DOI: 10.22032/dbt.57890

Simulation in Produktion und Logistik 2023
Bergmann, Feldkamp, Souren und Straßburger (Hrsg.)
Universitätsverlag Ilmenau, Ilmenau 2023
DOI (Tagungsband): 10.22032/dbt.57476

Automatic Generation of a Simulation Model to Support the Rescheduling of a Fixed-Layout Assembly System

Automatische Generierung eines Simulationsmodells zur Unterstützung der Umplanung einer Baustellenmontage

Harold Billiet, Rainer Stark, TU Berlin, Berlin (Germany), harold.billiet@tu-berlin.de, rainer.stark@tu-berlin.de

Abstract: Fixed-Layout Assembly (FLA) systems are used to assemble large and bulky products. These products are often unique and require customer-specific engineering and customization. FLA systems are frequently prone to disturbances and plan deviations throughout operations: delayed deliveries, incompatibility or failures of equipment, and unplanned absences of operators. Planners therefore need a simple and efficient tool to quickly forecast the impact of changes on the whole assembly system. A solution concept has been presented by the authors in a previous publication (Billiet and Stark, 2022). The authors presented a method to automatically generate a simulation model by using data concerning the products, orders and shifts from the ERP system. This paper describes the implementation of the previously presented solution concept by applying it to a FLA for the production of Large Motors and Converters (LMC) in Berlin.

1 Introduction

Fixed-Layout Assembly (FLA) systems are used to assemble large and bulky products such as ships, aircraft, locomotives, and large machinery (Guo et al., 2020). These Products are often unique and require customer-specific engineering and customization (Lotter and Wiendahl, 2012). FLA systems are frequently prone to disturbances and plan deviations throughout operations: delayed deliveries, incompatibility or failures of equipment, and unplanned absences of operators (Matt et al., 2015). These problems are due to specific characteristics of FLA systems, such as the small production volume and the high individuality and flexibility they offer to customers. Recent crises such as the Pandemic as well as military conflicts have aggravated these disturbances. Supply chains are less stable, and the availability of employees is less reliable due to illness (Allam et al., 2022). Planners currently use their experience to forecast the impact of these disturbances, as well as to compare different rescheduling options as a reaction. This often leads to errors and unexpected situations that can spread to the whole system over time (Qian et al., 2020).

Planners therefore need a simple and efficient tool to quickly forecast the impact of changes on the whole assembly system. A solution concept has been presented by the authors in a previous publication (Billiet and Stark, 2022). This paper presented an approach using Discrete-Event Simulation (DES) to analyse the impact of changes on the assembly system. However, the high variability and flexibility of FLA systems makes them difficult to model in a DES software. For that reason, the authors presented a method to automatically generate a simulation model by using data concerning the products, orders and shifts available in the ERP system.

This paper describes the implementation of the previously presented solution concept by applying it to a FLA for the production of Large Motors and Converters (LMC) in Berlin. The first part of the paper will focus on summarizing the theoretical aspects presented in the previous publication. The next section will focus on the FLA system for LMC. The results of an analysis that examined every commonly occurring problem as well as every rescheduling possibility will be presented. The following part of the paper will focus on the software architecture used for the implementation of the solution concept.

2 Previous work

This section summarizes the theoretical aspects of the implementation based on the author's previous publication (Billiet and Stark, 2022).

2.1 Fixed-Layout Assembly Systems

The high flexibility and variability of Fixed-Layout Assembly (FLA) systems leads to constant changes and disruptions during operations. Scheduling problems include delayed deliveries of components, unplanned absence of operators, and unavailable resources (Qian et al., 2020). Problems can also occur during the assembly operation: reachability issues, collisions, and compatibility of resources. In addition to these problems, FLA planners also have to deal with last-minute customer-requirements.

At this time, planners cannot precisely predict the effects of these changes and disruptions on the whole assembly system. They currently use their experience to guess these effects, which often leads to unexpected situations (Qian et al., 2020). Planners therefore need simple and effective methods to support the rescheduling of FLA systems during operations.

2.2 Discrete-Event Simulation for Fixed-Layout Assembly Systems

Discrete-Event Simulation (DES) excels at analysing inflexible and automated systems and is today widely spread in those areas. Highly variable and flexible systems are, on the other hand, hard to model using traditional DES methods. FLA systems strongly depend on the current customer-specific orders that need to be assembled. They do not have a specific layout or machine arrangement that could be optimized (Billiet and Stark, 2022).

Using DES for the operational scheduling and decision-making of FLA systems could be an interesting approach. However, a huge drawback to this approach would be the complexity and time requirement necessary to model the current situation every time a decision needs to be made. This time-consuming task contrasts with the need for planners to quickly compare solutions when reacting to a problem.

Classic DES modelling techniques include the verification and validation of the model. Verification ensures that the model is consistent and free of errors. This includes programming mistakes as well as runtime errors during the simulation. Validation, on the other hand, makes sure that the simulation model actually runs like the real system (Mayer, G. et al. 2020).

2.3 Solution Concept

The authors presented a solution concept for a method to quickly support decision-makers for the rescheduling of FLA systems by using DES and process modelling. The idea is to use data from the ERP system concerning current customer orders and planning of resources to generate a DES-model based on the assembly processes of the products. Relevant data as well as a framework are presented in the author's pervious publication (Figure 2 and 3 of the publication: Billiet and Stark, 2022).

Relevant data is imported and consolidated into an XML file by an algorithm. This data concerns orders, assembly operations, resources, parts, workers, and shifts. It is then imported and displayed in a user interface to enable the configuration of the simulation experiments that are going to be analysed. This information is then sent to a model generator, which uses it to generate a model in a DES-software. The simulation experiments are then carried out, and the results are sent back and displayed on the user interface. This method enables any user with basic IT-knowledge to quickly configure and run simulation experiments in a friendly user interface.

3 Use-Case: Large Motors and Converters

3.1 Presentation of the Use-Case

Large Motors and Converters (LMC) are machines that reach several meters and weigh up to 25 tons. Every product requires customer-specific engineering and is produced only once. The factory consists of several assembly stations, where a product can be fully assembled (see Figure 1). Resources and tools (also referred to as assets) are universal and can be shifted between the stations. Every assembled part is manufactured by an external supplier.

Depending on the product, it can take up to three days for a team of two workers to fully assemble a motor. Scheduling is planned by using Microsoft Excel: for every shift, orders are assigned to workers. This planning is then used to approximate when products will be fully assembled.

Figure 1: 3D-Model of the FLA for Large Motors and Converters

3.2 Requirements

3.2.1 Occurring Problems that need to be Analysed

The following list displays common errors ranked starting with the most frequent one:

- Supplied part was not delivered / will be delivered with delay
- Worker becomes unavailable for a specific time
- Assembly operation cannot be executed (collisions, reachability...)
- Last minute customer requirements
- Supplied part is faulty
- Resource needs to be repaired / is not available for a specific time

Part deliveries are the most frequent problems, especially because suppliers often prioritize bigger and more frequent orders instead of low volume customer specific orders. In the case of faulty parts, the issue also translates into a delayed delivery since the part needs to be sent back to the supplier. Worker availability became an important issue during the Covid-19 Pandemic. Even though it did not only concern FLA systems, it especially affected them because of the high specialisation and know-how required for the assembly of the product. Problems during operations happen when parts cannot be assembled because of feasibility issues. Because of the low production volume, it is often not possible to verify in detail if every new part could produce such issues. A rare issue is the availability of resources or assembly equipment, for example in the case of maintenance or repair.

3.2.2 Rescheduling Scenarios

When reacting to an error, planners have limited options. The most common option would be to change the current shift plan available in Microsoft Excel. By doing so, planners can prioritize other orders by changing the worker assignments of the next shifts. They can also add new worker assignments, and add shifts, for example during the night.

Another, more complex, option would be to change the sequence in which the assembly operations need to be carried out. By doing so, planners can prioritize specific assembly operations compared to other, that for example require a specific part that was delayed, know-how that currently is not available, or assets that are currently already in use. This option is complex because planners need to be sure that the operations are still going to be feasible after altering the assembly sequence.

3.2.3 Required Functions for the Implementation

The implemented solution is operated through a user-interface, in which a decision-maker configures and analyses simulation experiments. A map of the required functions of this user-interface has been created based on the current problems and possible reactions of the factory (See Figure 2).

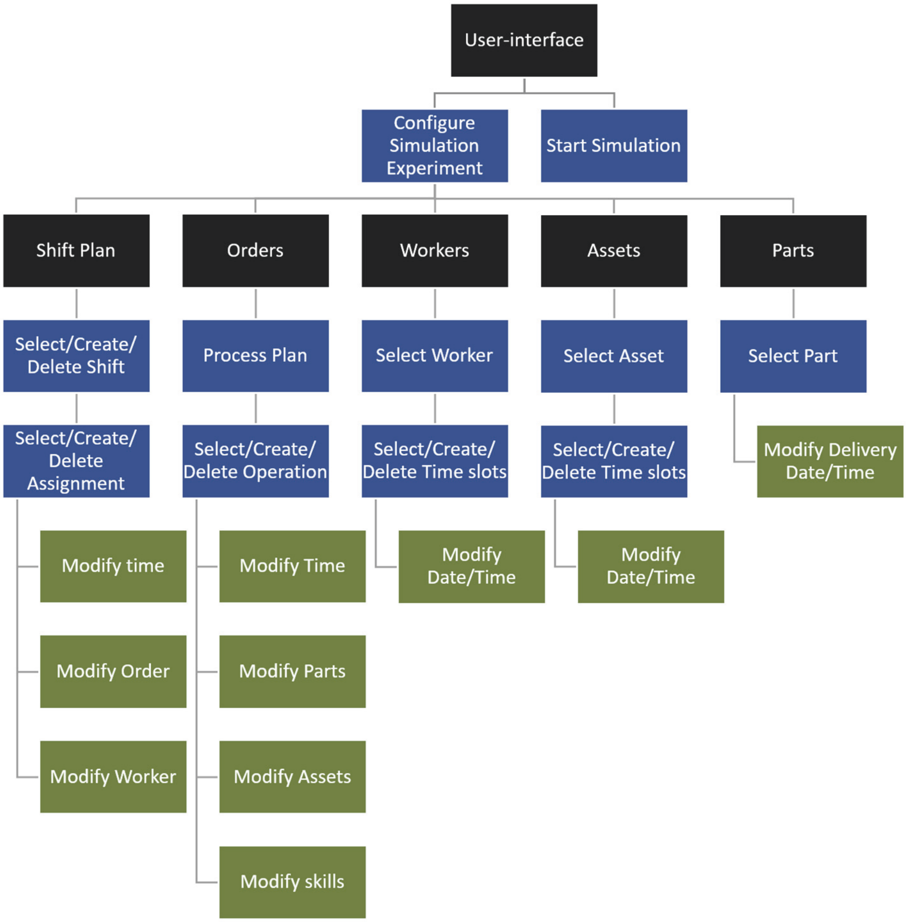

Figure 2: Required Functions of the User-Interface featuring Tabs (black), Buttons (blue) and User-Input (green)

This simulation experiment configurator enables planners to analyse the impact of problems that frequently occur during operations. For example, in the case of a delayed delivery, the decision-maker can select the concerned part and modify its delivery date and time. In the case of a last-minute customer requirement, planners would be able to select the corresponding process plan and then add a new assembly operation. A second option for planners would be to analyse the impact of new scheduling scenarios. After learning about a delayed delivery, planners could for example change the current shift assignments to another order to minimize unnecessary waiting times.

After starting a simulation experiment, the user interface will display the simulation results. These results are generated by the simulation model and transferred back to the user-interface. The authors gave more information about the simulation results in their previous publication (Billiet and Stark, 2022).

4 Implementation of the Solution Concept

4.1 Software Architecture

The implemented solution uses a web browser as a user-interface for the configuration of simulation experiments. This method enables lots of customization possibilities since it uses the same widespread programming techniques as websites. It also allows the possibility in the future to run the whole implementation on a server and access the user-interface through the web. The complete software architecture of the implementation is shown in Figure 3.

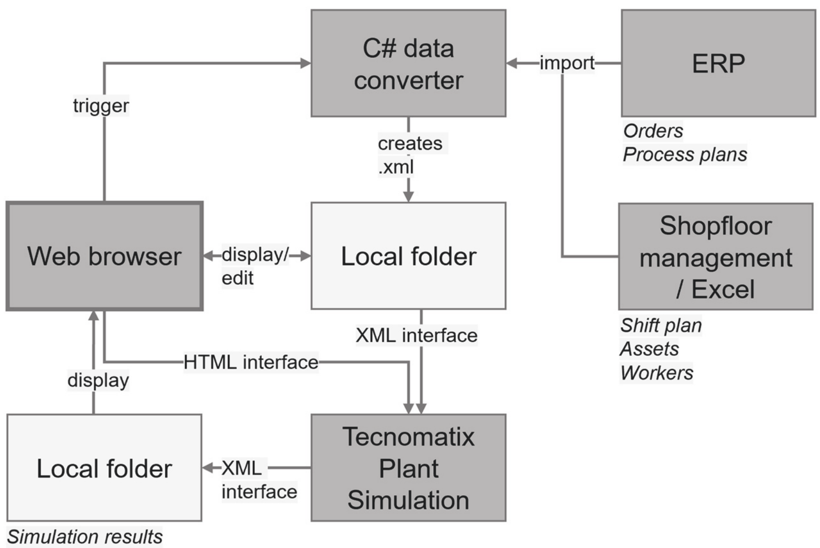

Figure 3: Software Architecture of the Implemented Solution based on the publication of Billiet and Stark (2022)

The user-interface triggers a C# algorithm, that imports data from the ERP system, as well as Microsoft Excel to convert it into a standardized local XML file.

This XML file is then imported into the user-interface to display the current state of the system. Everything including shift plans, orders, operations, assets, and workers is displayed. The user-interface enables the modification of these elements according to its required functions (See Figure 2). This represents the configuration of a simulation experiment. The modified version of the input data is then saved locally to be accessed by the simulation.

Tecnomatix Plant Simulation (version 16.1) uses the XML interface to access that local file. After running the simulation, it uses this interface again to export the results in a standardized local XML file. This allows the web user-interface to access and display the data once it has been generated.

4.2 Implementation in Tecnomatix Plan Simulation

Plant Simulation is a widespread DES tool that allows lots of customization and expansion. It uses its own programming language (*SimTalk*), to allow the creation and execution of algorithms before, during and after a simulation experiment.

In its normal state, the implemented simulation model is empty. When the simulation is started, *SimTalk* algorithms are triggered to import data using the *XMLInterface* element. This data is stored into various tables and lists in the simulation model. Another series of algorithms is then executed to use this data to generate the simulation model, see Figure 4 of Billiet and Stark (2022).

The algorithm generating the model strongly relies on the possibility of using *SimTalk* to create new simulation elements such as *Stations*, *Connectors* and *Exporters* in the simulation frame. This is done by creating a new class in the *.UserObjects* folder, in this case a modification of the *Station* class, which is going to represent a single assembly operation. This new class is then created on the simulation frame for every operation of every order existing in the imported data using the *.duplicate* method. After duplicating the element, its various attributes such as the position (*.Coordinates3D*), assembly time (*.ProcTime*), and required assets and parts (custom attribute) are updated.

Assets and workers are then generated in the simulation as *Exporters* with a single *Service*. Thus, operations can only be processed if the required *Services* are currently available.

Parts are also modelled using *Exporters*. Since every part has a unique part number, each part *Exporter* is only used once during a simulation experiment.

The shift plan and part delivery behaviour are modelled using an algorithm that activates or deactivates the specific *Exporter*. After the generation of the simulation elements, an algorithm uses the imported shift plan and part delivery tables to define when the *Exporters* will be active. This is done using the *.ExecuteIn* method. For example, if a part will be delivered in three days, then an algorithm activating that *Exporter* will be executed in three days of the simulation time. The *.ExecuteIn* method adds events to the list of discrete Events that are being processed during the simulation experiment.

After the complete generation of the model and its events, the simulation is started without any animation to maximize computational speed. At the end of the simulation, an algorithm gathers the results (delivery reliability, workload, etc.) and exports them into an XML file using the *XMLInterfaceExport* element. Every generated element is then deleted. Since the model generator does not include any stochastic behaviour in the simulation model, no simulation replication is required.

For this implementation, the verification started during the development of the generator. By using lists and tables that can be displayed in plant Simulation, it was possible to verify and validate the data importing algorithms. Break points in the code, as well as console printing have also been used to ensure that algorithms were properly executed at the expected time. Varying the input data also made sure that the algorithms were functional.

The implemented solution in Plant Simulation thus allows the complete generation of a model and its events, as well as starting the simulation and exporting its results only by pressing the *Simulation Start* button.

4.3 Implementation of the Web-based User-Interface

The central element of the solution is the user-interface. It is coded using React, an open-source JavaScript library. This library uses JSX, an extension to the JavaScript language syntax that integrates HTML elements in the code. For the current implementation, it is especially useful since it allows the generation of elements in the user-interface depending on the imported data from the ERP and shopfloor management systems. A screenshot of the user-interface is shown in Figure 4.

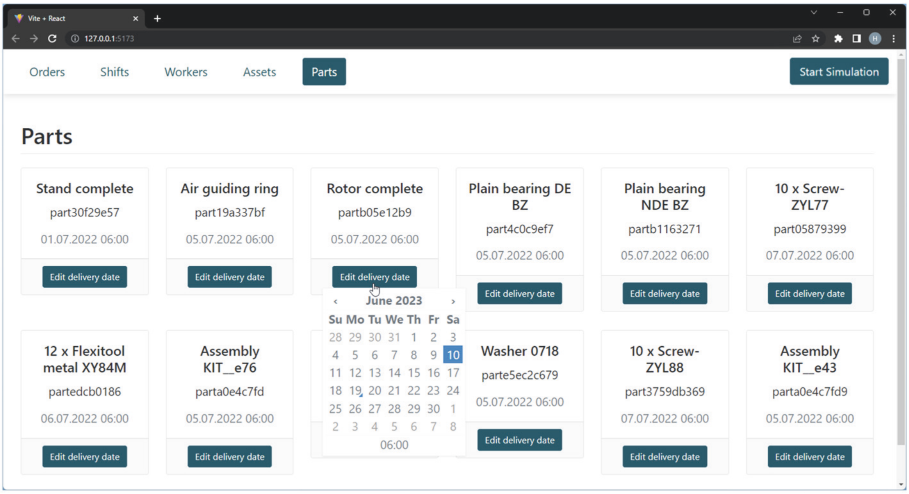

Figure 4: Implemented Web-based User-Interface for the Configuration of Simulation Experiments

The user-interface currently displays all the parts that were imported in the XML file. Every part is represented as a card with its number, description, and delivery date. In the case of a delayed delivery, the cards also feature a button that opens a date-time picker (See Figure 4, Rotor complete).

After configuring a few changes, the simulation can be started. This is made possible using the button on the right, which opens a link using the HTML interface of Plant Simulation. The used call is *SC_CallMethod.* It allows the execution of a specified method in Plant Simulation from a web browser. Note that Plant Simulation needs to be started in server mode. The results are then displayed in a new browser tab.

5 Conclusion

This paper presented the implementation of a solution concept published in a previous paper, with a use-case of a FLA system for LMC. A short description of the use-case is given, as well as a list of the required features for the solution concept (see Figure 2). This list was created by examining the cases that are going to be analysed by the simulation: disruptions and rescheduling scenarios.

The implemented solution concept uses Tecnomatix Plant Simulation for the generation and simulation of the model, together with a web-based user-interface for the configuration of the simulation experiments. Plant Simulation is started from the user-interface, runs in the background, and sends the simulation results back to the browser. Data is imported from the ERP system as well as from Microsoft Excel using a C# algorithm, that saves it locally in a standardized XML file. Users without IT or simulation knowledge are thus able to configure and analyse simulation experiments. The validity of the generated simulation model is currently dependant of the input data.

The next steps would be the integration of optimization algorithms for the automatic calculation of the best possible scheduling and rescheduling scenarios. Reinforcement learning could, for example, be a promising approach. In this case, it could also be useful to run the simulation on a server to minimize computational time. This would also facilitate the usage of the solution concept since it no longer would require a local installation of Plant Simulation and connection to the ERP and shopfloor management systems.

References

Allam, Z.; Bibri, S.E.; Sharpe, S.A.: The Rising Impacts of the COVID-19 Pandemic and the Russia–Ukraine War: Energy Transition, Climate Justice, Global Inequality, and Supply Chain Disruption. Resources 11 (2022) 11, pp. 99.

Bergmann, S.; Stelzer, S.; Strassburger, S.: Initialization of simulation models using CMSD. Proceedings of the 2011 Winter Simulation Conference (WSC), Phoenix, AZ, USA (2011), pp. 2223–2234.

Billiet, H.; Stark, R.: A Framework for Rescheduling a Fixed-Layout Assembly System Using Discrete-Event Simulation. 2022 Winter Simulation Conference (WSC), Singapore (2022), pp. 1762–1771.

Guo, D.; Lin, P.; Lyu, Z.; Ling, S.; Li, M.; Huang, G.Q.; Rong, Y.: Towards Assembly 4.0: Graduation Intelligent Manufacturing System for Fixed-position Assembly Islands. IFAC-PapersOnLine 52 (2019) 13, pp. 1513–1518.

Guo, D.; Zhong, R.Y.; Ling, S.; Rong, Y.; Huang, G.Q.: A roadmap for Assembly 4.0: self-configuration of fixed-position assembly islands under Graduation Intelligent Manufacturing System. International Journal of Production Research 58 (2020) 15, pp. 4631–4646.

Heilala, J.; Montonen, J.; Jarvinen, P.; Kivikunnas, S.; Maantila, M.; Sillanpaa, J.; Jokinen, T.: Developing simulation-based Decision Support Systems for customer-driven manufacturing operation planning. Proceedings of the 2010 Winter Simulation Conference, Baltimore, MD, USA (2010), pp. 3363–3375.

Lotter, B.; Wiendahl, H.-P.: Montage in der industriellen Produktion. Berlin, Heidelberg: Springer Berlin Heidelberg 2012.

Matt, D.T.; Rauch, E.; Dallasega, P.; Vidoni, R.; Spena, P.R.: Synchronisierung von ETO-Fertigung und Baustellenmontage. Zeitschrift für wirtschaftlichen Fabrikbetrieb 110 (2015) 1-2, pp. 9–13.

Mayr, T.: Softwarebasierte Evaluierung menschlicher Arbeit in der industriellen Baustellenmontage. Unter Mitarbeit von Sebastian Schlund und Patrick Rupprecht, 2021,

Negahban, A.; Smith, J.S.: Simulation for manufacturing system design and operation: Literature review and analysis. Journal of Manufacturing Systems 33 (2014) 2, pp. 241–261.

Qian, C.; Zhang, Y.; Jiang, C.; Pan, S.; Rong, Y.: A real-time data-driven collaborative mechanism in fixed-position assembly systems for smart manufacturing. Robotics and Computer-Integrated Manufacturing 61 (2020), pp. 101841.