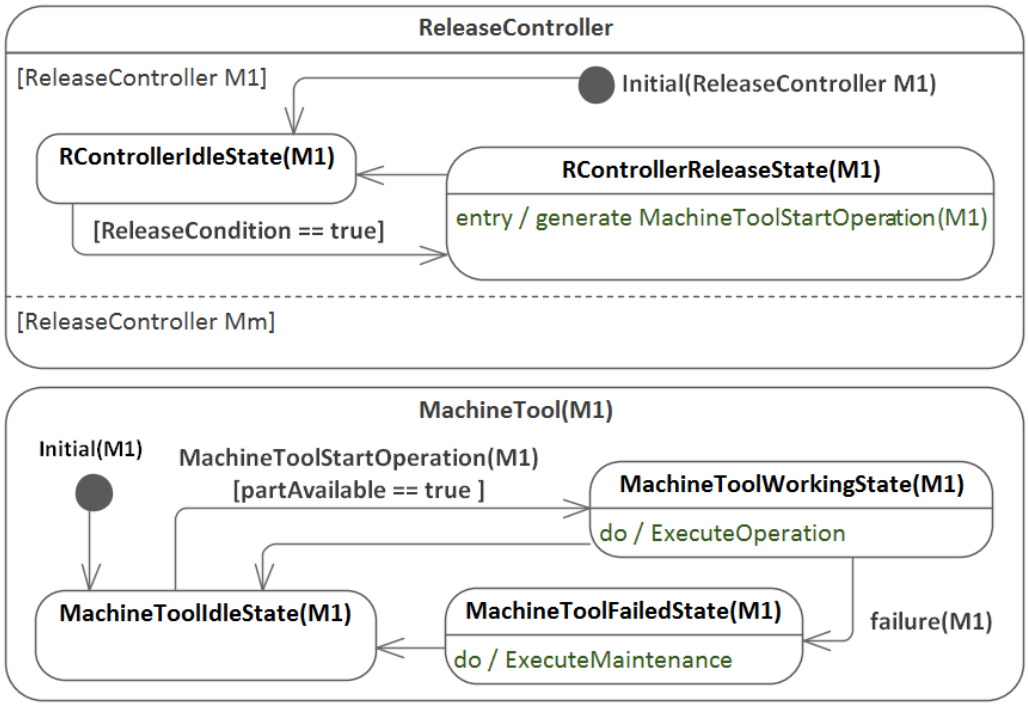

Figure 3: UML statecharts of Release Controller and Machine Tool

As soon as the controller enters the release state, an entry action (*osph:EventGeneration*) is started to generate an event (*fsm:Event*) that will trigger the actual release of a part when this event is intercepted by the state machine of the controlled production resource. When the entry action is run to completion the release controller goes back to idle state.
If the controlled production resource is a machine tool, then the event generated by the release controller will specify that an operation can be started (class *fa:MachineToolStartOperation)*. This case is further represented in Figure 3 by means of UML statecharts that are consistent with Figure 2.
The statechart of the release controller (Figure 3, top) shows the decomposition into orthogonal regions dedicated to a single machine tool (from *M1* to *Mm*). Each region takes care of the generation of a specific event belonging to class *fa:MachineToolStartOperation* when the release condition specified by the release policy is true.
The statechart of machine tool M1 (Figure 3, bottom) consists of three simple states (idle, working, failed). The event generated by the controller is intercepted to trigger the transition from idle to working state whenever a part to process is available (cf. guard condition). When the processing of the part is completed, the machine tool returns to the idle state.

3. A Plug-in for Performance Evaluation

The proposed modular modelling approach for control policies (specifically release policies) acts as a plug-in, enabling the possibility of activating/deactivating, switching and tuning the controllers and the policies operating on a given manufacturing system, without the need of reshaping or significantly modifying the whole modelling scheme. Each basic concept can be independently defined, and the overall behaviour emerges only after proper links are established. For example, a new controller can be plugged in a machine tool by simply creating a map between the event generated by the controller and the event intercepted by the machine tool. Therefore, reconfigurability is supported since production plans can be changed by switching controllers and a controller can be updated in terms of actions (i.e. controlled production resources), rules (i.e. adopted control policy) and observations (i.e. variables included in the policy).
For the sake of brevity, the plug-in potential of the modelling approach can be better illustrated by focusing the attention on a specific class of manufacturing system and release control policy, even though it can be easily extended to other cases.
The reference manufacturing system is a flow shop operating on a single part type. The production resources are workstations M_i ($i=1, \ldots, m$) and buffers B_j ($j=1, \ldots, b$). The possible material flows can be defined by a set of direct connections between machines and buffers (e.g. B_1-M_1, M_1-B_2). A process plan consists of a sequence of operations that can be assigned to a subset of workstations with a specific processing time. A production plan requests the production of a number of parts Q over a given time horizon H while adopting a release controller chosen in the set of available controllers R_n ($n=1, \ldots, r$). The release controller can be defined as a set of pairs (M_i, P_k) linking each controlled workstation M_i to its release control policy P_k ($k=1, \ldots, p$).
Herein, the reference release control policy is a CONWIP (CONstant Work In Process) that constrains the number of parts in a given subsystem according to a threshold. The expression of a CONWIP policy P_k is defined in equation (1):

$$\sum_{i \in M(P_k)} w_i + \sum_{j \in B(P_k)} l_j \leq T_k \qquad (1)$$

where T_k is the threshold of CONWIP policy P_k, w_i is the number of parts in workstation M_i, l_j is the number of parts in buffer B_j, and $M(P_k)$ and $B(P_k)$ are the sets of observed workstations and buffers, respectively. The expression means that the controller associated with a CONWIP policy P_k allows a part to be released to the controlled machine M_i if and only if the number of parts in the observed production resources is not greater than the threshold T_k. In addition, complex policies can be defined by joining single expressions via logic operators. If the policy changes in terms of threshold T_k or observed resources, only the expression associated with the policy itself has to be modified.
The plug-in potential applies also to the use of performance evaluation methods, since scripting capabilities and the functionalities of the vast majority of discrete event simulation (DES) software packages can be exploited [8]. In this work, PlantSimulation by Siemens is used, but other CSPs (e.g., Arena by Rockwell Automation) could play the same role. In PlantSimulation, production resources like workstations are usually modelled through a set of classes: *SingleProc* for resources acting as a single server, *ParallelProc* for resources processing parts in batches, *Assembly* for resources operating an assembly operation and *DismantleStation* for disassembly ones. For all of these classes, the possibility for a part to enter them and be processed can be inhibited by locking the entrance. Thus, if a release controller is in state *RControllerIdleState* (see Figure 2 and Figure 3), the entrance of its controlled resource is locked. When a transition to state *RControllerReleaseState* occurs, the entrance is unlocked and, if a part is available in the upstream buffer or machine, then it is allowed to enter and be processed. A script is embedded in the simulation model to calculate the expression associated with the control policy and then lock/unlock the entrance of the controlled resource. This check is operated every time a part leaves a production resource that is monitored by the expression of the active control policy.
Through this modular approach, controllers and policies can be activated/reconfigured both at model (ontology) level and simulator level.

4. Application case

4.1. Case description

The proposed approach is applied to support the design of an automatic system for the assembling of drawers' slides in the furniture market. The product to be assembled (Figure 4) consists of four parts: the upper, middle, and lower components and the self-closing mechanism.

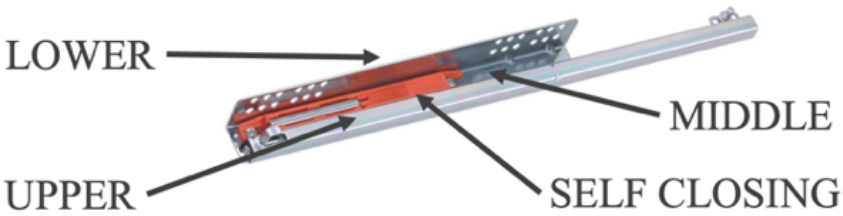

Figure 4. Drawer slide and components (courtesy of Cosberg).

The product is assembled in the system represented in Figure 4 that consists of three subsystems (A, B, C) processing three sub-components, and a subsystem D assembling them onto the remaining one to obtain the final product. The whole system consists of 26 workstations. The assumption is made that the first workstations in subsystems A, B and C are never starved, i.e., they always have parts to process.

The release controller operates on five workstations: four of them are placed at the beginning of each subsystem and the fifth workstation is placed in subsystem D after the connection with subsystem C. Each controlled workstation is associated with a CONWIP release policy (P_1, P_2, P_3, P_4 and P_5, respectively). P_1, P_2 and P_4 monitor the parts in subsystems A, B and C, respectively; P_3 monitors the number parts in subsystem D before the final assembling of parts coming from subsystem C; P_5 monitors the number of parts after joining of all the components.

Since the parts flowing in the system are quite often mounted onto a specific pallet, release control policies also impact the number of pallets that must circulate in the different subsystems and, thus, the associated investment cost.

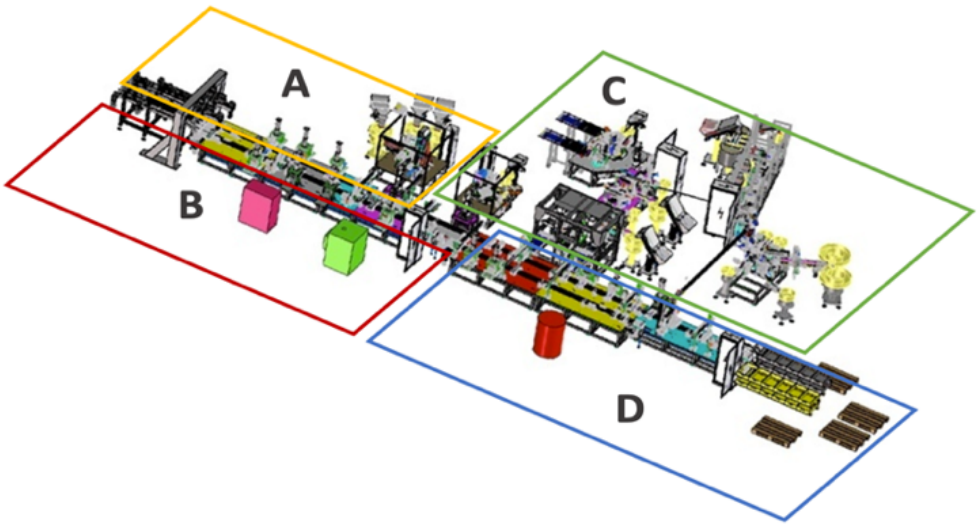

Figure 5. Layout of the automatic assembly system (courtesy of Cosberg).

4.2. Experiments

A first investigation has been carried out to assess the impact of control policies on the performance of the assembly system. Different control policies have been tested by varying the threshold T_k of the five CONWIP policies between one and the total number of part places in the monitored subsystem.

Grounding on the availability of a formal description of the hardware configuration (workstations and buffers) and the associated control policies, a discrete event simulation model is automatically generated using the PlantSimulation software package. This approach exploits the method described in [10], enriched with the capability of instantiating controllers and control policies according to the procedure in Section 3.

The performance in terms of throughput (*TH*, parts per time unit), parts in progress (*WIP*) and number of pallets in the system (*#pallet*), obtained by varying the control policies, are compared with reference values of the same configuration without enforcing any policy. The results are reported in Table 2, showing the average, minimum and maximum percentage difference (*%D*) with respect to the reference case. An optimized control policy has a significant impact on WIP and number of pallets (on average 24.6% and 33.7%) given a small negative impact on the throughput (on average 5.6%). Starting from the current system configuration, and activating optimized policy *(P_2, P_3, P_4)* and thresholds, it was possible to reduce *WIP* and *#pallet* of 21.2% and 37.3%, with the same *TH* (0.016% lower). Nevertheless, the impact range is wide and, in some cases, it could have no impact. Hence, the joint design of release control policies and system configuration is mandatory to take advantage of the full benefit.

Table 2. Impact of control policies on the system performance

KPI	Avg %D	Min %D	Max %D
TH	-5.6	-45.8	0.0
WIP	-24.6	-50.9	0.0
#pallet	-33.7	-68.4	0.0

Hence, the proposed approach has been exploited to optimize the design of the assembly line described in Section 4.1. Starting from the current configuration of the system (workstations and buffers), possible alternative configurations have been identified by varying the capacity of the buffers. Thus, a subset of the most promising solutions has been further investigated by varying the thresholds of the CONWIP policies, to find a near-optimal configuration as a trade-off between throughput and *WIP*.
For each of the possible configurations, 10 replicates have been executed, 144-hour long with a 24-hour warmup. A total of 500 different solutions have been tested for a total of 5000 simulation runs. The results are reported in Figure 6. The points in green represent solutions obtained by varying the position and dimension of the buffers without any active control policy.
The other clusters of points (pink, orange, red, blue and grey) represent the solutions having the same hardware configuration but different control policies. The Pareto-efficient frontier for the analysed solutions with respect to the throughput and the *WIP*, shows that staying in the optimal region (bottom-right of Figure 6) requires the tuning of a proper set of control policies with respect to the available hardware configuration.

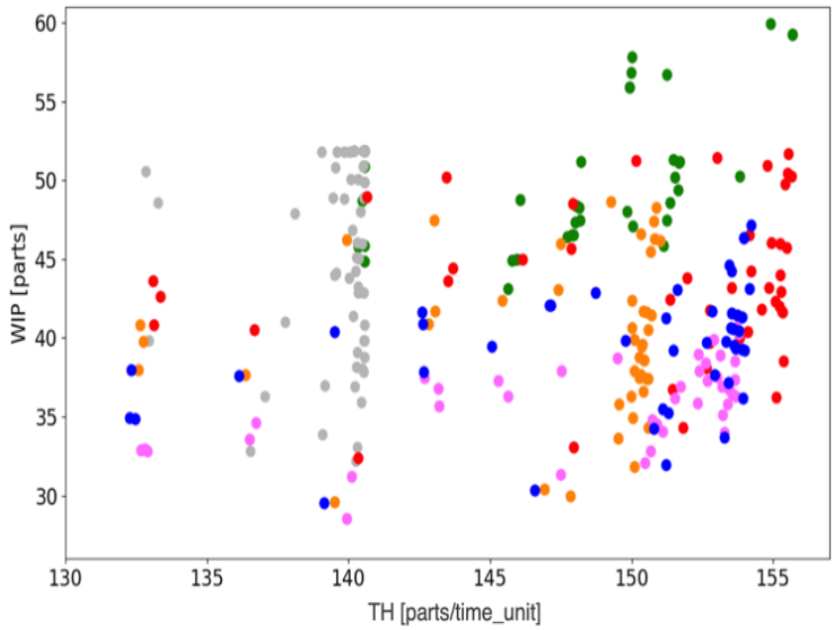

Figure 6. Results of the simulation experiments.

5. Conclusions

A formal modelling approach was proposed to support the optimal configuration of a manufacturing system and its release control policies by enabling the generation of DES models that represent a system together with different control policies. The approach has been demonstrated for a single product assembly line although its generality makes an extension to different and larger problems possible. Further developments will address other architectures (e.g., job shop), multiple parts and different classes of control policies. The authors thank Cosberg S.p.A. for the support in the definition of the application case.

References

[1] Philipoom PR, Fry D (1992) Capacity-based order review/release strategies to improve manufacturing performance. International Journal of Production Research 30(11):2559-2572.
[2] Ma YH, Koren Y (2004) Operation of Manufacturing Systems with Work-in-process Inventory and Production Control. CIRP Annals 53(1):361-365.
[3] Wisner JD (1995) A review of the order release policy research. International Journal of Operations & Production Management 15(6):25-40.
[4] Bonvik AM, Couch C, Gershwin S (2010) A comparison of production-line control mechanisms. International Journal of Production Research 35:789-804.
[5] Stevenson, M., Hendry, L.C., Kingsman, B.G., 2005, A review of production planning and control: the applicability of key concepts to the make-to-order industry, International Journal of Production Research, 43:869-898.
[6] Colledani M, Magnanini MC, Tolio T (2018) Impact of opportunistic maintenance on manufacturing system performance. CIRP Annals 67(1):499-502.
[7] Duffie NA, Roy D, Shi L (2008) Dynamic modeling of production networks of autonomous work systems with local capacity control. CIRP Annals 57(1):463-466.
[8] Kádár B, Lengyel A, Monostori L, Suginishi Y, Pfeiffer A, Nonaka Y (2010) Enhanced control of complex production structures by tight coupling of the digital and the physical worlds. CIRP Annals 59(1):437–440.

[9] Gödri I, Kardos C, Pfeiffer A, Váncza J (2019) Data analytics-based decision support workflow for high-mix low-volume production systems. CIRP Annals 68(1):471-474.
[10] Terkaj W, Tolio T, Urgo M (2015) A virtual factory approach for in situ simulation to support production and maintenance planning. CIRP Annals, 64(1):451-454.
[11] Frazzon EM, Kück M, Freitag M (2018) Data-driven production control for complex and dynamic manufacturing systems. CIRP Annals 67(1):515-518
[12] Dai W, Dubinin VN, Vyatkin V (2014) Migration From PLC to IEC 61499 Using Semantic Web Technologies. IEEE Transactions on Systems, Man, and Cybernetics: Systems 44(3):277-291.
[13] Urgo M, Terkaj W, Giannini F, Pellegrinelli S, Borgo S,(2019) Exploiting Modular Pallet Flexibility for Product and Process Co-evolution Through Zero-Point Clamping Systems. In: Tolio T., Copani G., Terkaj W. (eds) Factories of the Future. Springer.