Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 52-13 (2019) 2104–2109

Implementation of an IoT- and Cloud-based Digital Twin for Real-Time Decision Support in Port Operations

Wladimir Hofmann*, Fredrik Branding**

**, ** Lufthansa Industry Solutions AS GmbH, Schützenwall 1, 22844 Norderstedt, Germany*
**wladimir.hofmann@lhind.dlh.de, **fredrik.branding@lhind.dlh.de*

Abstract:

Seaborne trade volumes are growing and the resulting traffic load on the road infrastructure in port areas calls for new solutions to improve handling and coordination of vehicles and shipments. In this contribution, a digital twin for truck dispatching operator assistance is presented, which enables the determination of optimal dispatching policies using simulation-based performance forecasts. Proprietary simulation software with limited interfaces, the application deployment in demanding industrial environments, and the integration of real-time sensor information represent three major challenges when realizing digital twin applications for logistics systems. This implementation, therefore, uses an extensible open-source simulation package and it is coupled with an IoT-platform for easy integration of real-time information. The digital twin is deployed as a cloud-based service, allowing for simple deployment and scalability.

Keywords: Digital Twin, Internet of Things, Cloud Computing, SimPy, Maritime Economy

1. INTRODUCTION & PROBLEM DESCRIPTION

The world seaborne trade is still growing and the increasing volumes challenge not only the capacity of terminals but also the whole port area (United Nations 2017). Truck congestion at terminal gates due to peaks in arrivals, caused by rising ship dimensions and corresponding berthing of large vessels, often leads to heavy traffic (Lange et al. 2018). Recent terminal planning projects in German port areas face these problems and implement information technology solutions.

Due to the limited waiting area in terminals, arriving trucks are directed to additional buffer areas outside the port zone, where a first identification is done. Whenever there is enough space in the terminal, a dispatcher releases the next truck from the buffer area. The challenge for the dispatcher results from the fact, that there is a multitude of process fluctuations and interference factors (distance to the terminal, traffic congestion, duration of processing previous trucks at the terminal). Furthermore, different vehicle types need to be handled separately. Due to time differences between dispatch and arrival at the terminal, the situation can differ from the preceding estimation. The container terminal therefore continuously faces a conflict between over- and understocking. In the case of overstocking, there is not enough space in the terminal and the trucks need to wait in front of the terminal. This causes additional congestion. In the case of understocking, the loading area is not efficiently used. The consequence is less utilisation, thus leading to delays. Other restrictions are opening hours of the terminal and ship departure times. This creates the need for decision support for the operating dispatcher. Another challenge is to provide the system with sufficient information about the reality (real-time and historical data).

The outline of the paper is as follows: First, the underlying planning problem is analysed and related literature is presented, motivating the application of simulation methods, and followed by a detailed description of the digital twin concept. Subsequently, the solution concept and its implementation are described, both from a functional and technical perspective.

2. PROBLEM ANALYSIS & RELATED LITERATURE

The described problem of minimising the probability of deadline violations due to low utilisation of bottleneck resources, while at the same time minimising the expected time of exceeding soft buffer capacity limits can be addressed in different forms and using various solution methods.

Modelling the underlying planning problem as an inventory control problem represents one possible approach. The number of trucks that are located in the waiting area at the terminal may be seen as the system's inventory level at a certain point in time, while the demand varies according to the time actually needed for loading. Overstock situations lead to (undesirable) waiting times of trucks outside the waiting area due to the limited capacity, while stock shortages reduce the utilisation of the loading facility, thus leading to a delayed overall loading time (possibly causing delayed departure times of ships and/or overtime work). The lead-time for replenishment (driving time of the trucks to the loading area) is varying stochastically, as well as the demand (loading time of trucks). Since the average driving time depends on the traffic situation at the specific time of the day and since the average loading time varies according to the type of truck, both stochastic processes are non-stationary (Jula et al. 2006). Time may be considered to be continuous

10.1016/j.ifacol.2019.11.516

since loading may be finished and new orders can be released at any time (by dispatching a new truck to leave the buffer area). The (discrete) maximal number of waiting trucks determines the maximum inventory level. Consequently, a safety stock level at the waiting area needs to be determined, balancing the probability of over-stocking and stockouts. While simpler inventory management problems such as certain deterministic multi-period problems or even certain problems with stochastic lead time and demand (maintaining a given service level) can be solved optimally and efficiently in short time, the inherent complexity and stochastics make the problem described in this paper considerably harder. (Muckstadt and Sapra 2010)

From a scheduling perspective, the problem could be described as a stochastic flow-shop problem with intermediate storage, where the driving process of a truck may be represented by an unlimited number of parallel machines in the first stage (since different trucks may be driving towards the loading area at the same time). Target variables to optimise would be the makespan as well as the time of overstock situations (when more than the actual maximum number of trucks are located at the intermediate storage). Depending on whether the order of trucks to be dispatched might be assumed to be fixed or not, there might be defined as a static or a dynamic list policy (Pinedo 2016).

Additional complexity results from the fact, that there is no single target variable to be optimised. Rather, a configuration needs to be determined, which balances the expected time of both overstock and shortage situations, making this a multi-criteria optimisation problem. Due to different stakeholders influenced in various ways by the decision, this balance cannot be simply described e.g. by a fixed-weighted sum but needs to be determined by the system operator, communicating with the individually affected stakeholders (T'kindt and Billaut 2006). The successful solving of complex scheduling problems under stochastic influences with multi-criteria target functions is shown e.g. by Aurich et al. (2016), using a simulation-based optimisation approach. With respect to the operational use of this approach, Donhauser et al. (2018) describe a lack of studies implementing an automated application as one currently unresolved research challenge.

In manufacturing environments, order release mechanisms represent a widely used concept to solve problems of balancing lead times (work-in-progress) and system utilisation. Examples of dynamic control heuristics implementing pull-mechanisms to limit the work-in-progress are Kanban and ConWiP (Lödding 2013). Back in 1989, Glassey described an order release mechanism called “starvation avoidance”, which is focusing on single bottleneck systems. A virtual inventory of work time at the bottleneck is calculated from all corresponding orders (time for which the bottleneck station is supplied with work). Similar to before-mentioned inventory control problems, new orders are released when the inventory level drops below the time needed to reach the bottleneck station, shifted by a time buffer (safety stock) for compensating fluctuations (Mönch et al. 2013). Originally, the lead-time for orders to arrive was calculated by summing up processing times at stations to pass, ignoring transport and waiting times, and thus leading to imprecise estimations. In subsequent work, Glassey et al. extended their approach by a mechanism to estimate queue times based on real-time information (following the idea of computer-integrated manufacturing – CIM) (Glassey et al. 1989). The release decision is taken at periodic intervals, and the integration with continuous release concepts is still subject to ongoing research, as shown by Fernandes et al. (2017). In all mentioned works, simulation models were used to evaluate the performance of the concepts.

In a recent contribution, Kunath and Winkler (2018) highlight that these simple dispatching rules seldomly perform perfectly in more complex systems, whereas simulation-based decision support systems are able to increase the system performance by evaluating different rule configurations, enabling sophisticated optimisation, and considering complex stochastic influences. Rosen et al. (2015) describe the development of simulation from an experts’ tool in the 1960s, used solely for very specific applications, towards being a standard, core functionality of technical systems. Simulation is going to offer a seamless assistance along the entire life cycle and will be directly linked to operational data. Key enabling technologies of this more operational use of simulation include sensors and attached pre-processing hardware (edge computing), standardized communication protocols such as OPC UA and MQTT, as well as cloud computing and big data analytics.

In this context, the term “Digital Twin” (DT) is of growing popularity, extensively used in an increasingly blurred manner for different kinds of simulation models (Kritzinger et al. 2018). Originally, the term DT was coined by Michael Grieves in a PLM context meant for mirroring products (2002, at this time also called “mirroring spaces model”), and its sense later shifted from descriptive towards actionable, with focus on complex systems (Grieves and Vickers 2017). This paper adopts the definition of (Kritzinger et al. 2018), who describe a DT in manufacturing as the “virtual and computerised counterpart of a physical system, used to simulate it, exploiting real-time synchronisation of data”. There are three stages of integration distinguished, according to the degree of automation of the data flow between physical and digital object:

- *Digital Model* (manual data flow, state changes do not have a direct impact on the physical system and vice versa)

- *Digital Shadow* (automatic data flow from the physical to the digital object)

- *Digital Twin* (automatic data flow into both directions)

A DT is described to act as a controlling instance and allows forecasting and deciding between a set of actions in order to orchestrate the production system in an optimal manner, resulting in higher efficiency, accuracy and economic benefits.

Grieves and Vickers (2017) describe three different main use cases for DTs: maintenance, layout planning, and production planning and control (PPC). A mature example of a DT for maintenance purposes is the Aviatar project, which provides an open, modular and manufacturer-independent platform for collecting aeroplane data and realises condition monitoring and predictive maintenance services in aviation. (Aviatar 2018) The layout planning use case focuses especially on the evaluation of reconfiguration alternatives for changeable production systems (Hofmann et al. 2018). An example of a DT use case from the field of production planning and control is given in the paper at hand.

In their literature review, Kritzinger et al. (2018) conclude, that most of the reviewed literature on "Digital Twins" could rather be classified as "Digital Models" or "Digital Shadows" according to the proposed definitions. Furthermore, the share of bare "concept papers" grows with the level of integration to up to 55%, leading to the conclusion that DT research is still "at its infancy", expressed by a "lack of case studies which apply concepts in practice". Possible explanations include a growing complexity of the models, which is increasing additionally during the system life cycle, and the necessary computing infrastructure for managing the integration and central availability of data from various physical devices (Rosen et al. 2015).

3. DIGITAL TWIN: CONCEPT & ARCHITECTURE

The developed digital twin approach for truck dispatching operator assistance in port areas expands the usual dispatching process by a dispatching algorithm based on starvation avoidance (which calculates dispatching times), and by a simulation model which is used as a digital twin of the reality. The digital twin enables a forecast of the system performance and evaluates different dispatching policy alternatives (characterized by different safety buffer level parameters). This way, the operator can communicate with the affected stakeholders and decide for the best possible configuration in order to guarantee a stable utilisation for the loading area despite stochastic fluctuations, while still keeping the number of trucks waiting at the terminal within acceptable limits. Real-time information from the physical system serves both as input data and as an execution trigger for the digital twin.

3.1 Dispatching Process

The dispatcher is the central human role in this concept. He receives assistance from the output of the simulation. He knows the average loading times and the duration of transports from the buffer area to the terminal. He can only estimate the arrival time and the particular rest time for the truck in process in the terminal. Varying vehicle types increase complexity. Each vehicle has its own loading process in the terminal with different terminal areas.

Whenever a dispatching decision is executed, the cumulated remaining time in the system is compared with the desired arrival time, see (1). The desired arrival time represents the safety stock in front of the loading area and needs to be higher than the cumulated remaining time. Otherwise, there will be no dispatching activity triggered, see (2). The higher the safety factor is, the earlier will be dispatched.

The desired arrival time consists of the estimated processing time of the two trucks dispatched before the one, which is waiting and is multiplied by a safety factor, as described in (3). The safety factor S is adjustable and is the focus in the parameter studies in the following chapters.

The cumulated remaining time provides the dispatcher with information about the current state of the system. It contains the estimated processing time for all trucks currently on its way to the terminal or inside of it, see (4). For the truck in the gate is only the estimated remaining processing time for loading considered. The total time is corrected by the transportation time for the truck, which is waiting to be dispatched.

$$\text{to be dispatched: } t_{cumulatedRemaining} < t_{desiredArrival} \quad (1)$$

$$\text{no dispatching: } t_{cumulatedRemaining} > t_{desiredArrival} \quad (2)$$

$$t_{desiredArrival} = (t_{truck-1} + t_{truck-2}) * S \quad (3)$$

$$t_{cumulatedRemaining} = (\sum t_{transport} + \sum t_{entrance} + \sum t_{waiting} + \sum t_{gateRemaining}) - t_{transportDispatchable} \quad (4)$$

t : time S : safety factor

Whenever a truck leaves predefined zones, estimated data is compared with real-time data. Additionally, several simulation runs are triggered and offer the operator new predictions about different versions of the future. The impact of alternative S-values is displayed and automatically compared with each other. The operators' task is to monitor the system and in case of unforeseen conditions, he can modify the safety factor ($S_{current}$ → $S_{elected}$ → $S_{current}$) (Fig. 1). From now on, the dispatching algorithm regulates the system based on the new safety factor.

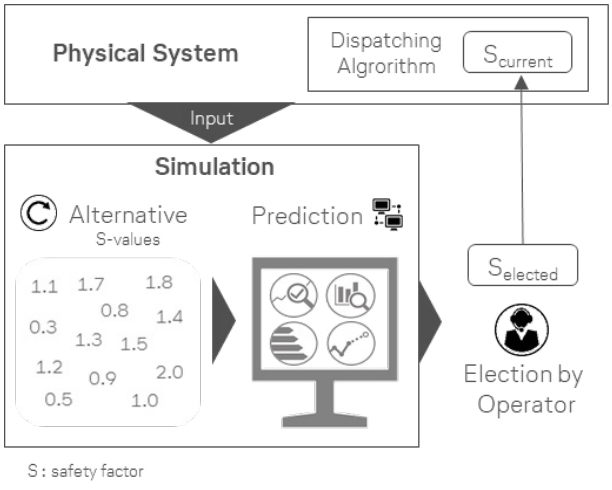

Fig. 1. Digital Twin Concept

3.2 Technical Architecture

There are four different data sources for simulation input data. The first is an existing appointment system, which contains information about the registered trucks for the current day. The second source is data from the past. The data consists of processing times in the terminal for different truck types and traffic information for the route from the buffer area to the terminal. Additionally, as a third source, it is possible to use external sources (e.g. HERE Routing API for more accurate estimations of the trip length). The fourth source is responsible to supply the simulation with real-time data via sensor technology e.g. from cameras and proximity sensors, registering the arrival of trucks at the different process stages.

The technical architecture is shown in Fig. 2. Input data changes are communicated to an IoT-platform and the data is stored in a database. The IoT-platform provides simplified device management as well as harmonisation and action management. It serves as a trigger for the simulation, which then uses the work in progress (WIP) information from the database to initialise the current system status. Subsequently, different dispatching algorithms are evaluated. The results are displayed in a Web-Application. The operator monitors the system and the dashboard to make strategic adjustments. Afterwards, the dispatching algorithm automatically releases the waiting vehicle in the buffer area.

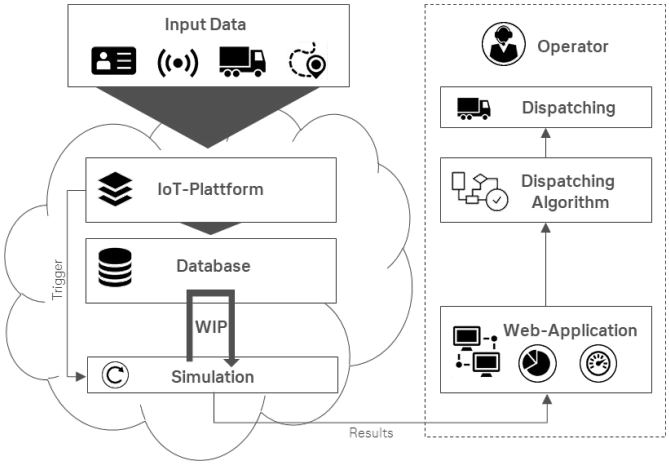

Fig. 2. Technical Architecture

4. DIGITAL TWIN: IMPLEMENTATION

To realise the described simulation functionalities, a simulation framework has been developed based on the popular open-source package SimPy for discrete event simulation (Simpy 2018). Python as the programming language has been chosen because of its widespread popularity, its high development speed, the high number of existing libraries, the native support that major cloud providers offer, and because of existing open-source discrete event simulation frameworks used as blueprints (Salabim, ManPy). Salabim is a simulation package that offers sophisticated animation possibilities based on Pillow and tkinter (Salabim 2018). Tkinter has also been used in our framework to create animation for debugging, model validation, and process visualisation purposes. ManPy is based on SimPy and offers a variety of manufacturing-specific object classes. It was developed as part of the EU funded research project "DREAM" (Dream 2018; Dagkakis et al. 2016) and our framework uses similar concepts of inheritance and of defining entity flows according to defined processes by the help of a model graph. The disadvantage of comparably high computation times and memory usage using Python is waived in favour of the described advantages and since simulation time has not been proving to be critical for this application. (Weingartner et al. 2009)

To ease the creation of simulation models containing longer and more complex processes, the Camunda modeler tool has been used for graphical process definition (see Fig. 3). By using a defined subset of available BPMN elements combined with naming conventions, model graphs can be created and parsed to derive corresponding python modules from base templates in order to extend or modify the predefined basic behaviour. Furthermore, the model graph is used to define the instantiation of model objects at runtime, and to define predecessor/successor relationships between these objects. The initialisation of work in progress (WIP) in the simulation model is realised using special entry points of the model elements, which account for seized resources as well as for already passed processing time at the simulation start by comparing an entities entrance timestamp to the starting time of the simulation. Subsequently, the WIP entities follow their normal processing flow. Model verification and validation were done using code reviews and expert interviews.

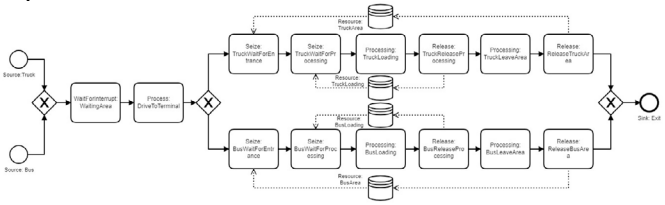

Fig. 3. Model Graph

The IoT platform is used to process events from connected devices and to trigger the execution of simulation runs by invoking a defined workflow via web API, which queries the database for simulation input data, and then invokes the simulation service. This service is deployed as a server-less function, taking input arguments, running the simulation, and submitting the computed results to a database, which feeds a web dashboard for operator information. The automatic scaling of the simulation service allows multiple simulation runs to be executed in parallel to both capture stochastic behaviour, and to evaluate different system configuration alternatives. Integration testing of the overall implementation was done using a simulation model acting as an emulation. This model was then connecting to the IoT-Platform, emulating the physical system. For more information on this concept, see (Gutenschwager et al. 2000; Hofmann et al. 2018).

5. APPLICATION

The implemented decision support system assists the dispatcher by estimating the expected cumulative duration of overstock and shortage situations in terminal operations. In the following part, an analysis of system parameters and simulation results derived from an application example are presented.

5.1 Parameter studies and computational experiments

Several parameter combinations have been analysed. For the distance from the buffer area to the terminal different values were included. The loading times in the terminal vary according to truck type and utilised loading capacity.

The focus of the parameter studies is on the influence of the desired arrival time used by the dispatcher (3.1). The higher it is, the earlier will be dispatched in the buffer area. The reason for this is the safety factor, which is multiplied with the desired arrival time (Fig. 4). S represents the safety stock level parameter. If S is zero, the average maintained safety buffer is zero. Assuming fixed driving and processing times, the utilisation is high, but considering even small fluctuations, the utilisation of the loading area is reduced substantially.

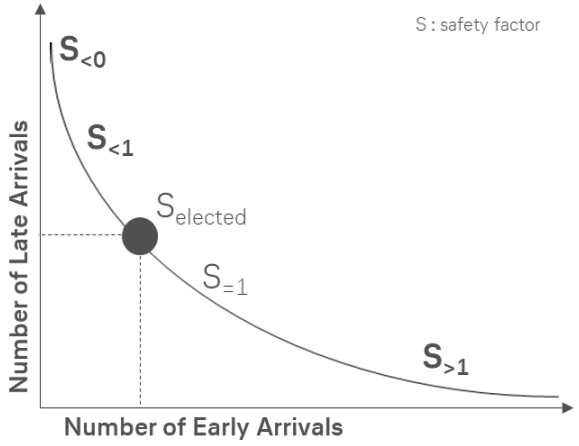

Fig. 4. Adjustment of the Safety Factor

With S=1 the buffer is balanced and the late arrivals as well. Statistical influences are equalised. When S is higher than 1, the early arrivals at the terminal entrance increase. A lower S has a negative effect on the utilisation at the loading area, caused by the high number of late arrivals.

With a growing distance between the buffer area and the terminal and the corresponding absolute growth of the fluctuations in driving times, the number of late and early arrivals increases accordingly.

5.2 Application for operative usage

The application for operative usage shows the potential of adjusting parameters of the algorithm. The dispatcher can adapt his policy according to operative influences (Fig. 5). In the beginning, a safety factor (S) is defined to realise the desired time of completion ($t_{desired}$) and utilisation ($u_{desired}$). Caused by stochastic influences, the estimated value can differ from the actual value. S is adjusted to modify the arrival frequency of incoming trucks. At t_3 the estimation for the time of completion is higher as expected beforehand (e.g. due to an equipment failure in the loading area, leading to uncommon deviations of the time).

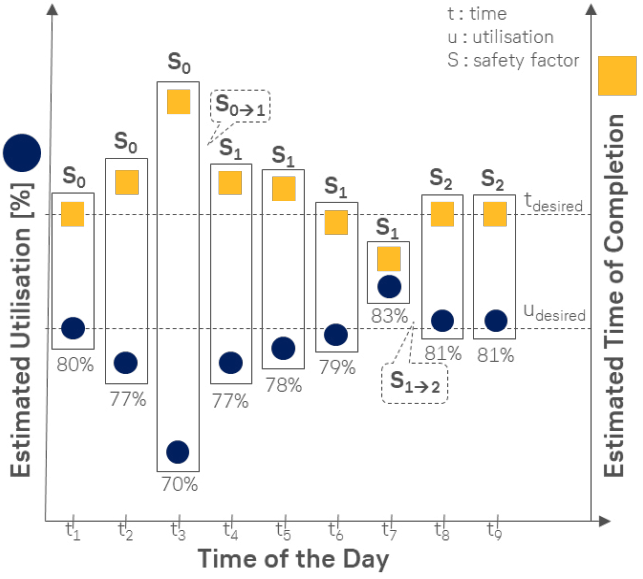

Fig. 5. Operative Adjustment of Dispatching Strategies (1/2)

Readjusting S to a higher value at t_2 increases the terminal utilisation and therefore reduces as well the estimated time of completion for the terminal. On the other hand, the amount of early arrivals is increased as well (Fig. 6). The simulation forecast enables the evaluation of this trade-off in order to keep target values within acceptable levels.

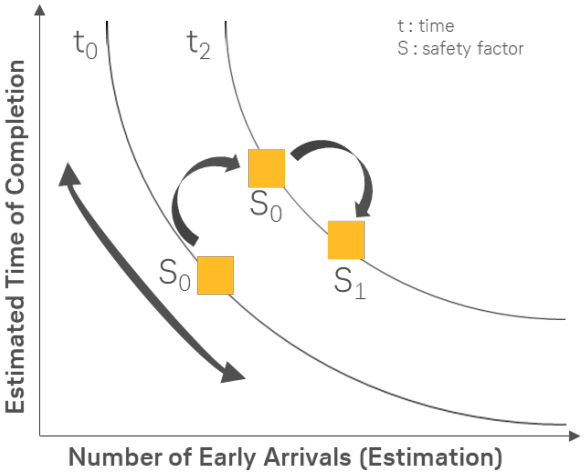

Fig. 6. Operative Adjustment of Dispatching Strategies (2/2)

6. CONCLUSION & OUTLOOK

In this paper, concept and implementation of a Digital Twin for dispatching assistance in port logistics have been described. The Digital Twin continuously evaluates the current dispatching policy as well as configuration alternatives by providing a performance forecast based on the current system status. The implemented dispatching algorithm is based on the concept of starvation avoidance, adapted to the specific use case in order to balance the probability of overstock and shortage situations at the system's bottleneck according to operator preferences. The simulation was developed using Python SimPy and it is deployed as a cloud service that is invoked following the occurring events in the physical system, which are captured by IoT-connected sensors. This architecture enables seamless integration into port operations while at the same time minimising the effort of infrastructure administration.

Planned extensions of the Digital Twin include the integration of a time-slot management system for incoming vehicles and the consideration of a limited buffer area capacity (extending the optimisation problem by a multi-echelon dimension). Furthermore, the execution speed of the simulation could be improved by using e.g. a containerised PyPy version, possibly yielding an acceleration of factor 6-7 (van der Haam, 2018). Alternatives include the use of a compiled language implementation of a discrete event simulation package (SimJulia 2018). Another often-reported difficulty when creating simulation models comes from the high manual modelling effort, which is required to build large-scale simulations. Methods for automatic model generation try to mitigate this, and the extension of our current model creation workflow by integrating process-mining tools such as ProM could be a promising feature (van Dongen, et al. 2005; Kovalchuk et al. 2018).

REFERENCES

Aurich, P.; Nahhas, A.; Reggelin T.; Tolujew, J. (2016): Simulation-based optimization for solving a hybrid flow shop scheduling problem. Proceedings of the 2016 Winter Simulation Conference. In: *Simulating complex service systems*. Piscataway, NJ.

Aviatar (2018): AVIATAR – Our innovative and holistic platform for the entire aviation industry [online] https://www.aviatar.com/ [21.12.2018]

Dagkakis, G.; Papagiannopoulos, I.; Heavey, C. (2016): ManPy: an open-source software tool for building discrete event simulation models of manufacturing systems. In: *Softw. Pract. Exper*. 46 (7), p. 955–981. DOI: 10.1002/spe.2347.

Donhauser, T.; Ebersbach, T.; Franke, J.; Schuderer, P. (2018): Rolling-reactive Optimization of Production Processes in a Calcium Silicate Masonry Unit Plant Using Online Simulation. In: *Procedia CIRP* 72, p. 249–254. DOI: 10.1016/j.procir.2018.03.266.

Dream (2018): Dream [online] https://github.com/nexedi/dream [21.12.2018]

Fernandes, N.O.; Thürer, M.; Silva, C.; Carmo-Silva, S. (2017): Improving workload control order release: Incorporating a starvation avoidance trigger into continuous release. In: *International Journal of Production Economics* 194, p. 181–189. DOI: 10.1016/j.ijpe.2016.12.029.

Glassey C.R.; Petrakian, R.G. (1989): The use of bottleneck starvation avoidance with queue predictions in shop floor control. Proceedings of the 1989 Winter Simulation Conference E.A. Berkeley, CA 94720, U.S.A.

Grieves, M.; Vickers, J. (2017): Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In: Kahlen et al.: *Transdisciplinary Perspectives on Complex Systems*, Bd. 89. Cham: Springer International Publishing, p. 85–113.

Gutenschwager, K.; Fauth K.-A.; Spieckermann, S.; Voß, S. (2000): Qualitätssicherung lagerlogistischer Steuerungssoftware durch Simulation. InformatikSpektrum. Technische Universität Braunschweig.

Hofmann, W.; Langer, S.; Lang, S.; Reggelin, T. (2017): Integrating Virtual Commissioning Based on High Level Emulation into Logistics Education. In: *Procedia Engineering* 178, p. 24–32. DOI: 10.1016/j.proeng.2017.01.055.

Hofmann, W.; Ulrich, J.H.; Lang, S.; Reggelin, T.; Tolujew, J. (2018): Simulation and Virtual Commissioning of Modules for a Plug-and-Play Conveying System. In: *IFAC-PapersOnLine* 51 (11), p. 649–654. DOI: 10.1016/j.ifacol.2018.08.392.

Jula, H.; Dessouky, M.; Ioannou, P. (2006): Truck route planning in nonstationary stochastic networks with time-windows at customer locations. Pennsylvania State University - Harrisburg, Middletown, PA.

Kahlen, F.-J.; Flumerfelt, S.; Alves, A. (Hg.) (2017): Transdisciplinary Perspectives on Complex Systems. Cham: Springer International Publishing.

Kovalchuk, S.V.; Funkner, A.A.; Metsker, O.G.; Yakovlev, A.N. (2018): Simulation of patient flow in multiple healthcare units using process and data mining techniques for model identification. In: *Journal of biomedical informatics* 82, p. 128–142. DOI: 10.1016/j.jbi.2018.05.004.

Kritzinger, W.; Karner, M.; Traar, G.; Henjes, J.; Sihn, W. (2018): Digital Twin in manufacturing: A categorical literature review and classification. In: *IFAC-PapersOnLine* 51 (11), p. 1016–1022. DOI: 10.1016/j.ifacol.2018.08.474.

Kunath, M.; Winkler, H. (2018): Integrating the Digital Twin of the manufacturing system into a decision support system for improving the order management process. In: *Procedia CIRP* 72, p. 225–231. DOI: 10.1016/j.procir.2018.03.192.

Lange, A.-K.; Branding, F.; Schwenzow, T.; Zlotos, C.; Schwientek, A.; Jahn, C. (2018) Dispatching Strategies of Drayage Trucks at Seaport Container Terminals with Truck Appointment System. Hamburg University of Technology, 21073 Hamburg, Germany. Published In: *Dynamics in Logistics Proceedings*, Bremen, Germany.

Lödding, H. (2013): Handbook of Manufacturing Control. Berlin, Heidelberg: Springer Berlin Heidelberg.

Mönch, L.; Fowler, J.W.; Mason, S.J. (2013): Production Planning and Control for Semiconductor Wafer Fabrication Facilities. New York, NY: Springer New York (52).

Muckstadt, J.A. and Sapra, A. (2010): Principles of Inventory Management. New York, NY: Springer New York.

Pinedo, M.L. (2016): Scheduling. Cham: Springer International Publishing.

Rosen, R.; Wichert, G.; Lo, G.; Bettenhausen, K.D. (2015): About The Importance of Autonomy and Digital Twins for the Future of Manufacturing. In: *IFAC-PapersOnLine* 48 (3), p. 567–572. DOI: 10.1016/j.ifacol.2015.06.141.

Salabim (2018*):* Salabim - discrete event simulation in Python [online] https://github.com/salabim/salabim [21.12.2018]

SimJulia (2018): A discrete event process oriented simulation framework written in Julia [online] https://github.com/BenLauwens/SimJulia.jl [21.12.2018]

Simpy (2018): Simpy - Bitbucket [online] https://bitbucket.org/simpy/simpy/ [21.12.2018]

T'Kindt, V.; Billaut, J.-C. (2006): Multicriteria Scheduling. Theory, Models and Algorithms. Dordrecht: Springer-Verlag Berlin and Heidelberg GmbH & Co. KG. [online] http://gbv.eblib.com/patron/FullRecord.aspx?p=259385.

United Nations (2017): Review of Maritime Transport 2016. United Nations Conference on Trade and Development. United Nations, p.1.

van der Ham, R. (2018): salabim: discrete event simulation and animation in Python. In: *JOSS 3* (27), S. 767. DOI: 10.21105/joss.00767.

van Dongen, B. F.; Medeiros, A. K. A. de; Verbeek, H. M. W.; Weijters, A. J. M. M.; van der Aalst, W. M. P. (2005): The ProM Framework: A New Era in Process Mining Tool Support. In: Hutchison et al.: *Applications and Theory of Petri Nets* 2005, Bd. 3536. Berlin, Heidelberg: Springer Berlin Heidelberg, p. 444–454.

Weingartner, E.; Vom Lehn, H.; Wehrle, K. (2009): A Performance Comparison of Recent Network Simulators. In: 2009 *IEEE International Conference on Communications*. Dresden, Germany, 14.06.2009 - 18.06.2009: IEEE, p. 1–5.