Plan Executor MES: Manufacturing Execution System Combined with a Planner for Industry 4.0 Production Systems

Petr Novák(✉), Jiří Vyskočil, and Petr Kadera

Czech Technical University in Prague – CIIRC, Jugoslávských partyzánů 1580/3, 16000 Prague, Czech Republic
{petr.novak,jiri.vyskocil,petr.kadera}@cvut.cz
http://ciirc.cvut.cz

Abstract. Industry 4.0 production systems have to enable flexibility in products, processes, and available production resources. Types of production resources can vary not only during maintenance process of the production systems, but also at runtime. Manufacturing recipes and assignments to production resources can no longer be hard-coded in automation and control systems, but the production has to be planned and scheduled dynamically with regards to the current status of the production systems and of customer needs. This paper proposes an architecture for a new generation of manufacturing execution systems that are tightly coupled with planners. The proposed approach is demonstrated on the Industry 4.0 Testbed use-case. An exemplary production plan deals with a robotic assembly of a construction made up from Lego bricks.

Keywords: Production system · Automation system · Planning · Control · Manufacturing execution system

1 Introduction

Current industrial production systems are becoming very complex and large-scale from their design and control perspectives. Components of production systems such as 6-axis robots, conveyor belts, work stations, or milling and 3D-printing devices have to be integrated in a flexible way to be able to react on changes in market needs. They can lead to redesigning products being manufactured on the production system, production system is frequently updated to be suitable for new types of goods, and the efficiency of the production and of the maintenance processes becomes very important for industrial stakeholders.

Operation of industrial production systems is automated by (industrial) automation systems. They have a hierarchically layered architecture, which is frequently called an automation pyramid. Many particular graphical expressions of the pyramid exist, depending on what aspects they emphasize. One of the

V. Mařík et al. (Eds.): HoloMAS 2019, LNAI 11710, pp. 67–80, 2019.
https://doi.org/10.1007/978-3-030-27878-6_6

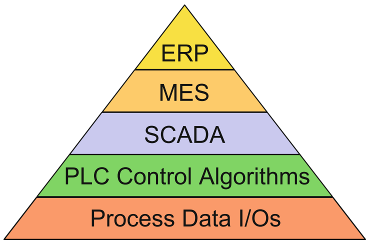

Fig. 1. Automation pyramid expressing the layered architecture of industrial automation systems. This paper contributes to the MES layer by extending it with a planner.

representations can be found in [9]. Although research effort as well as needs in industry tend to flatten the pyramid into a flexible dynamically re-configurable ecosystem as a part of the Industry 4.0 movement, the solutions being used in industry nowadays and in the near future still strongly rely on the hierarchical structuring. Due to this fact, we still consider the automation pyramid in this paper as a reference architecture for industrial automation systems.

The automation pyramid depicted in Fig. 1 represents the view on the data architecture in automation systems considered in this paper. On the very bottom layer, process data are measured by sensors in shop floor (i.e., process data outputs) and set up to control the shop floor (i.e., process data inputs). The low-level real-time control is done by programmable logic controllers (PLCs) or adequate hardware with equivalent functionality, which constitutes the second level of the automation pyramid. On the third level, a SCADA system (abbreviation standing for supervisory control and data acquisition) is a system that is intended to provide access to industrial plants, both for human operators and for upper software systems. The most visible part of SCADA systems are human machine interfaces (HMIs). They are intended to access runtime data by human operators, and to set up actions and set-points by them. They typically visualize trends, current values and their limits; they can violate alarms when any value exceeds its required limits. On the fourth level, there are manufacturing execution systems (MESs). They are responsible for executing production plans, assignment to manufacturing resources and reporting warehouse flows and product flows to the fifth level. MES systems play a role in improving manufacturing and financial performance of industrial companies [3] as many companies still rely either on manual planning and executing of production or utilize very basic MESs that are not capable even to re-order production tasks within a batch. On the fifth level of the automation pyramid, there is an enterprise resource planning (ERP) system. ERP is frequently considered as a synonym for SAP, but there exist many other ERP systems such as ABRA.

This paper is focused on improving the fourth level of the automation pyramid, i.e., on MES systems. They should no longer be simple monolithic systems, but they should consist of various sub-tools, including a planner and a digital twin for the real production system. The planning system should be generic and it should not be just a heuristics for a specific limited class of problems.

The remainder of this paper is structured as follows. Section 2 formulates the state of the art by summarizing related work in the areas of MESs and production planning. Section 3 presents the proposed improved architecture of MES systems equipped with planning, which enables not only planning of the production, but also re-planning when a problem occurs. The proposed architecture is demonstrated in Sect. 4 utilizing Industry 4.0 Testbed as use-case. The paper is concluded in Sect. 5 providing ideas for future work as well.

2 Related Work

Since the proposed approach emphasizes a tightly-coupled combination of MES and planner, the related work can be found in both domains, but with limited overlap between them in current solutions.

2.1 Manufacturing Execution Systems

Limited capabilities of commercial MESs are discussed in [1], which is mainly focused on distributed MES. The article identifies the basic common set of components of distributed MES: (1) order managers, (2) resource managers, (3) supervisors, and (4) brokers. This classification is in line with [15], dealing with strengths and threats of adopting multi-agent paradigm for industrial control especially on the MES level.

Among holonic architectures for MES, we have to remind the reference architecture PROSA [2]. The name refers to three basic types of holons: (1) product holons, (2) resource holons, and (3) order holons. Each of them is focused and responsible for a specific view on holonic MES and the operation of the system is driven by negotiation among these holons. The PROSA architecture has been extended and adapted in many ways, one of the most successful and famous is the architecture ADACOR [14]. ADACOR introduces four types of holons according to their functions and objectives: (1) product, (2) task, (3) operational, and (4) supervisor holons. The holon types (1)–(3) correspond to PROSA holon types, whereas the supervisor holon type is an extension in ADACOR that does not have an equivalent in the PROSA architecture. Although both PROSA and ADACOR are capable of solving a wide range of industrial tasks, they have not been widespread in industrial applications, excepting several pilot studies. The lessons learned from these reference architectures reflected in this paper are the focus on maximal utilization of current industrial systems used in practice and the focus on traditional way of thinking and programming rather than significant paradigm shifting as it was in strict holonic and multi-agent architectures.

Design of generic MES systems is discussed in [4]. The paper identifies the following key components of MES systems: "(1) Equipment management, (2) Production process management, (3) Quality management, (4) Order management, (5) Production scheduling management, (6) Resource management" [4].

After detailed discussion of these areas, utilization of Java 2 Platform, Enterprise Edition (J2EE) is proposed in [4], as it is platform independent and it is suitable for flexible implementation and integration of modules realizing the aforementioned needed components of MES.

Possible trends in MES development are discussed in [16]. They can be summarized as: "(1) cloud-based MES (2) IoT-based MES (3) intelligent MES (4) collaborative MES (5) supply chain linkage (6) MES mobility (7) industrial data analysis" [16]. An example of IoT-based solution for MES is discussed in [6], where IoT provides access to a broad range of sensor data.

MES functionalities can be implemented with the framework Eclipse BaSyx[1]. It is built on top of Eclipse framework. BaSyx is in progress and has not been finished yet. It should provide various communication possibilities including OPC UA and REST, a workflow engine based on BPMN 2.0, and support for various emerging industrial standards.

Integration between MES and ERP is standardized by ISA-95 [20]. This standard provides terminology and models for data integration and the approach presented in this paper is inline with ISA-95 modeling approach.

2.2 Automated Planning and Scheduling

The terms planning and scheduling are often used in different contexts with different meanings. Therefore, we have to first define these terms in the context of this paper.

Automated planning and scheduling (sometimes denoted AI planning [7,8]) is a branch of artificial intelligence that solves a problem of finding a plan (a set of tasks that need to be completed) that is represented as action sequences or action graphs (typically for execution by smart control systems, robots, or even for partial execution on various connected devices/autonomous agents) for given domains where allowed actions and related constraints are defined.

In fully specified environments with complete domain models available, planning can be done offline. Solutions/plans can be found and evaluated prior to execution. In dynamically changing environments (such as industrial production lines or shop floors), the plans frequently need to be revised online. Domains and policies have to be adapted. Finding such solutions usually tend to iterative trial and path finding/branching commonly seen in artificial intelligence. It includes machine learning, dynamic programming, and combinatorial optimization.

Planning refers to the action of establishing a plan, whereas scheduling is less concerned with what is being done and why, but more with when and where. A plan may (e.g., temporal planning) or may not (e.g., classical planning) incorporate times and dates associated to it, whereas a schedule most certainly will. Scheduling is concerned with mathematical formulations and solution methods of problems of optimal ordering and coordination in time of certain operations. Scheduling includes questions on the development of optimal schedules (Gantt charts, graphs) for performing finite (or repetitive) sets of operations.

[1] https://projects.eclipse.org/projects/technology.basyx.

The problems that scheduling deals with can be formulated as optimization problems for a process of processing a finite set of actions/jobs in a system with limited resources. In scheduling, the time of arrival for every action into the system is specified. Within the system the every action has to pass several processing stages, depending on the conditions of the problem. For every stage, feasible sets of resources are given, as well as the processing time depending on the resources used. Constraints on the processing sequence and actions are usually described by transitive anti-reflexive binary relations.

Given a description of the initial state of the world, a description of the goal conditions, and a formal specification of a set of possible actions, the planning task is to synthesize a plan that is guaranteed to generate a state (at the end) which satisfies all goal conditions.

For specification of planning tasks, several languages have been developed. *Planning Domain Definition Language* (PDDL) is supported by most state-of-art planners and we will use it also in this paper. The planning task/problem consists of two parts/files:

1. *domain description* – The problem-domain specification including every allowed *action* on state-space with its input parameters, *precondition* (condition that must hold before the action starts) and *effect* (description of changes on state-space immediately after the action is finished)
2. *problem description* – The specific planning-problem instance including its initial state and goal-state conditions.

A *solution* of some PDDL problem specified by its domain and problem description is a sequence of actions that can be sequentially applied (one by one) on the initial state of the problem and after application of all actions then the goal-state conditions of the problem are satisfied.

The latest version of the language is PDDL 3.1 [12] but there exist many variants/extensions that support various features like ontologies, probabilistic effects, numbers and goal-achieved fluents, durative actions (temporal planning), explicit problem decomposition (multi-agent planning) and others.

A selection of PDDL extensions including explanation of techniques in successful solvers is provided in [19]. A collection of simple prototypical industrial problems with their formalization in PDDL is presented in [18]. Compared to [18], the approach proposed in this paper is much more oriented to a real system of industrial scale, and we are able to utilize PDDL not only for planning but for digital twin (see Sect. 3.2) as well.

3 Proposed Architecture for MES

A large variety of current systems consist of relatively autonomous units. Such kinds of systems are frequently called systems of systems [11]. The problem of integrating autonomous units into one virtual system emerges not only in production system engineering, but also in many areas such as smart grids, water distribution networks, or logistics.

An important formal approach how to tackle these types of systems is a concept of multi-agent systems [22]. Although the multi-agent community has invested a lot of effort into a standardization of various properties and methods dealing with software aspects of distributed and multi-agent systems, the multi-agent or holonic systems still have not been widely spread in industrial applications. Due to this fact, the approach presented in this paper does not rely on any traditional multi-agent platform such as JADE, but uses a well-standardized communication protocol OPC Unified Architecture (UA), which is being widely adopted in industrial practice.

OPC UA is an industrial standard developed on a basis of the OPC classic specification and it combines data access, historical data access, and alarms & events into one unifying specification [13]. It is used for data acquisition from various distributed shop floor agents/actors such as PLCs, robot controllers, and smart sensors. Important benefit of OPC UA is that it is not limited for client-sever communication only, but it supports publish-subscribe communication as well. OPC UA is thus a privileged way for integrating the proposed MES with the two bottom-most layers of the automation pyramid.

From the top level side of the automation pyramid, we consider the traditional ERP systems to be used and we assume that manufacturing orders originate either in the ERP system, or they can be directly input via dedicated GUI, whose role is very crucial during commissioning of the system.

The overall proposed architecture of the MES accompanied with planner is depicted in Fig. 2. It includes fundamental components of the system (depicted with colored blocks) and data flows between them (depicted with numbered arrows).

In the previous text, we have already mentioned the shop floor layers of the automation pyramid (depicted in the bottom part of Fig. 2) to which the communication is solved via OPC UA (arrow numbered 8 in Fig. 2). As well, we have clarified relationships to ERP, which depicted in the upper part of Fig. 2.

The data flows are depicted in Fig. 2 by arrows and they have the following meaning, which is described in details in the subsequent paragraph:

1. Production order
2. Planning problem
3. Lisp plan without temporal information
4. Schedule as the lisp plan extended with temporal information
5. Production operation control based on OPC UA
6. Current and finished operations
7. Production status graph, statistics, and status
8. Status report about (semi-)products and warehouse flows to ERP
9. Re-plan order in case of a failure

The core part of the proposed approach is the Plan Executor (see the left-hand side part of Fig. 2). It accepts production orders from ERP or MES GUI (arrows 1a, 1b). The production order is passed (arrow 2) to the planner, which calculates the plan and if it exists, the planned production plan is provided to the scheduler (arrow 3). The plan extended with schedule information is

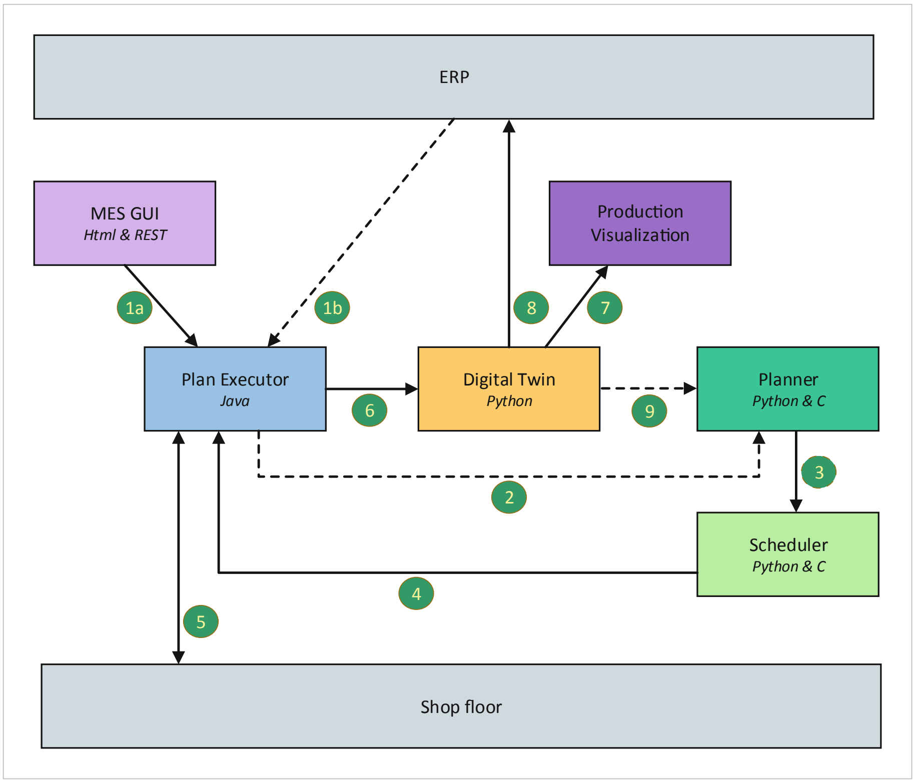

Fig. 2. Proposed architecture of the new generation of MES.

handed over back to plan executor (arrow 4). The plan executor parses this extended plan and considering OPC UA servers of production resource on the shop floor, it starts executing the plan. It starts such production operations on all resources, whose pre-conditions are satisfied. The communication related to starting operations on the shop floor and the backward notification about finished operation utilizes OPC UA (arrow 5). Checking the pre-conditions of remaining production operations and starting operations that can be started unless the production plan is finished are the main tasks of the plan executor. In addition, the plan executor updates the state of the digital twin (arrow 6) when any operation is started and finished. Hence, the digital twin has a detailed history of the production, which is important for visualizing the production for human operators (arrow 7). As well, warehouse status is updated in ERP system (arrow 8). Last but not least, if any failure happens, the digital twin detects such a problem and initiates re-planning (arrow 9). This is done in order to recover from the failure and to finish the production task.

3.1 Resource Management

The setup of production systems can be modified even at runtime. Some resources can be (re-)allocated for different production processes and in some special cases even to different manufacturers, such as in manufacturing as a service case which is based on reaching maximal utilization of resources by sharing them among a portfolio of production tasks. The importance for resource management is a crucial part of MES and even the traditional reference architectures such as PROSA and ADACOR incorporate foundations for the resource management.

A manual approach for the resource management is not sustainable for modern production systems due to the increasing level of complexity and needs for optimization and resource utilization. Thus the proposed solution utilizes a knowledge base facilitating management of available resource knowledge. Available resources are described in AutomationML. The data format AutomationML is standardized as IEC 62714, and it is becoming widely adopted for production system engineering.

The international standard ISA-95, which is widely accepted in industrial practice for many years, is used as a definition of basic terminology for the knowledge base model. This is one of the most important benefits of ISA-95 as its common terminology and modeling constructs target on systems of diverse types and engineering domains. ISA-95 knowledge can be serialized into the data format AutomationML. A bi-directional data transformation is standardized and specified by the Whitepaper [21], which is publicly available at the AutomationML Association website.

To be able to efficiently process the knowledge about the resources, we are using resource description in the AutomationML data format, with the use of ISA-95 terminology and models, and for processing purposes, we transform AutomationML to the AutomationML Ontology[2], which can be easily queried with SPARQL and new pieces of knowledge can be inferred with SWRL.

3.2 Digital Twin

A digital twin is a common term used for a digital replica of a physical system. Digital twins create living digital simulation models that update and change as their physical counterparts change. A digital twin continuously learns and updates itself from multiple sources to represent its near real-time status, working condition or position.

One of our contribution in this paper is to represent a digital twin by PDDL (see the Digital Twin module in Fig. 2). From PDDL point of view such a digital twin can be modeled and observed in the following way:

- Digital twin outer control signals need to be translated to PDDL actions that can be processes only under well specified conditions (PDDL preconditions)

[2] http://i40.semantic-interoperability.org/automationml/Documentation/index.html.

and that can have some effects to the internal digital twin state (PDDL effect on state-space).
- Interactions among digital twin components can be simulated by PDDL actions as well.
- The current PDDL state-space can access relevant output/sensor signals from digital twin sensors and returning values from digital twin components.

The major problem with creation of a digital twin, according to the previous points, is to translate outer control signals into PDDL actions. Sometimes PDDL actions need more information (as arguments) than the real outer control signals contain. In this case, the missing arguments need to be completed according to the preconditions of such actions.

Our PDDL digital twin can be used for the following different purposes:

- *Recomputation* of a new plan in case of failure or in case of any modification of production line.
- *Visualization* of the current state of the production line.
- *Global overview* that can support centralized, consistent, and formalized (computer readable) data source for further processing/analysis in related systems (e.g. ERP, predictive maintenance, etc.)

3.3 Planning and Scheduling

The task for the Planner module is to receive:

- The current status of a production line via the Digital Twin module.
- Goals of production from MES module.
- The specification of production line domain (operations/actions that are allowed on production line)

The output of Planner module is a sequence of operations/actions. The current implementation of Planner uses off-the-shelf Fast Downward Planning System [10].

The task for Scheduler module is to receive the output from Planner and creates a more detailed plan/schedule for MES module. For that purpose we developed a special format called *LispPlan*[3] that supports:

- *LISP like syntax* [5] that is human and computer readable. It can be quickly enriched by new features or we can easily encapsulate/translate this format into another more/better standardized format like XML.
- *Task and sub-task definitions* – tasks can be recursively divided into sub-tasks.
- *Locations* – description of resource location for each task.
- *Actions* – description of target actions/operations in PDDL format.
- *Requirements* – description of dependencies among tasks (tasks can be processed/executed in parallel).

[3] A short example of LispPlan is depicted in Listing 1.1.

The current implementation of Scheduler does not support scheduling tasks for concrete times and dates. Now, only accesses to resources are analyzed to produce LispPlan. In the future we would like to improve scheduling with time/duration support. For that improvement we can use algorithms based on widely studied and very successful Mixed Integer Programming [17] (as a part of combinatorial optimization techniques) or we can use temporal PDDL planners with durative actions support.

4 Industry 4.0 Testbed Use-Case

For the detailed explanation and evaluation of the proposed approach, this section describes the use-case dealing with the system Industry 4.0. The entire cyber-physical system is depicted in Fig. 3. The most apparent part of this experimental system is a monorail transportation system Montrac. It consists of rails called tracs, trac curves, trac switches, and positioning units that assure exact position of shuttles in working cells.

Three positioning units of the Montrac systems are accessed by four industrial robots. Each positioning unit is shared between two robots. This layout brings opportunity for cooperation between robots, which can be beneficial for example for final assembly.

The Industry 4.0 Testbed is equipped with industrial robots of the two types:

– 3x KUKA KR Agilus: Very fast industrial 6-axis robots programmed in the language KRL
– 1x KUKA LBR iiwa: Modern cooperative 7-axis robot programmed in the language Java

For testing purposes, assembling Lego bricks is used to evaluate designed approaches, algorithms, and tools in Industry 4.0 Testbed. The final Lego product is drawn in Lego Digital Designer[4]. This drawing is transformed to the problem file in the PDDL notation. The planner and scheduler are utilized to plan the production recipe in the form of LispPlan. The exemplary final product designed in Lego Digital Designer is depicted in Fig. 4. This assembly is uploaded via MES GUI to the Plan Executor, which hands it over to the planner. After planning the production operations, the plan is captured in the lisp plan format. An excerpt of the obtained production plan is shown in Listing 1.1. Subsequently, the production is scheduled and then executed by the Plan Executor by means of OPC UA communication to/from the shop floor.

[4] https://www.lego.com/en-us/ldd/download.

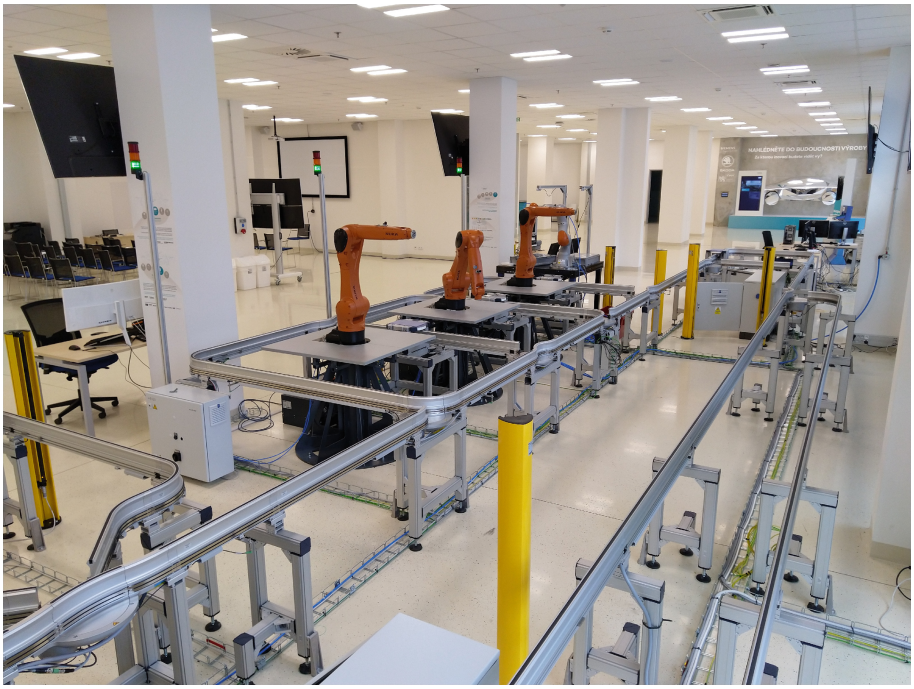

Fig. 3. Industry 4.0 Testbed at the Czech Technical University in Prague – CIIRC.

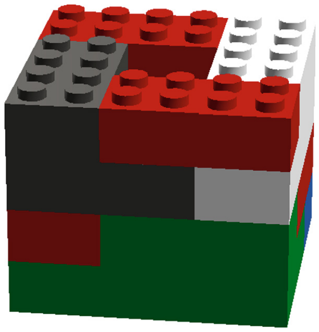

Fig. 4. Lego tower as a product to be built by the production system, exported from Lego Digital Designer 4.3.11.

Listing 1.1. Production plan example for the Lego tower

```
(define
  (task LegoProduct)
  (:location "testbed.ciirc.cvut.cz")
  (define
    (task 0)
    (:location SHUTTLE1)
    (:action (SHUTTLE_MOVE_LOCK SHUTTLE1 STATION3 STATION2))
  )
  (define
    (task 1)
    (:location SHUTTLE2)
    (:action (SHUTTLE_MOVE SHUTTLE3 STATION10 STATION100))
  )
  ... Skipping lines ...
  (define
    (task LegoAssembling)
    (:location R1)
    (define
      (task 0)
      (:action
        (PICKUP_ROTATION0 R1TABLE X-6 Y4 Z1 O0 BLACK)
      )
    )
    (define
      (task 1)
      (:requirements 0)
      (:action
        (PUTDOWN_ROTATION0 SHUTTLE1 X-5 Y-2 Z1 O0 BLACK)
      )
    )
    (define
      (task 2)
      (:requirements 1)
      (:action
        (PICKUP_ROTATION0 R1TABLE X6 Y4 Z1 O0 BLACK)
      )
    )
    (define
      (task 3)
      (:requirements 2)
      (:action
        (PUTDOWN_ROTATION0 SHUTTLE1 X-5 Y2 Z1 O0 BLACK)
      )
    )
  ... Skipping lines ...
  )
)
```

5 Conclusion and Future Work

To provide flexibility of production systems, manufacturing execution systems have to be prepared to fulfill flexibility requirements. The proposed architecture for a new generation of MES supports planning of production plans. Furthermore, when a problem in a production plan that is just being executed occurs, the continuously running digital twin provides the needed support for re-planning the remaining part of the production and continuing the manufacturing process. This is the issue that current commercial tools do not support.

The important strength of the presented approach is that the overall solution has been implemented on a prototype level and it has been deployed and tested in Industry 4.0 Testbed. It was utilized as a foundation for further testing of scientific and practical applications of methods and algorithms for Industry 4.0.

In the future work, we would like to strengthen the distributed nature of MES and to leverage it to a distributed MES. We would also like to integrate the proposed MES as a module that is able to cooperate with a commercial MES to make this approach better accessible for industrial partners without needs for significant re-implementations of current production plant setups.

Acknowledgements. The research presented within this paper has been supported by the DAMiAS project funded by the Technology Agency of the Czech Republic, by the H2020 project DIGICOR, and by the OP VVV DMS project Cluster 4.0.

References

1. Bratukhin, A., Sauter, T.: Functional analysis of manufacturing execution system distribution. IEEE Trans. Ind. Inf. **7**(4), 740–749 (2011). https://doi.org/10.1109/TII.2011.2167155
2. Brussel, H.V., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: PROSA. Comput. Ind. **37**(3), 255–274 (1998). https://doi.org/10.1016/S0166-3615(98)00102-X
3. Chao, L., Qing, L.: Manufacturing execution systems (MES) assessment and investment decision study. In: 2006 IEEE International Conference on Systems, Man and Cybernetics, vol. 6, pp. 5285–5290, October 2006. https://doi.org/10.1109/ICSMC.2006.385148
4. Fei, L.: Manufacturing execution system design and implementation. In: 2010 2nd International Conference on Computer Engineering and Technology, vol. 6, April 2010. https://doi.org/10.1109/ICCET.2010.5486065
5. Gabriel, R., Steele, G.: The evolution of Lisp. In: Companion to the 23rd ACM SIGPLAN Conference on Object-oriented Programming Systems Languages and Applications, OOPSLA Companion 2008. ACM, New York (2008)
6. Gao, Q., Li, F., Chen, C.: Research of internet of things applied to manufacturing execution system. In: 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), pp. 661–665, June 2015. https://doi.org/10.1109/CYBER.2015.7288019
7. Ghallab, M., Nau, D.S., Traverso, P.: Automated Planning - Theory and Practice. Elsevier, Amsterdam (2004)

8. Ghallab, M., Nau, D.S., Traverso, P.: Automated Planning and Acting. Cambridge University Press, Cambridge (2016)
9. Harjunkoski, I., Nyström, R., Horch, A.: Integration of scheduling and control - theory or practice? Comput. Chem. Eng. **33**(12), 1909–1918 (2009). https://doi.org/10.1016/j.compchemeng.2009.06.016
10. Helmert, M.: The fast downward planning system. J. Artif. Intell. Res. **26**, 191–246 (2006). https://doi.org/10.1613/jair.1705
11. Jamshidi, M.: Systems of Systems Engineering - Principles and Applications. CRC Press Taylor & Francis Group, Boca Raton (2008)
12. Kovacs, D.L.: Complete BNF description of PDDL 3.1. Language specification, Department of Measurement and Information Systems, Budapest University of Technology and Economics (2011). https://helios.hud.ac.uk/scommv/IPC-14/repository/kovacs-pddl-3.1-2011.pdf
13. Lange, J., Iwanitz, F., Burke, T.J.: OPC - From Data Access to Unified Architecture. VDE Verlag, Berlin (2010)
14. Leitão, P., Restivo, F.: ADACOR: a holonic architecture for agile and adaptive manufacturing control. Comput. Ind. **57**(2), 121–130 (2006). https://doi.org/10.1016/j.compind.2005.05.005
15. Mařík, V., McFarlane, D.: Industrial adoption of agent-based technologies. IEEE Intell. Syst. **20**(1), 27–35 (2005). https://doi.org/10.1109/MIS.2005.11
16. Pan, F., Shi, H., Duan, B.: Manufacturing execution system present situation and development trend analysis. In: 2015 IEEE International Conference on Information and Automation, pp. 535–540 (2015). https://doi.org/10.1109/ICInfA.2015.7279345
17. Pochet, Y., Wolsey, L.A.: Production Planning by Mixed Integer Programming. Springer, New York (2006). https://doi.org/10.1007/0-387-33477-7
18. Rogalla, A., Fay, A., Niggemann, O.: Improved domain modeling for realistic automated planning and scheduling in discrete manufacturing. In: Proceedings of the 23rd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 464–471 (2018). https://doi.org/10.1109/ETFA.2018.8502631
19. Sousa, A.R., Tavares, J.J.P.Z.S.: Toward automated planning algorithms applied to production and logistics. IFAC Proc. Vol. **46**(24), 165–170 (2013)
20. Unver, H.O.: An ISA-95-based manufacturing intelligence system in support of lean initiatives. Int. J. Adv. Manuf. Technol. **65**, 853–866 (2012). https://doi.org/10.1007/s00170-012-4223-z
21. Wally, B.: Application recommendation provisioning for MES and ERP - support for IEC 62264 and B2MML. AutomationML e.V. c/o IAF, 7 November 2018. https://www.automationml.org/o.red/uploads/dateien/1542365399-AR_MES_ERP-1.1.0.zip
22. Weiss, G. (ed.): Multiagent Systems, 2nd edn. Massachusetts Institute of Technology, Cambridge (2013)