system, this work focuses on aggregating data for DS. In this work, DS is implemented using a simulation-based scheduler. The DS system is connected to the AAS of the production system through a service that queries data from the AAS. Communication with the DS framework is realized via its API. The presented approach is validated using a virtual production system for precast concrete modules consisting of parallel circulation systems as a proof of concept. This paper will answer the following research questions: *How can data collected during production be used to optimize production scheduling?* and *How can a service for dynamic scheduling be integrated into an I4.0-based production system*?

## 2. State of the Art

Although automation approaches are available for most precast concrete module production processes, efficiency still needs to improve in the precast concrete production industry. It is essential to digitize the production process for automation to improve efficiency. Reichenbach and Kromoser (2021) note that digitization approaches are needed to interconnect production processes along the value chain. As Cheng et al. (2023) suggest, the precast concrete industry has to transition from experience-based manual production to data-driven automation. In the manufacturing industry, the digital transformation toward data-driven production is referred to as I4.0. The term is a reference to the upcoming fourth industrial revolution, which will lead to a convergence of the physical and virtual worlds by integrating advanced technologies such as Artificial Intelligence (AI) and the Internet of Things (IoT) (Monostori et al., 2016). The main goal of I4.0 is to create smart factories that are more efficient, flexible, and responsive to market method is often used as a digital basis for linking and integrating other technologies (Wang et al., 2020). However, BIM lacks the semantic integration of sensor and IoT data as well as real-time capability (Boje et al., 2020).

In I4.0, the DT is a key concept for capturing and providing data along the value chain (Uhlemann et al., 2017). In construction, a DT is generally understood to be a digital model going beyond a building information model through the sensor and IoT data integration and enabling data exchange between the digital model and its physical counterpart via a data link (Boje et al., 2020). BIM and the DT, however, are not mutually exclusive but complementary and respond to different industry requirements (Davila Delgado and Oyedele, 2021). When implementing DTs, the construction industry is still at an early stage. Up to now, there have mainly been isolated solutions based on different architectures and not readily compatible with each other. In previous work (Kosse et al., 2023), we have proposed a modular approach based on information containers where data is captured use case-related and exchanged via standardized interfaces enabling a flexible implementation of DTs. We introduced the AAS from I4.0 and Information Container for Linked Document Delivery (ICDD) as examples. In addition, to support the automated production of precast concrete modules, we have proposed (Kosse et al., 2022) presented an AAS-based DT for precast concrete modules containing production-relevant data stored in submodels, each representing a closed view of one aspect or use case. Standardized interfaces enable seamless data access, and the DT can be extended throughout all phases of the module s lifecycle. However, the paper is limited to the component level. The extension to the production system level and the use of data for data-based decision-making have yet to be considered. Only a few works exist on the implementation of dynamic scheduling. Villalonga et al. (2021) propose a decision-making framework based on DTs for DS of cyber-physical production systems (CPPS). The framework includes a DT of the CPPS with a decision-making module that generates scheduling decisions based on real-time data (i254(.)]TJET

software platform for implementing I4.0 solutions providing tools and libraries for developing and integrating smart manufacturing systems that can connect to and interact with other systems and devices.

### 4.2 Dynamic Scheduling Framework

A production system made of parallel circulation systems can be classified as a parallel flow shop. However, since the heat treatment process takes significantly longer than other processes, parallel heat treatment stations are needed to achieve even utilization rates of the stations. This feature makes the circulation system a flexible flow shop (FJS). The simulation-based scheduler is implemented using Sim4FJS, an open-source Python framework for simulating flexible job shops built upon the discrete-event simulation framework *Simpy*. Since flow shops are a special case of job shops, Sim4FJS is well suited to model the production of precast concrete modules. However, circulation systems critically differ from job shops, consisting of chained stations with limited or no buffer between them. Since product carriers can only move to the next station after the previous carrier leaves the pr0 G[75 cvous ctation the pimulatinn podu17(e)-5(s)-4( )-482(p)-4(

optimization run is triggered with another API call. After completion of the optimization process, the optimized schedule is queried by the service and converted back into the appropriate format for the AAS. Finally, the production schedule is transmitted to the AAS via its HTTP/REST interface. The schedule data from the production system AAS is distributed over the actors in the production system, such as products, machines, and processes, which are all represented through AASs. However, this step is not the focus of this work and will be discussed in detail in future research.

Figure 3: Sequence diagram of data exchange between AAS and optimization framework by a service.

## 5. Verification

In this section, we validate the proposed framework using a virtual production system consisting of two parallel circulation systems, each consisting of one workstation for forS46.48 TmTmTm(a)-5 can be expanded or reduced by adding or removing production units and jobs and adjusting the amount of data being collected.

Table 1: Processing times (in hours) of precast concrete modules according to Du et al. (2021).

| **Part Type** | **Formwork** | **Reinforcement** | **Concrete** | **Curing** | **Stripping** |
|---|---|---|---|---|---|
| **1** | 0.5 | 1 | 0.4 | 7 | 0.5 |
| **2** | 2 | 2 | 0.5 | 7 | 1 |
| **3** | 1.5 | 1.5 | 0.5 | 7 | 1 |
| **4** | 1.5 | 2 | 0.5 | 7 | 1 |
| **5** | 0.5 | 1 | 0.2 | 7 | 0.5 |
| **6** | 0.7 | 2 | 0.5 | 7 | 0.6 |
| **7** | 2 | 2.5 | 0.5 | 7 | 1.5 |
| **8** | 0.5 | 1 | 0.3 | 7 | 0.4 |
| **9** | 3.5 | 3 | 1 | 7 | 1 |
| **10** | 1.5 | 1.5 | 0.5 | 7 | 1 |

Figure 4: Gantt chart of the optimized production schedule.

## 6. Conclusion

This paper presents an approach to integrating dynamic scheduling into an I4.0-based production system. The proposed method employs a DT to aggregate data collected at the product and machine levels. This data is available to an optimization framework through a