Conceptualization and development of a standardized framework for virtual commissioning of modular assembly control systems

CHRISTIAN ARNET
Conceptualization and development of a standardized framework for virtual commissioning of modular assembly control systems

Author: Christian Arnet
Supervisor KTH: Gunilla Franzén Sivard
Examiner KTH: Lars Wingård
Supervisor AUDI: Sebastian Mayer
Submission Date: 06.10.2019
I confirm that this master thesis is my own work and I have documented all sources and material used.

Stockholm, 06.10.2019

Christian Arnet
Preface

This master thesis is the final part of the two-year master program in Production Engineering and Management at the KTH Royal Institute of Technology, Stockholm. The project is equivalent to 30 ECTS and was performed from February to August 2019. The project has been carried out in cooperation with AUDI AG at their headquarters in Ingolstadt, Germany. The involved department is responsible for the planning support of powertrain manufacturing.
Acknowledgments

First, I would like to thank my supervisor at AUDI AG, Mr. Sebastian Mayer, for providing me with the opportunity to work on this research project and his support and assistance throughout the thesis work. Further, the department I/PA-17 headed by Dr. Carl Hans Dickhaus receives my appreciation for the support during the project. I also want to thank my academic supervisor from KTH, Gunilla Franzén Sivard, for her guidelines and inspirational discussions. Finally, I would like to thank my family for their endless love and support during my studies.
Abstract

Today’s industries are challenged by an ongoing trend of more product variants to meet differing customer needs. This leads to efficiency losses within traditional production systems using fixed cycle times within an assembly line. Contrarily, flexible modular assembly systems are seen as a promising solution. These are characterized by offering flexible routes of the products throughout the production system. Utilizing the capabilities of cyber-physical production systems, automated guided vehicles are performing the transports of workpiece-carriers in-between decoupled workstations. In order to manage the complexity, sophisticated control approaches are required to coordinate all involved entities. These are organized either in a centralized or decentralized manner. For the performance evaluation of both types of algorithms concerning parameters characterizing the factory layout, no standardized framework exists.

Following the principles of virtual commissioning, this work presents a framework composed of three elements: a flexible simulation model representing a modular assembly system, a multi-agent system incorporating the logic of centralized or decentralized control approaches, and an interface for data exchange.

The framework has been successfully validated by incorporating a centralized approach following a global schedule and a decentralized approach using a negotiation-based agent-system which controls the production flow.
Sammanfattning


Ramverket har framgångsrikt validerats genom att jämföra ett centraliserat tillvägagångssätt som följer ett globalt schema samt ett decentraliserat tillvägagångssätt som med hjälp av ett förhandlingsbaserat agent-system kontrollerar produktionsflödet.
# Contents

Preface ......................................................... iii

Acknowledgments ........................................ iv

Abstract ................................................... v

Sammanfattning ........................................... vi

1 Introduction ............................................... 1
   1.1 Context .................................................... 1
   1.2 Problem formulation .................................... 3
   1.3 Scope, aim and research questions ...................... 4
   1.4 Thesis work methodology ................................ 4

2 Literature review ......................................... 6
   2.1 Characteristics of the modular assembly ............... 6
   2.2 Production control approaches for modular assemblies . 10
      2.2.1 Centralized control approaches .................. 11
      2.2.2 Decentralized control approaches ................. 13
   2.3 Concept of Digital Factories ............................ 14
      2.3.1 Development phase .................................. 14
      2.3.2 Commissioning ....................................... 15
      2.3.3 Production run ...................................... 18
   2.4 Service-Oriented-Architecture .......................... 19
   2.5 Key Performance Indicators of a production system ... 20
      2.5.1 Job-related Key Performance Indicator (KPI)s .... 21
      2.5.2 Workstation-related KPIs .......................... 21
      2.5.3 Production System-related KPIs .................... 22
3 Requirements of the tool suite 23
  3.1 Simulation model ........................................ 23
  3.2 Control logic ............................................ 24
  3.3 Data-exchange interface .................................. 27
  3.4 User-interface ............................................ 27

4 Tool sets for the development of the framework 29
  4.1 Simulation model ........................................ 29
  4.2 Intelligent and distributed system ......................... 30
    4.2.1 Hardware ........................................... 30
    4.2.2 Programming environment ............................ 31
    4.2.3 Communication within the IDS ....................... 33
    4.2.4 Multi-Agent-System ................................ 34
  4.3 Data-Exchange Interface ................................ 35

5 Technical implementation 36
  5.1 PlantSim model ........................................ 37
  5.2 Multi-Agent-System ..................................... 40
    5.2.1 Active Agents ..................................... 41
    5.2.2 Passive agents ..................................... 44
  5.3 TCP-Socket-connections ................................ 46
  5.4 Network configuration of involved sub-systems ............ 50

6 Validation 52
  6.1 Centralized approach .................................... 52
  6.2 Decentralized approach .................................. 56
  6.3 Evaluation of validation experiments ....................... 59

7 Discussion on the developed framework 61
  7.1 Reflection on research questions ........................ 61
  7.2 Utilization of the framework ............................. 63
  7.3 Limitations and challenges ................................ 65

8 Conclusion and future work 66
  8.1 Conclusion ................................................ 66
Contents

8.2 Future work .......................................................... 67

A Control approach implementations ................................ 68
  A.1 Centralized approach .............................................. 68
  A.2 Decentralized approach ......................................... 70

List of Figures ......................................................... 73

List of Tables ........................................................ 75

Acronyms ............................................................... 76

Bibliography ............................................................. 78
1 Introduction

When Henry Ford introduced the assembly lines for the production of his Model T, he revolutionized the standards of factories. Specialization in tasks and fixed orders of operations were characterizing this production system for the next decades [1]. With further improvements considering the philosophy of Toyota’s Lean Production, factories were more and more focusing on an efficient line balancing, takt times and pull-principles [2]. However, indications show that today’s production systems shift towards more flexible and agile approaches. This chapter introduces the context, problem and the scope of this master thesis:

1.1 Context

Today’s factories are affected by several trends. The increase of product variants due to the possibility of letting the customer configure his or her product is driving the complexity that a production system has to manage. Especially the automotive industry is a role model for this today. Utilizing mechanisms for standardization, such as a platform strategy or modularization, several derivatives can be assembled within a single line [3]. This approach, known as mixed-model assembly, requires tremendous efforts in production planning and control [4]. In order to still be able to meet the desired takt time of the line, all workstations have to be balanced with respect to their cycle times. If the assembly process of one specific variant takes more time than the other one, a waste of waiting is arising in the context of lean management. Fig. 1.1 visualizes this effect.
1 Introduction

According to studies from the management consulting firm McKinsey, the trend of an increasing amount of product variants in the automotive industry will continue [5]. Especially, when cars are equipped with different drivetrain technologies, such as internal combustion, hybrid-electric, battery-electric or fuel cell, the assembly processes differ considerably. Therefore, traditional assembly lines will struggle to achieve this kind of mixed-model support.

At the same time, another trend of digitization leads to the ongoing incorporation of intelligent systems into factories. Commonly known as Smart Factory or Industry 4.0, its four primary objectives are to improve processes, to increase capacity utilization, to reduce production costs, and to address specific customer wishes [6]. This trend considers the further introduction of Information- & communication technologies (ICT) and enables applications, such as condition monitoring, predictive maintenance, and human-machine-collaboration [7].

A third trend is the further utilization of Autonomous Guided Vehicle (AGV) in factories. Different types of AGVs can be distinguished based on the used tracking mechanism. In earlier stages, rails and inductive tracks were common [8]. Today’s state-of-the-art AGVs gained in flexibility using optical or digital tracks. At the production facility of Lamborghini for its Urus in Sant’ Agatha di Bolognesa, AGVs have replaced the conveyor belts to move car bodies from one assembly station to another [9].

The first trend introduced in this chapter is a challenge that can be mastered with the enabling technologies derived from the other two trends, digitization and AGV-incorporation. A production system built-on these principles is the so-called
modular assembly. It consists of workstations that are no longer connected with a conveyor belt. Instead, AGVs are conducting the transports in between them. This principle leads to specific flexibility, which does neither define the predecessor nor the successor of an individual workstation. Based on the product characteristics, the Assembly Priority Chart (APC) of a specific product variant leads to multiple routes through the production system. [10]

For assigning the jobs to available workstations, several different approaches exist. The theoretical concepts of both the modular assembly itself and the production control approaches, have been part of previous works. Their characteristics are further outlined in the literature review in chapter 2.

1.2 Problem formulation

Different approaches to derive the next decisions are focusing on improving other KPIs and are characterized by alternating features. A centralized approach, such as a global schedule, typically maximizes the output in perfect conditions. However, unforeseen modifications need to be considered, for example when a machine shutdown occurs or post-processing is required. Contrarily, decentralized approaches, such as bidding algorithms, excel in adaptability, but usually have lower utilization rates.

In order to fully understand the differences between approaches and the suitability to different factory layouts, a standardized framework for modeling and simulating modular assemblies is required. When companies are thinking about modifications and especially when changing traditional habits such as the production system, it is of outstanding importance to predict and confirm the output and other KPIs. With respect to the concept of virtual commissioning, a standardized tool suite is required to adjust it to the characteristics of the factory, to incorporate the control logic, and to evaluate the overall performance.
1.3 Scope, aim and research questions

The scope of this thesis is to identify the relevant systems to build up a standardized framework for modeling and simulating modular assemblies. The involved systems have to collaboratively make decisions on the next operations and support the concept of modular assembly systems. Additionally, a communication architecture on the shop-floor-level has to be defined, which is supporting the data flow requirements. Furthermore, the following research questions will be addressed in the thesis:

1. Which sub-systems are needed to evaluate and compare different control algorithms?
2. What are the data flow differences for both a centralized and a decentralized control approach?
3. Which communication protocol is useful to enable the data exchange between different cyber-physical entities?
4. Does the implemented system fulfill the defined requirements?

1.4 Thesis work methodology

To answer the introduced research questions, this thesis is sub-divided into the following parts: Chapter 2 presents a literature review to analyze the concept of the modular assembly. Furthermore, the two main different types of control approaches are introduced to outline the differences between centralized and decentralized decision processes. Additionally, the key enabler of this shift in production systems, the further incorporation of digitization, is described. Inclusively, the concept of virtual commissioning is introduced.

Furthermore, the requirements of the tool-suite are defined in chapter 3. In chapter 4, different tools that are feasible for the implementation were analyzed. The further implementation of the solution consisting of its sub-systems is presented in chapter 5. Afterwards, chapter 6 is validating the developed framework with exemplary instances of both centralized and decentralized approaches. In chapter 7, a discussion refers to the research question and describes possible use-cases and limitations of
1 Introduction

the developed solution. A summary and an outlook to further research topics is concluding this thesis in chapter 8.
2 Literature review

This chapter is presenting the results of the literature review and outlines the fundamentals of this master thesis. Firstly, it characterizes the concept of the modular assembly. Secondly, it introduces the two different classifications of control approaches of the modular assembly, which are either centralized or decentralized. Thirdly, it describes the rise of digital tools within the life-cycle of production, which includes the trend of virtual commissioning. Furthermore, it outlines the concept of Service-Oriented-Architecture (SOA). Lastly, this chapter introduces different KPIs to evaluate the performance of a production system.

2.1 Characteristics of the modular assembly

The modular assembly is a vision describing a production system that is characterized by its flexibility. In contrast to traditional assemblies using a conveyor belt to build up a product layer-by-layer by adding components in a fixed order of processes, the modular assembly uses AGVs to find alternating routes through the production system. Without the need for balanced processing times, the workstations are decoupled and independent from each other. [11, 12]

The typical layout is visualized in Fig. 2.1.
In his work, Kern introduced nine principles that characterize a modular assembly system. Those are summarized in table 2.1 and further outlined in the following: [10]

Table 2.1: Overview of the nine principles of modular assembly systems.

<table>
<thead>
<tr>
<th>Principle number</th>
<th>Principle label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principle 1</td>
<td>Variable order of assembly</td>
</tr>
<tr>
<td>Principle 2</td>
<td>Smart assignment of assembly stations</td>
</tr>
<tr>
<td>Principle 3</td>
<td>Variant-dependent assembly and cycle times</td>
</tr>
<tr>
<td>Principle 4</td>
<td>Flexible reaction to disturbances</td>
</tr>
<tr>
<td>Principle 5</td>
<td>Flexible reaction to product mix variables</td>
</tr>
<tr>
<td>Principle 6</td>
<td>Adaption to changes</td>
</tr>
<tr>
<td>Principle 7</td>
<td>Integrated transportation of cars and components by AGVs</td>
</tr>
<tr>
<td>Principle 8</td>
<td>Flexible integration of quality control loops</td>
</tr>
<tr>
<td>Principle 9</td>
<td>Adaption to assembly workers</td>
</tr>
</tbody>
</table>
Products that shall be assembled within the modular assembly need to support variable assembly orders to unleash its potential. Therefore, not all designs for assembly are useful. For example, the conventional sandwich strategy, which builds-up a product layer-by-layer, does not benefit the integration into a modular assembly system. However, having multiple alternatives for the next processes lead to varying routes through the production system (principle 1). In order to manage the resulting complexity, decisions regarding the following assembly stations are crucial for the performance. To optimize specific KPIs, jobs need to be assigned thoroughly to workstations (principle 2). Decoupled workstations lead to the main advantage that their individual processing times are not required to be balanced. Therefore, the modular assembly supports several variants with alternating cycle times (principle 3). Another advantage of decoupling compared with traditional assembly lines is that machine failures do not automatically lead to an interruption of the overall production. Contrarily, the remaining available workstations can continue with their operations, and already assigned jobs can be re-scheduled (principle 4). Additionally, the system can react flexibly to changes in the product mix (principle 5). Therefore, even short-term modifications in orders are possible to address. However, even long-term adjustments that typically affect the factory layout can easily be incorporated. For example, a further workstation can be added to the system without stopping the remaining stations (principle 6). Rigid conveyor belts and other transportation systems are replaced by AGVs to enable flexible route guidance through the production (principle 7). Some product variants might require more or different quality checks than others. Within modular assemblies, this can easily be implemented with adding quality stations that can be assigned to jobs in the same way as workstations (principle 8). Deviations in cycle times have already been explained with alternating product variants. However, another root cause for non-constant processing times is the different capability and experience level of assembly workers. Production control approaches in modular assemblies can consider this knowledge, when assigning jobs to workstations (principle 9).

To further differentiate the concept of modular assemblies, Kern distinguishes between product-based and process-based modular assembly systems [10]. The product-based modular assembly takes the product structure into account and assigns a sequential number of assembly steps corresponding to a set of components to a
workstation. Therefore, a workstation is selected, if a product variant requires a specific component. Contrarily, the process-based modular assembly primarily allocates skills to workstations. Examples for these skills are welding, screwing or the equipment of special tools. In both alternatives, several identical workstations can be used to perform the required processes. If workstations are able to allocate an operation to at least two stations, this is called partial operation flexibility by Kacem et al. [13]. In terms of having workstations that can perform each operation, this has been introduced as total flexibility. Furthermore, the APC of a product is of importance to describe flexibility. If the given options of a product’s APC allow alternative assembly orders, this has been introduced as order flexibility by Watermeyer [14]. Both types of flexibility lead to a summary of the modular assembly’s bottom line characteristic which is alternative routing through the production system.

Fig. 2.2 does exemplary visualize both types of flexibility. Fig. 2.2a displays a job consisting of four operations, which can be assembled in two different orders. These are defined in two process plans and is summarized in one APC, which belongs to the specific product variant. After the completion of operation 1, it can either be continued with operation 2 or operation 3. Furthermore, operation flexibility is incorporated for operation 3. It can be performed either at workstation $M_1$ or $M_2$. Both types of flexibility lead to four different possible routes through the production system which is displayed in Fig. 2.2b.
2 Literature review

2.2 Production control approaches for modular assemblies

In the previous section, several characteristics of the modular assembly were introduced. The flexibility of such a production system leads to the requirement in coordinating jobs, workstations, and AGVs. To address the full complexity, Mayer identified five decision categories that production control systems have to consider [16]. They are finding answers for the questions described as follows:

- **Job release management:**
  When is the best point in time to release a job, and which product variant fits preferably in the current state of the production system?

- **Job flexibility management:**
  Which route from the decomposed priority chart of a product variant should be
selected? Which operation should be performed on which workstation?

- **Workstation management:**
  In the case of having several jobs being assigned to one workstation: Does it make sense to re-arrange the order of jobs, for instance, to reduce the number of tool changes?

- **Vehicle management:**
  Which AGV shall execute the transport from a workstation to the successor?

- **Deviation management:**
  What is the best reaction of a production system on deviations, such as extended transport times, machine shut-downs, etc.?

Approaches for an effective production control can be classified in being centralized or decentralized. Their difference is introduced in the following:

### 2.2.1 Centralized control approaches

A typical way of assigning jobs to workstations is the utilization of scheduling algorithms that take all introduced decision categories into account at a single instance. Before the production run is initiated, a global schedule is generated based on the type and amount of products being produced in the designated period of time. The global schedule consists of the assignments of all involved operations from all products to the most suitable workstations and also includes the AGVs for the required transports. An exemplary global schedule is displayed in Fig. 2.3.
Due to the high amount of dependencies, a lot of computational power is required for the optimal distribution of jobs. Therefore, different approaches use heuristics to reduce the time needed to develop a good solution. In previous work, Shao et al. presented a genetic algorithm that is conducting the assignments. Out of all possible work distributions, it selects and re-configures the most promising assignments with respect to a fitness-function. Hence, re-combinations of the jobs to workstations while planning occur with the objective of improving the overall performance of the schedule. [17]

While the production is running, the entities of the production system are following the introduced sequences. Whenever dynamic deviations are detected, such as machine down-times, the schedule has to be adapted. Recalculations of the overall schedule are very time-extensive and, therefore, not a very efficient reaction. Van Brackel [18] has introduced a predictive-reactive approach. It consists of a second schedule that is calculated whenever deviations occur. It uses dispatching rules to avoid stopping production. However, since this does not provide very efficient assignments, a further optimized schedule taking the deviations into account is generated. As soon as it is available, the entities are following the optimized one. Another approach for taking dynamic deviations into account was presented by Niehues et al. [19]. It utilizes means of improvement to adjust the current schedule to the consequences of the deviations. If further adjustments are needed, a new schedule is calculated using genetic algorithms. Due to limitations in real-time-assignments, a quick solution is prioritized over quality.
2 Literature review

The characteristics of centralized control approaches are summarized as follows:

- very good results in total output due to its holistic composition in perfect conditions
- high computational requirements to derive schedules
- difficulties in deviation management due to long reaction times

2.2.2 Decentralized control approaches

Contrarily to solving a global optimization problem by a single instance, the decentralized control approach is addressing the required assignments by local decision-makers. Those are responsible for minor categories and try to reach their individual objectives.

Several approaches using agent-based assignments can be found in the literature. Those agents can represent the physical components of the production system. In approaches by Parunak et al. [20] and Shen and Norrie [21], job and machine agents interact with each other in the form of negotiations.

Gankin [16] developed a negotiation-based bidding process in which workstations are offering time-slots that can be requested by jobs in the form of placing a bid. The value of the bid is calculated using reinforcement learning techniques to improve the quality of assignments over time. Therefore, decisions are collaboratively made by individual agents by focusing on a common goal, which is minimizing the total production makespan.

In another approach using machine-learning techniques, Waschneck et al. [22] developed a set of agents using reinforcement learning to find solutions collaboratively. Each agent is optimizing the rules of one work center in line with a deep neural network with respect to the actions of others.

The characteristics of decentralized control approaches are summarized as follows:

- lower output rates due to the missing holistic view
- fast decisions due to the distribution of computation
- good performance in deviation management
2.3 Concept of Digital Factories

The life-cycle of a production system can be subdivided into the development phase and the production run, which is initiated with the successful start-of-production. The transition in-between is the so-called process of commissioning. All phases are visualized in Fig. 2.4.

In recent years, a concept of Digital Factories raised awareness. It integrates different tools and methods from various levels of the life-cycle to thoroughly plan production processes and systems [23]. It utilizes models that represent physical resources and processes, and therefore, have to match the actual data. According to Kühn [24], the primary motivation is about increasing the planning quality, standardizing planning procedures, and reducing the time until the start-of-production. In the following sections, several approaches are introduced based on their position within the life-cycle.

2.3.1 Development phase

The development phase of a production system covers several hierarchical levels. A single workstation basically consists of two types of components: hardware and software, which need to be developed individually, but consistently. Collectively, they have to fulfill the process requirements that are derived from product characteristics. Therefore, changes in the product design affect the ongoing production development. The traditional way of addressing this problem is to avoid those late adjustments. Consequently, the product-development-process is completed with a product-freeze, before the production workstation development is initiated. [25]

This procedure leads to a long time-to-market and inflexibility to unforeseen needs of modification. With the concept of Digital Factories, these work-flows are parallelized...
High transparency between different groups of responsibility is required to generate collaboration between teams independently from their location and their organizational unit. This can be achieved with a constant exchange of digital representations of the individual solutions. Further tools can keep track of the consequences of adjustments. Therefore, interferences between different developments are detected at an early stage. [27]

On the next levels, several workstations can be combined with further handling tools to cells and lines. The complexity of managing the interoperability on these levels increases when different suppliers of machine tools are involved.

2.3.2 Commissioning

The next step within the life-cycle of production equipment is the commissioning phase. This term describes the verification of the behavior of an automation system concerning the interaction of the control program with the corresponding hardware [28]. Traditional commissioning is performed with already installed equipment inside the factory. However, to reduce the ramp-up time, several functionality tests can be performed using virtual representatives. Auinger [29] introduced three alternative approaches which are referring to Virtual Commissioning (VC). They are visualized in Fig. 2.5 and further described in the following:
1. **Soft commissioning:**
   This approach executes the control logic on a virtual controller, which is connected to the physical hardware of the automation system.

2. **Hardware-in-the-loop commissioning:**
   Contrarily, this approach is using the real control unit which is connected with a virtual representation of the hardware.

3. **Software-in-the-loop commissioning:**
   This approach is the purest virtual commissioning variant since both the control unit and the hardware are virtual representations.

The increased utilization of VC-applications has been initiated with the validation of the control logic used for Programmable Logic Controller (PLC)s that are widely used within automation technology at machine level [30]. In this level, Hardware-in-the-loop commissioning methodologies are connecting the involved PLC to a virtual representation of the hardware components of a machine. Depending on the commands being sent, specific manipulations of the machine model will occur. The modifications can be evaluated with respect to the expected outcome. Using this approach, the functionality of the developed logic can be evaluated at an early stage. With further successful implementations, VC has also been introduced on other levels.
within the automation pyramid. For example, at process level, Qin et al. compared environments for combining multiple specialized control systems to collaboratively perform machining processes with respect to the kinematics and dynamics of the processes [31]. Pritschow introduced machine models that can be combined with each other to simulate the behavior of cells with respect to each element to avoid collisions [32]. With the focus on Manufacturing Execution System (MES) considering the plant level, Heinrich [33], Römberg [34], and Zhang [35] have presented approaches on using discrete event and material flow simulation.

For successfully incorporating the methodology of software-in-the-loop, Zäh has named three involved steps for VC: [36]

1. **Hardware model:**
   The system, which is using the control logic, such as on machine, cell or plant level, needs to be developed in a virtual environment. The model has to be in line with the actual system that will use the same control logic after commissioning. The requirements are divided into two main categories. Firstly, it has to follow the physical properties that can be summarized in geometry and kinematics. Secondly, the behavior of such a system also needs to be included into the model. Furthermore, the model can also be used for visualization purposes.

2. **Control System:**
   The program consisting of the control logic is the system under test. With VC, its functionality has to be confirmed. Therefore, it uses the same information model and deriving-decision-processes as the fully-developed system.

3. **Interface:**
   In order to forward necessary signals from one system to the other, a consistent data flow has to be enabled. This includes the commands being sent from the control system to the hardware model. The communication in the other direction consists of feedback with the purpose of further consideration in the decision processes.

In recent years, the utilization of VC-applications has gained in importance, which results in the benefit that developers can further focus on optimizing and testing the
control programs by using digital models of the production equipment. This leads to multiple advantages which are the reason for VC becoming a standard in automated production plants [37, 38, 39]:

- Increasing the level of robustness, quality, and maturity of the control programs
- Less traveling to the factory during the ramp-up phase for the involved software developers
- Earlier initialization of the implementation phase for new production plants and earlier Start-of-Production (SOP)
- More verification tests are feasible leading to higher optimization capabilities
- Higher transparency for the plant owner to check the software’s capabilities with respect to his requirements
- Verification of new design changes or adjustments as a result of the continuous improvement process

2.3.3 Production run

During the life-cycle phase production run, the primary motivation for utilizing digital tools is in controlling the behavior of the production system [40]. This can be subdivided into three factors of influence. Firstly, it is in increasing the process control and, therefore, to reduce potential quality problems. This can be addressed with surveillance tools that continuously collect process data. For example, vibration sensors can be placed on the machine tool to gain insight into the dynamics of the process. Typically, the concept of storing those data sets is called 'Digital Shadow' [41]. This historical data can be utilized for predictions about the future process quality. Therefore, if abnormalities within the process are detected at an early stage, countermeasures can be taken to avoid expensive post-processing.

Secondly, digital tools can be used to increase cooperation between the planning and the production departments. Especially within the concept of ‘continuous improvement,’ the planning engineers should interact with the shopfloor in close collaboration.
A digital twin is a model of a workstation, cell, or line, which is coupled with its physical representation [42]. The fundamental idea of this concept is that it is aligned with the characteristics of the actual system. Similar to the digital shadow, this is achieved by collecting data from the process. The differentiation between both concepts is that the digital twin is adaptable to adjustments. It can be used by the planning department to evaluate the modifications on the system [43]. Therefore, it is a beneficial concept to improve already existing systems with the support of a digital tool suite.

Thirdly, real-time adjustments within the production system are feasible with extended connectivity of all involved systems. This links typical hardware components, such as workstations, conveyor belts, or lifting devices with computing systems. The resulting entities connecting the physical with the digital world are called cyber-physical-entities which are collectively summarized as the Cyber-Physical-Production-System (CPPS). This involves the possibility of exchanging data with each other. Variations within the execution of processes can individually be detected and shared among the remaining entities. Therefore, the reaction to deviations is improved. [44] Well-functioning CPPS seem to play an outstanding role in future implementations of the modular assembly, because of its high demand in communication among the involved entities.

### 2.4 Service-Oriented-Architecture

Within CPPS, a transformation of the network communication is arising. Traditionally, the automation pyramid including its hierarchical layers defines a communication order with instructions from the top to the bottom layers, and the results the other way around. However, this is not feasible when entities collaboratively exchange data with the purpose of further optimization. [45] Therefore, a new concept of sharing data with each other is required. One approach is the so-called SOA. Initially, it is an approach for the development of distributed systems within IT applications. The software is partitioned in small units that can be executed with low-scaled computational power. [46] Other members can request each of these services within the architecture. This leads to ongoing requests for further processing data sets through standardized interfaces. The
The main advantage is its modular composition, which leads to high scalability. Another positive side-effect is that services can be offered with technological neutrality. As long as the interface corresponds with the standard, the services can be written in any programming language, protocol, etc. [47] The structure of SOA can be incorporated in developing CPPS. Each entity is a client that requires services from other clients. Therefore, it also has the advantages of scalability and flexibility. Instead of having stiff hierarchical communication orders, it is a loosely coupled network of entities and services. This shift within the communication is illustrated in Fig. 2.6. It is important to emphasize that the traditional systems within the automation pyramid will not lose their right of existence. The functionality of a MES or Enterprise Resources Planning (ERP) will still be required by the CPPS. However, services can specialize in a specific domain. [48]

![Figure 2.6: Within SOA, the hierarchical functions of the automation pyramid are distributed among entities and services. Source: Adapted from [45]](image)

### 2.5 Key Performance Indicators of a production system

In order to improve systems or to compare different approaches with each other, standardized KPIs can be calculated. The performance of a production system can be evaluated from many different perspectives. This section introduces KPIs focusing on the jobs, the workstations, and a holistic consideration. [49]
2.5.1 Job-related KPIs

Typically, job-related KPIs are dealing with the extent of value-adding processing, transportation, and waiting times:

- **Actual Order Execution Time (AOET):**
  The time needed for producing one unit from the start with the order release to its completion. It is also commonly known as throughput time.

- **Actual Production Time (APT):**
  The time in which the job is actually processed within the workstations, which only includes value-adding tasks.

- **Actual Transportation Time (ATT):**
  The time needed for conducting all transports of one job, including the time for loading and unloading.

- **Actual Queueing Time (AQT):**
  The time a job is waiting to go through a manufacturing process, i.e. queueing time at a buffer.

The relationship between these KPIs is the following:

\[
AOET = APT + ATT + AQT
\]  

(2.1)

2.5.2 Workstation-related KPIs

For workstations, multiple KPIs exist to evaluate the quality of their processes. However, in this overview, KPIs being useful for scheduling are introduced:

- **Availability:**
  Workstations usually require a certain amount of inspection and maintenance. However, down-time behavior can occur during the production run. The term availability describes the ratio of up-time to the planned busy time.

- **Allocation Efficiency:**
  Allocation Efficiency represents the ratio of the actual usage with respect to the total planned capacity of a workstation. The complementary ratio describes the unit downtime.
2.5.3 Production System-related KPIs

From a more holistic view, KPIs of multiple workstations or jobs can be combined to evaluate the overall performance of a line or factory:

- **Allocation Ratio:**
  This KPI computes the percentage of the actual busy time of all machines with respect to all involved AOETs.

- **Production Process Ratio:**
  It describes the efficiency of a production and is calculated by the percentage of APT among AOET for all jobs within a given time period.

- **Throughput Ratio:**
  This KPI describes the ratio of the actual throughput among the planned output.
3 Requirements of the tool suite

Based on the insights from the literature review and further analyses, the requirements of a framework to address the question of control approach performance have to be defined. It primarily has to follow the concept of modular assembly systems and needs to be flexible enough to enable the incorporation of varying control approaches. Centralized and decentralized control approaches have, by definition, different requirements in exchanging data and in the process of deriving decisions. However, it needs to be standardized enough to allow meaningful comparisons. The framework shall consist of a tool-suite that follows the principles of VC. Following the three steps described by Zäh in chapter 2, the full framework will consist of three sub-systems. Those are ‘simulation model’, ‘control logic’ and ‘interface’. Furthermore, the user needs to be incorporated by a ‘user-interface’.

3.1 Simulation model

The simulation model needs to represent a virtual factory that is based on the characteristics of a modular assembly system. It shall consist of decoupled workstations that can be accessed by AGVs. In order to be aligned with the physical role model, it has to consist of workstations representing the internal material flow with AGV-(un-)loading stations, in- and outbound buffers, and processing blocks. Fig. 3.1 represents the sub-systems of a workstation.

Figure 3.1: Visualization of the material flow within a workstation used in modular assembly.
3 Requirements of the tool suite

Since the processing times at workstations depend on the product variant and designated operation, it has to be adjusted for every process performed on each workstation. This tool-suite shall also be used for evaluating the impact of modifications in the factory layout. Changes in the number of workstations, workpiece-carriers, buffer sizes AGVs shall be realized quickly. Therefore, the simulation model has to be modular and scalable.

The simulation model is planned in excluding the necessary intelligence of assigning jobs to workstations. Contrarily, it is only used for incorporating the external decisions from the control system. Therefore, it is primarily used for visualizing external decisions. In order to derive decisions externally, specific sets of data from the model might be needed outside. Therefore, data about the current state has to be exported. In evaluating the performance of control approaches, production planning engineers are interested in the reaction of the system on occurring, unexpected problems, such as machine downtimes. Therefore, the model shall have the possibility to simulate downtime behavior.

Overall, the tool-suite should be able to calculate production-related KPIs. Therefore, a data-log is required for collecting all information being relevant for the introduced KPIs in chapter 2.5.

Therefore, the requirements of the simulation model are summarized as follows:

- virtual representation of the actual material flow within a modular assembly
- quick configuration and adaption to multiple parameters
- exporting triggers to ask for decisions
- importing decisions and updating the simulation model accordingly
- simulation of down-time
- data-logging of events for KPI-calculation

3.2 Control logic

The primary task of the control logic is to derive the decisions that are required to assign jobs to workstations and transport tasks to AGVs. The functionality of
the control logic is determined by the approach being evaluated. An analysis of different control approaches outlines the data-flow differences between centralized and decentralized approaches. Within the centralized approach illustrated in Fig. 3.2, the five decision categories introduced in chapter 2, job release management, job flexibility management, workstation management, vehicle management and deviation management are addressed by a single instance. Based on several input parameters, such as orders, process plans, and skills, this algorithm computes an optimal global schedule. The results will be forwarded to the entities of the production system, which are workstations, AGVs, and jobs. A loop-back from workstations ensures the reaction to downtime-occurrences.

![Data-flow within centralized control approaches](image)

Contrarily in the decentralized control approach, which is displayed in Fig. 3.3, the decision categories are distributed among the involved entities and collectively addressed. Therefore, it is necessary to have a system consisting of multiple intelligent and distributed entities. The involved entities represent the workstations, workpiece-carriers, and AGVs. Another entity introduced as coordinator consists of the provided input parameters. Collectively unified in the form of an Intelligent Distributed System (IDS), they are able to make decisions collaboratively. Based on the collective results, each entity can further create its individual local schedule.
In order to address both approaches, the framework is required to support the different needs of each principle, such as one-directional commands and bi- or multi-directional request and -response messages. The decentralized approach requires flexible communication among the involved entities. Therefore, the concept of SOA introduced in chapter 2.4 shall be used for the architecture of the entities. Each entity can offer different services that other entities can request. These services can be adapted to specialties of the specific control approaches being evaluated with this framework.

In the physical factory, the entities are most likely to communicate using Wireless Local Access Network (WLAN) and are physically separated from each other. Therefore, the IDS should follow the same principles to be in line with these characteristics. Furthermore, the number of involved entities should be flexibly adjustable to evaluate different parameters of the factory layout. Because of expected messaging traffic of high density, low latency times, and the avoidance of lost data within the IDS have to be considered throughout the implementation.

Therefore, the requirements of the control logic are summarized as follows:

- entities representing hardware components from the factory or information systems being used for decision processes
- IDS based on the principles of SOA
3 Requirements of the tool suite

- quick configuration and adaption to characteristics of the control approach being evaluated
- quick adjustments to parameters of the physical factory, such as number of workstations and AGVs
- physical separation of entities and communication over WLAN
- low latency times

3.3 Data-exchange interface

The sub-systems simulation model and control logic are dependent on each other in terms of providing each other with relevant data. Therefore, a standardized interface for bi-directional data exchange is required. Because of physical separation, a local network has to be established to enable communication of the involved sub-systems. The protocol used for generating the data exchange has to minimize the latency-times and to avoid data losses, which would result in malfunctioning of the overall framework. Furthermore, both endpoints of the interface have to ensure that the incoming data is forwarded to the right object (within the simulation model) and entity (within the IDS).

Therefore, the requirements of the data-exchange interface are summarized as follows:

- continuous messaging in both directions
- focus on stability and latency times
- forwarding of messages to the designated recipient

3.4 User-interface

Setting up both the simulation model and the IDS requires multiple parameters that influence the overall system performance. To reduce the complexity for the
user, the settings should be adjusted from one central user-interface which affects all sub-systems. This procedure is visualized in Fig. 3.4.

Figure 3.4: Centralized parameter settings lead to an adjustment in all sub-systems.

Besides building-up the experiment, the user-interface should also be used to provide a continuous surveillance of the production system. Even though the simulation model is used for the visualization of the decisions, the user-interface could summarize the most essential points. Therefore, the requirements of the user-interface are summarized as follows:

- graphical user-incorporation for single-point adjustments
- real-times notifications about the production system
4 Tool sets for the development of the framework

For deriving solutions that fulfill the requirements described in chapter 3, several tool sets were analyzed for further usage in this project. The key findings are introduced using the same distinction of the sub-systems ‘simulation model’, ‘control logic’ referring to IDS, and ‘Data Exchange Interface’ in the following:

4.1 Simulation model

As outlined in chapter 3.1, the simulation model is used to request and to visualize decisions from the control logic. In order to notify about specific events occurring in the simulation, a Discrete Event Simulation (DES) can be used. It is characterized by the process of codifying the behavior of a complex system as an ordered sequence of well-defined events [50]. In this definition, an event is seen as a change in the state of the system at a specific time.

A prevalent DES-environment is Technomatix PlantSim (PlantSim), which is distributed by Siemens. It is using the concept of object-orienting programming, has the possibility to code methods in its own language ‘SimTalk’, and simulates downtime behavior with a fixed seed of random numbers. By the last point, standardized comparisons of arbitrary situations are possible. Additionally, pre-built interfaces enable data exchange with external databases, files, and applications. [51]

Alternatively, a python package SimPy can be used to generate events. It is available under the MIT-license. However, it is not equipped with a Graphical User Interface (GUI) to visualize the current states of the production system. All events are printed on the console. [52] Another DES-alternatives can be found in Anylogic and
ExtendSim. However, because of its status as a standard application for modelling and simulation, it was decided to use PlantSim within this project. By this decision, it is ensured to increase the acceptance of this tool-suite further. At the time of developing the framework, AUDI was using the professional license of PlantSim V14.2. A further package called ‘interface’ was also used within this project.

4.2 Intelligent and distributed system

The control system will be developed following the principles of SOA and will be distributed among multiple entities. Therefore, the IDS is divided into its hard- and software.

4.2.1 Hardware

The requirement from the control logic of having physical entities communicate with each other leads to the consideration of using Single Board Computer (SBC). They are characterized by a simplified structure having all components attached to a single circuit board. They are widely used in industrial applications in such as metrology or embedded systems. Further usage of SBC can be found in the private environment, for example, in home automation or media centers. Schools and universities are using them as demonstrators for computer science courses. [53]

Although many different variations are offered, the basic structure consists of a microprocessor, a Random Access Memory (RAM), and input/output-panels. Instead of a hard-drive disk, a slot can accept a SD-card on which the operating system is installed. The significant difference between SBC and conventional computers is the size. Besides the mainboard, extensions such as touchscreens, are also offered by retailers. In this project, the three alternatives ‘Raspberry Pi 3 Model B+’ [54], ‘CubieBoard5’ [55] and ‘BananaPi M3’ [56] were taken into consideration. Their comparison is summarized in table 4.1:
For this project, it was decided to select the Raspberry Pi 3 Model B+, because of its outstanding cost-benefit-ratio. Furthermore, it has the most extensive customer base and a helpful community. Therefore, a lot of support can be found for projects developing applications using it.

### 4.2.2 Programming environment

A programming tool that fulfills the requirement of a distributed intelligent system can be found in Node-RED or Robot Operating System (ROS). Both can be installed on SBC having installed the operating system Raspbian, which is based on Debian and optimized for the Raspberry Pi hardware.

**Node-RED**

IBM provides an open-source development tool which is based on Node.js and commonly known as Node-RED. It is widely used for Internet of Things (IoT) applications and addresses the topic of data fusion. Each application is based on flows which relate to a specific sequence of nodes that are wired together. Each node can be seen as a template of a program which is configured individually. The in-between
connection of nodes is realized with messages that are based on Java Script Object Notation (JSON). Typically, a message consists of a topic, a payload, and a timestamp. Its graphical browser editor increases the transparency among the application. Fig. 4.1 shows a simple flow implementation. It consists of a node to inject data, a second one to process it and the third node to store it. [57]

Figure 4.1: Node-RED is a graphical programming environment for developing application flows.

Node-RED has a tremendous community support, which continuously increases the library of nodes. For IoT-projects, nodes for communication are relevant. Node-RED supports multiple protocols, such as MQTT, OPC-UA, and TCP-IP. Furthermore, it consists of pre-built connectors for external databases like mySQL.

**Robot Operating System**

ROS is a framework developed by researchers from Stanford University and published with the Berkeley Software Distribution - license. It is used for developing re-usable software modules for autonomous robotic systems. ROS is characterized by five basic principles: [58]

- **peer-to-peer:**
  The robotic system can be distributed among multiple computers that are connected via ethernet or WLAN.
4 Tool sets for the development of the framework

- **multilingual:**
  ROS supports many different programming languages. Its modules can be developed within Python, C++, and LISP. A standardized interface concludes the data exchange in-between the systems.

- **tool-based:**
  Standardized modules are provided by ROS to execute and manage basic functions.

- **lean:**
  The philosophy of ROS-modules is to re-use its algorithms. Therefore, dependencies between hardware-functionalities and software-libraries are avoided.

- **open-source:**
  By its broad and free availability, ROS is a common framework for small development projects. This led to a large user-community in recent years that supports each other with idea generation, improvements, and debugging.

4.2.3 Communication within the IDS

For coding the control logic in the introduced distributed manner, a standardized communication protocol for the efficient data exchange is required. One possibility is Message Queuing Telemetry Transport (MQTT). It was developed with the purpose of an extremely lightweight publish/subscribe messaging transport and excels in applications, in which a small code footprint and low network bandwidth usage is required. Especially in handling high latency or bad network connections, MQTT can fully exploit its advantages. [59]

The concept of publishing and subscribing is illustrated in Fig. 4.2. It primarily differentiates from the traditional client-server architecture, which consists of point-to-point connections, by decoupling the involved endpoints. In-between each other, a broker is established that is collecting all incoming messages and forwarding them to the corresponding subscribers.
The aspect of decoupling consists of three points: [61]

- **Space decoupling**: It is sufficient to know the Internet Protocol (IP)-address of the broker, instead of the other involved endpoints.

- **Time decoupling**: The broker can save messages to forward at a later point in time.

- **Synchronization decoupling**: Program sequences do not need to be interrupted while a message is published or subscribed.

These characteristics, in combination with its very-low latency times, make it selected for the data exchange within the IDS.

### 4.2.4 Multi-Agent-System

A Multi-Agent-System (MAS) is defined as a ‘loosely coupled network of problem-solving entities that work together to find answers to problems that are beyond the individual capabilities or knowledge of each entity’ [62]. Each agent is based on a computerized system and reacts flexibly and independently within the network to reach its predefined goals. Jennings introduced three characteristics of agents: [63]

- **Situated**: Agents interact with their environment by receiving input from the surrounding. Furthermore, they can process the data and share it among other agents.
4 Tool sets for the development of the framework

- **Incomplete information**: Each agent only has knowledge about a small unit of available information. By collaboration, the MAS can find answers collectively.

- **Autonomy**: Within given constraints due to programming, agents can make their own decisions without human support. Thus, a global system of control is not required.

- **Flexibility**: Agents are striving for their individual goals by cooperating with other agents. Therefore, they have to react flexibly to changes in the surroundings.

4.3 Data-Exchange Interface

For efficient data exchange in-between the sub-systems simulation model and IDS, socket-connections can be used. A socket is defined as an end-point to enable data-exchange between two applications that are executed within the network. The connection is established between a client and a server. Each socket is identified by an IP address and a port number. When the connection is established between both sockets, the client is using the socket to communicate with the server. For the transport of data, the Transmission Control Protocol (TCP)-protocol can be used. It belongs to the fourth layer within the OSI Basic Reference Model. [64] TCP-connections are characterized by the following attributes which make it suitable within this framework:

- Data packages arrive in the same order as they have been sent.

- Continuous checks for duplicate data will discard them when detected.

- Re-transmission is issued when no arrival-notification is sent.
5 Technical implementation

This chapter is introducing the different systems that have been developed to enable virtual commissioning of scheduling and control algorithms within a modular assembly. The technical solution that fulfills the requirements presented in chapter 3 is following the same distribution of sub-systems. Fig. 5.1 shows the composition of the tool-suite. The simulation model is generated with the simulation environment 'Tecnomatix Plant Simulation 14.2' (PlantSim), the IDS is realized by a MAS implementation within Node-RED and executed on a stack of Raspberry Pis, and the interface between both systems is established with TCP-socket-connections.

Figure 5.1: The framework consists of a simulation model running on a computer, a MAS executed by a stack of Raspberry Pis and TCP-socket-connections in-between.
5 Technical implementation

5.1 PlantSim model

As outlined in chapter 4.1, PlantSim is used for generating the simulation model in the form of a DES. On one hand, it is used to visualize the decisions from the MAS. On the other hand, it utilizes its events to trigger new calculations and requests. The model is divided into four major networks which are ‘Initializer’, ‘Transport-grid’, ‘Workstation’, and ‘Finisher’. Their roles and functions are outlined in the following:

- **Initializer:**
  The network ‘Initializer’ (see Fig. 5.2) consists of three sources to generate the movable elements, which are workpieces, workpiece-carriers, and AGVs. The items workpiece-carrier and AGV are going to circulate within the production system. Therefore, connectors for bringing these elements back to the starting point exist. Furthermore, a centralized parking lot for AGVs is implemented to store these items when they are not needed. Additional assembly stations

![Figure 5.2: The network 'Initializer' deals with the coordination of the release of movable elements.](image-url)
are used to load a workpiece on the workpiece-carrier and collectively on AGVs. When the workpiece is loaded on top of the workpiece-carrier, the job-release trigger is activated. Since the workpiece is connected to the workpiece-carrier throughout all operations, the workpiece-carrier temporarily represents a specific job.

- **Transport-grid:**
  The network 'Transport-grid' (see Fig. 5.3) consists of a standardized path-structure for AGVs that can be duplicated in both x- and y-dimension in order to build up blocks. A method is conducting the extension using information about the actual layout, which is derived from a configuration file. In addition to the tracks, it places an instance of the network 'Workstation' in the center of each block. Each workstation is named accordingly to its position, e.g. `MID_21` refers to the one being placed in the second row and first column.

![Figure 5.3: The Transport-grid connects all workstations with tracks used by AGVs.](image)
• **Workstation:**
The network 'Workstation' (see Fig. 5.4) is representing the material flow within an actual workstation. It starts with a docking station in which the workpiece-carrier is removed from the AGV. Furthermore, the carrier is placed inside an Inbound-Buffer of adjustable size before a working cell is performing the operation. This is followed by an Outbound-Buffer and another docking station for placing the workpiece-carrier on top of an AGV again. Additionally, a decentralized parking-lot for AGVs is part of the network ‘Workstation’ that can be used when an AGV is not assigned to a follow-up transport job. Whenever a workpiece-carrier arrives at a workstation, a method is called to request the information about the follow-up operation for this specific workpiece-carrier. Since the processing times vary for each product variant, each workpiece-carrier-object is implemented with a variable in which this information is stored. The method controlling the entrance logic of the processing block is extracting this data and adjusts the process time before the operation starts.

Figure 5.4: Workstations are representing the material flow of its physical role-model.

• **Finisher:**
The network 'Finisher' (see Fig. 5.5) is the last workstation within the production system. For the last time, the workpiece-carrier is removed from the AGV, and the disassembly of the workpiece from the carrier is performed. In the
simulation model, the workpiece is forwarded to a drain while the carrier is moving back to the ‘Initializer’-network to be re-used for the next order.

Figure 5.5: The network ‘Finisher’ consists of the dis-assembly of the workpiece from the workpiece-carrier.

5.2 Multi-Agent-System

Following the principles of SOA and CPPS, and the requirements from chapter 3.2, it was decided to distribute the control system among different agents to exploit its advantages of scalability and robustness. Further, each agent represents a cyber-physical-entity of the CPPS, such as a workpiece-carrier or workstation. This has the main advantage that the control system can easily be integrated into the real production system and that the MAS-dataflow is equivalent to the actual one.

The hardware of the MAS is realized by using multiple SBCs of the type Raspberry Pi Model 3B+. The software is programmed within the Node-RED environment and can be adjusted to the specific needs of the control approach being evaluated. Fig. 5.6 shows the composition of the MAS. Each Raspberry Pi executes a specific set of scripts to interact as an agent. In total, six different types of agents are in-use and are characterized by their sets of services. The communication in-between agents within the MAS is achieved using the publish-subscribe patterns of the MQTT-protocol. A message from one agent to the other represents the request of a service. By using a message broker in-between the different agents, it is possible to send data to the receiver without knowing its IP-address. This simplifies communication with respect
to scalability. The only information required is the topic that the agent has subscribed to. This is using logical assignments, including the type and id of the agent. The messages that are transferred from one agent to another are of the type JSON, which allows a compact storage of its data. The MAS is used to find scheduling solutions with respect to the decision categories presented in chapter 2. Depending on the approach being evaluated with the standardized framework, the decision categories are addressed by different agents and services.

The main differentiation is based on the extent of being involved in the process of deriving decisions. This leads to the distinction between active and passive agents. The roles and functions of all agents are outlined in the following.

### 5.2.1 Active Agents

The active agents are characterized by being directly involved in the decision processes. Therefore, the full problem is solved by dividing it into smaller pieces. By collaboration, the agents can collectively find good solutions. In total, four types of active agents work together. Those are ‘Central Coordinator’-Agent (CC-A), ‘Work-piece Carrier’-Agent (WPC-A), ‘Workstation’-Agent (MID-A), and ‘Transport’-Agent (Trans-A).
• ‘Central Coordinator’-Agent:
The CC-A stores a lot of information in its mySQL-databases that can be requested by other agents. This includes data about jobs to be released, assembly priority charts of product variants, and skill assignments to the workstations. The content of the data storages can be requested by other agents for collective decisions. The CC-A is designed to work as a single instance and therefore, it is unique. Its architecture is visualized in Fig. 5.7.

![Central Coordinator Agent diagram](image)

Figure 5.7: The CC-A consists of multiple data storages that are used within its services.

Its services depend on the control approach being virtual commissioned. However, they always follow the scheme of giving answers to specific requests from other agents. For example, in the case of a decentralized control approach, the CC-A receives a request on the next possible operations from a WPC-A. Therefore, it passes the results of the corresponding query based on the variant backwards.

• ‘Workpiece-Carrier’-Agent:
The WPC-A is assigned to one specific workpiece-carrier that corresponds to a digital representation within the simulation model. Therefore, multiple instances of this agent exist simultaneously. The linkage is identified with the unique ID being used in the model. Additionally, the WPC-A contains information about the workpiece being carried by it, such as the variant, a list of all already performed processes, and its current position in the factory. The variables and their data format is described in Fig. ??

Since the workpiece is attached to the workpiece-carrier throughout the whole assembly system until
it reaches the final station, the WPC-A is temporarily representing a specific job. Its services primarily deal with requests, for example about releasing which job and deciding the next operation.

Figure 5.8: The WPC-A digitally represents a workpiece-carrier inside the MAS that exists within the simulation model.

- ‘Workstation’-Agent:
The MID-A is representing an existing workstation. Therefore, the amount of this agent type used within the MAS depends on the design of the factory layout. Each MID-A stores data about its location, its availability, and the set of skills that the workstation is capable of offering. WPC-A subscribe topics that contain their individual Workstation ID or their offered skills. This principle is also introduced in Fig. ?? Its services depend on the extent of utilizing their knowledge. It varies between confirming their availability status and participating in negotiations by calculating individual bids.

Figure 5.9: Services from the MID-A can either be requested with a message topic consisting of the skill or the ID of the workstation.

- ‘Transport’-Agent:
The task of assigning AGVs to perform transports of workpiece-carriers from one workstation to another is executed by the Trans-A. Its service of selecting the best AGV can be requested by either WPC-A or MID-A. In order to be
able to derive these decisions, it has to keep control of the current positions of all AGVs. Therefore, this agent represents the fleet manager within the real production environment. Its functionality is summarized in Fig. ??.

Figure 5.10: The Trans-A assigns AGVs to transport jobs.

5.2.2 Passive agents

Furthermore, two passive agents are used in this framework. They are relevant to initialize the experiment by centrally setting-up involved parameters and to collect data for further evaluation purposes. They are called ‘User Interface’-Agent (UI-A) and ‘Data Collection’-Agent (DC-A).

- ‘User Interface’-Agent:
  
  In order to set up the framework correctly, the UI-A is used to incorporate the user and to adjust all required parameters centrally. This is sub-divided into three parts. Firstly, the physical properties of the production system, such as the number of workpiece-carriers, workstations, and AGVs, can be modified individually. Secondly, advanced settings of the workstations can be incorporated, which addresses the sizes of In-/Outbound-buffers and parking lots for AGVs. Thirdly, buttons have been implemented that can be used for importing data directly from CSV-files. Those exist for the specific production plans for product variants and the corresponding processing times, the skills that are required by processes and offered by workstations, and order lists. The CSV-files can be filled with data in advance, before the import-function distributes the content to the corresponding decentralized data storage within the MAS. Finally, the UI-A consists of a button that initializes the simulation. Fig. 5.11 shows a screenshot of the implemented dashboard that is used for incorporating the settings into the framework.
‘Data-Collection’-Agent:
Since the main use-case of the framework is to evaluate different approaches, a lot of data needs to be collected to derive comparisons covering varying aspects of the performance criteria. Therefore, occurring events in the simulation model have to be logged. This is the task of the DC-A. It stores data including the timestamp, the type and the location of events, such as arriving at and leaving from a workstation and the movements in the buffer for all workpiece-carriers in a MySQL database. It is connected to a visualization tool that quickly summarizes the current state of the production system. It shows the current position of each workpiece-carrier and the amount of waiting items in the buffers of each workstation. However, for generating more-in-depth analyses, SQL-queries can be used. By request, they calculate KPIs for both workpiece-carriers and workstations. This includes the throughput and waiting times for jobs, which generates knowledge about the fraction of value-adding-processes. From the view of workstations, the utilization and the identification of bottlenecks are supported. Both perspectives lead to an holistic comprehension of the performance of the production system with respect to the control approaches. Fig. ?? shows an exemplary data log of one simulation run.
5 Technical implementation

5.3 TCP-Socket-connections

The interface between the MAS and the computer running PlantSim is realized with socket-connections following the TCP-IP-standard. Each socket has the setting of the IP-address of the server and the required port-number. In this framework, each socket connection is mono-directional with the message being transferred from the client- to the server-socket. On the PlantSim side of the interface, a network called ‘Sockets’ consists of all sockets being used in this framework. They are connected with the simulation model by methods that are called by occurring events. For example, when a workpiece-carrier is arriving in the Inbound-Buffer of a workstation, a method is called to establish the TCP-connection and to send a message requesting the next process step. Alternatively, when a message arrives at a socket on the PlantSim-side, a callback function is called to process the incoming data. Both client- and server-sockets, including the corresponding callback functions, are centrally placed within the network ‘Sockets’. Fig. 5.13 shows the involved sockets in PlantSim.
5 Technical implementation

Figure 5.13: All PlantSim-sockets are centrally stored within a network and accessible by methods.

On the other side of the connection, the MAS also centralizes all sockets within one flow. This has the advantage of reducing the number of required sockets because all instances of an agent can use the same connection. All incoming messages will be forwarded to the instance representing the item that pulled the trigger in the simulation. For example, a message arrives requesting the next operation for the workpiece-carrier with the ID 4. A message containing the request is published with the topic ‘/Broker_rnj_4’. The WPC-A corresponding to the ID 4 is subscribing this topic and receives this request. Based on its services, the MAS will collectively find a solution. Finally, all decisions for each agent instance will be centrally collected and sent back to the simulation model via the sockets. Fig. 5.14 shows the implemented flow to exchange data using socket-connections.
The messages being sent between PlantSim and MAS are clustered by their functions. In total, six different categories of connections are used. These are Request-next-job (rnj), Request-next-step (rns), ‘Request-transport’, ‘Update-MID-status’, ‘Config-PlantSim’, and ‘Data-collection’. Their roles and functions are further outlined in the following:

- **Request-next-job:**
  The connection rnj is established, when a workpiece is loaded on a carrier inside the ‘Initializer’-network. The message being exchanged contains the unique ID of the workpiece-carrier to inform the corresponding WPC-A to initiate the flow of requesting the next required job. This is determined by the MAS depending on the algorithms being evaluated and transmitted through the TCP-connection back to PlantSim.

- **Request-next-step:**
  In contrast to rnj, which is only used once per workpiece, rns is established whenever a workpiece-carrier arrives at a workstation. However, the message also contains information about the carrier ID that is requesting a decision.
to designate the following workstation and triggers the corresponding agent within the MAS as well. Furthermore, the decision that results from centralized or decentralized job-shop-scheduling algorithms is messaged through the TCP-connection. The message contains the ID of the workpiece-carrier, the workstation performing the next operation, and the corresponding process time. When it arrives in the PlantSim-socket, the callback-function assigns the data as variables of the corresponding object.

- **Request-transport:**
  This connection is dealing with two types of messages. Firstly, it forwards the results of algorithms that are assigning transport jobs to available AGVs. Secondly, changes in the position of all vehicles are communicated via this connection, as well. This is used to track and to take possible delays of AGVs into consideration for planning purposes.

- **Update-MID-status:**
  PlantSim has the possibility to include downtime-behavior of workstations. Therefore, the MID-A inside the MAS needs to be informed to be excluded, whenever its simulation representation is unavailable. These updates are transmitted via this connection.

- **Config-PlantSim:**
  Inside the MAS, the is used to adjust parameters of the production system centrally. Some properties, such as the number of workstations, AGVs and workpiece-carrier, are not only relevant for the MAS, but for the simulation model as well. Therefore, those data sets need to be sent to PlantSim. Furthermore, the command to start the simulation is also sent via this connection.

- **Data-collection:**
  To enable proper functionality of the DC-A, a continuous data communication of data logs has to be ensured. This socket-connection is responsible for forwarding these data sets consisting of the timestamp of relevant events.
5.4 Network configuration of involved sub-systems

The previous sections have introduced different sub-systems that have been implemented to conduct virtual commissioning. However, one further important step is to outline the network configuration, which explains how the sub-systems are linked to each other. In total, three different networks are in usage, each being responsible for one sub-system. An overview of the involved networks and their subnets is summarized in Fig. 5.15.

Figure 5.15: In total, three different networks are used to enable a consistent data-flow in-between the involved sub-systems.

The computer running the PlantSim-simulation is required to be connected to the company-network, because of the need for permanent access to the license server. This is achieved with the regular WLAN-subnet. Additionally, the MAS is realized with a private WLAN, due to firewall-restrictions of the company network. All
Raspberry Pis having an agent-instance assigned are connected with the private WLAN. The socket-connections that link the simulation model with the MAS are using their own subnet, which is generated with an ethernet cable. However, to enable a consistent data-flow between all sub-systems, the interfaces need to be part of the adjacent subnets as well. For example, the Raspberry Pi that is used to distribute incoming messages from the sockets within the MAS is connected with both networks, the WLAN and Local Access Network (LAN). This is also true for the socket-network within PlantSim. With the help of an AUDI-internal application, it is possible to connect the computer with both subnets. This leads to the fact that both the computer and the selected Raspberry Pi have assigned two different IP-addresses.
6 Validation

As presented in chapter 2.3.2, VC has the objective to confirm the functionality of a specific system consisting of hard- and software. In order to validate the framework presented in chapter 5 for two control algorithms, each representing a centralized or decentralized approach, the question whether or not the framework is capable of performing the decision tasks in the MAS and of forwarding messages to the simulation model, needs to be answered. The validation of the implemented framework focuses on testing the ability to execute both centralized and decentralized control approaches and their adaptability to different layouts of the factory. To achieve this, two sets of control logic were implemented following the introduced architecture and tested in multiple simulation runs using varying parameter sets. Further, this chapter concludes with an evaluation of the validation results.

6.1 Centralized approach

One exemplary centralized approach with a pre-calculated schedule has been integrated into the implemented framework. The already determined routes for each job are centrally stored in a database at the CC-A. The data-flow for this scenario is illustrated in Fig. 6.1 and further outlined in the following:

1. Inside the simulation model, the workpiece-carrier with the unique Identification number (ID) arrives at a workstation. This event is used as a trigger to request the next process and workstation.

2. A message with the ID and the request is sent to MAS using the socket-connection.
3. Incoming messages at the MAS are forwarded to the corresponding agent. In this case, the request is forwarded to 'WPC-A_1'.

4. The WPC-A is contacting the CC-A with the need for the next process and workstation.

5. The CC-A consists of a global schedule having all jobs of a specific period of time assigned to workstations. Using a MySQL-query, the CC-A extracts the required information. Furthermore, this data is messaged back to 'WPC-A_1'. Afterwards, the CC-A initiates a parallel work-stream, which continues with number 9.

6. The decision is forwarded to MAS-sockets.
7. The socket-connection rns from the MAS to PlantSim is messaging the information about the ID, process, and the next workstation.

8. The callback-function of the rns-socket is updating the next destination of the specific workpiece-carrier.

9. The CC-A is also initiating the assignment of a suitable AGV. Therefore, the CC-A informs the Trans-A about the required transport from the workpiece-carrier’s current position to the next workstation.

10. Based on the availability of all AGVs, the Trans-A assigns an AGV to conduct the transport. The ID of the AGV and the departure point of the route is forwarded to MAS-sockets.

11. The socket-connection rtr is used to export the information to PlantSim.

12. The callback-function of the rtr-socket is used to inform the AGV with the selected ID to pick-up the specific workpiece-carrier and to transport it to its destination.

In order to confirm the functionality, this centralized approach was performed in several experiments with different properties of the parameters ‘amount of workstations’, ‘amount of AGVs’, and ‘amount of workpiece-carriers’. The different sets of properties are summarized in table 6.1. The objective of the experiments was to validate that the implementation is able to integrate the decisions from the MAS into the simulation model. Furthermore, the response time of each execution is of interest to get an insight into the scalability and stability of the framework. The response time is defined as the time interval starting with the trigger event requesting a new operation and ending with the assignment in PlantSim.
Table 6.1: Overview of the parameter properties for validating the centralized approach.

<table>
<thead>
<tr>
<th>Set ID</th>
<th>workstations</th>
<th>AGVs</th>
<th>workpiece-carriers</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>II</td>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>III</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>IV</td>
<td>6</td>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

The response times for all sets of experiments are summarized in Fig. 6.2. Each set was executed ten times.

![Box plot of response times](image)

Figure 6.2: All experiments were performed successfully with response times varying from 0.5 to 1 s.
6.2 Decentralized approach

The validation of a decentralized approach has been achieved by integrating a bidding process into the framework. Its key characteristic is having WPC-As selecting both the next operation and workstation depending on a bid price computed by all suitable MID-As that are able to conduct this operation. This approach follows the program sequence visualized in Fig. 6.3 and is further outlined in the following to emphasize the more complex data flow of the decentralized approach:

1. Within the PlantSim model, the event trigger is pulled by the workpiece-carrier with the ID 1 to request a new decision on the next operation.
2. A TCP-connection using the rns-sockets is established to send a message containing the ID of the workpiece-carrier and a code.

3. The rns-message is forwarded to the corresponding WPC-A_1.

4. The WPC-A asks the CC-A about the next possible operations for its loaded product based on the already performed steps.

5. The CC-A determines the next possible operations and workstations with respect to the production plans of the specific product variant. In this example, the workstations 11 and 12 have the relevant skills to perform the next operation and receive information about the type of process and the corresponding cycle time.

6. If the workstations are available, the corresponding MID-As calculate their bids based on the optimizing parameters (e.g., minimizing their idle time) and send it back to the CC-A.

7. The CC-A collectively forwards all incoming bids to the requesting WPC-A.

8. Based on the bids of the MID-As, the WPC-A selects one operation at one workstation. A message containing the decision is sent to MAS-socket. Furthermore, a second work-stream at the WPC-A is initiated, which continues with number 11.

9. Transmitted via the rns-socket-connection, the decision of the upcoming decision is sent to PlantSim.

10. Using the rns-callback-function in PlantSim, the data about the next workstation and process time is directly forwarded to the corresponding workpiece-carrier inside the model.

11. The WPC-A also informs the ‘winning’ MID-A.

12. The WPC-A is contacting the Trans-A to request a transport job from the workpiece-carrier’s current position to the upcoming workstation.

13. Based on the availability of all AGVs, the Trans-A assigns an AGV to conduct the transport. The id of the AGV and the departure point of the route is forwarded to MAS-sockets.
14. The socket-connection rtr is used to export the information to PlantSim.

15. The callback-function of the rtr-socket is used to inform the AGV with the selected id to pick-up the specific workpiece-carrier and to transport it to its destination.

In analogy to the centralized approach, several simulation runs were performed to confirm the functionality with respect to different parameter properties of the production system. Furthermore, the sets of validation also include taking the type of flexibility into account, such as operation or order flexibility or a combination of both. The reason why this is included in the decentralized approach validation is that the assignments are done in real-time. Contrarily in the centralized approach, the decisions were all made beforehand.

Table 6.2: Overview of the parameter properties for validating the decentralized approach.

<table>
<thead>
<tr>
<th>Set ID</th>
<th>workstations</th>
<th>AGVs</th>
<th>workpiece-carriers</th>
<th>flexibility type</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>operation flexibility</td>
</tr>
<tr>
<td>II</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>order flexibility</td>
</tr>
<tr>
<td>III</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>operation &amp; order flexibility</td>
</tr>
<tr>
<td>IV</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>operation flexibility</td>
</tr>
<tr>
<td>V</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>order flexibility</td>
</tr>
<tr>
<td>VI</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>operation &amp; order flexibility</td>
</tr>
<tr>
<td>VII</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>operation flexibility</td>
</tr>
<tr>
<td>VIII</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>order flexibility</td>
</tr>
<tr>
<td>IX</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>operation &amp; order flexibility</td>
</tr>
</tbody>
</table>
6 Validation

The response times for the decentralized implementations with the specific set properties are summarized in Fig. 6.4. Each set was executed ten times.

Figure 6.4: The response times primarily depend on the amount of involved agents, not on the type of flexibility.

6.3 Evaluation of validation experiments

The experiments demonstrate the functionality of the conceptualized and implemented framework. It is in line with the requirements that have been defined in chapter 3. In all runs of both centralized and decentralized control approaches, the requested data arrived on time within the simulation model, which led to a continuous interruption-free and consistent execution. The response time for all tested scenarios was short enough, and messaging queues did not occur. For the centralized approach, the response time is around 50% shorter than in the implemented decentralized approach. This is due to the more complex data flow, which consists of placing bids and, therefore, of more publish and subscribe processes. In both approaches, the differences in the mean values of their corresponding experiments are relatively low.
6 Validation

The standard deviation for the centralized approach is $\sigma_{central} = 47.02\text{ms}$ and for the decentralized approach $\sigma_{decentral} = 117.51\text{ms}$. Therefore, adjustments in the amount of involved entities didn’t considerably affect the performance of the framework. Therefore, it also fulfills the requirement of scalability.

The adjustments considering the different sets of experiments representing different factory layouts were achieved with the implemented user-interface. Due to its user-friendliness, the single-point adjustments lead to fast and straightforward modifications.
7 Discussion on the developed framework

This chapter provides a discussion on the overall framework and its sub-systems by referring back to the research questions introduced in chapter 1.3. Further, it describes how it shall be used for unleashing its main potential and which limitations and challenges exist.

7.1 Reflection on research questions

Research question 1:

Which sub-systems are needed to evaluate and compare different control algorithms?

Based on the principles of VC described by Zäh, the developed framework consists of three sub-systems that are ‘Simulation model’, ‘Control logic’, and ‘Interface’. This distribution enables the utilization of the advantages of VC in the early development phase of control approaches. Especially, the latest version of the control approach can be evaluated with the virtual representation of the modular assembly. This accelerates the development process and reduces the corresponding costs.

For each of the involved sub-systems, a technical solution for the implementation was investigated and selected. The final decision to use a PlantSim model to simulate the behavior of the modular assembly and a stack of Raspberry Pis for the control logic is the result of a broad evaluation of multiple alternatives to ensure a selection of high performance and support.

Since the final layout of the factory does not exist in this stage of development, it is of
high importance to allow flexibility in the modification in multiple parameters. With the support of the implemented passive 'User-Interface'-Agent, quick modifications of parameters, such as the number of workstations, AGVs, and buffer sizes, can be performed in a single window. This increases the extent of user-friendliness of the overall framework.

**Research question 2:**

What are the data flow differences for both a centralized and a decentralized control approach?

The differences in the data flow requirements were investigated in chapter 3.2 and were used to integrate the individual control logic in a SOA. This has the primary motivation to enable the adaption of the general and underlying structure to differing requirements of further control approaches. If a new control approach is evaluated with the framework, its individual data flow requirements can be considered in designing and programming new services that are provided by the involved agents. However, services can also be reused from alternating control approaches to some extent.

**Research question 3:**

Which communication protocol is useful to enable the data exchange between different cyber-physical entities?

Since cyber-physical entities represent both the hardware and the software components, their representation within the framework is also spread among the different sub-systems. Their hardware-related aspects are represented by the simulation model with the key information about their current position within the factory, whereas their ability to communicate with each other is assigned to the MAS represented by the Raspberry Pis. In order to have significant insights into the control approaches to derive meaningful evaluations, the form of communication for the Raspberry Pis has to be aligned with the entities of the planned modular assembly. At the same time, the simulation character of the framework requires very low latency times to avoid the loss or queue of messages. In chapter 6, the conducted experiments with the focus on the response times demonstrated that the selected MQTT protocol is
useful for communication among the involved agents. Its pattern in publishing and subscribing to topics is also supported by the standard OPC-UA, which has gained in importance over the last years in Industry 4.0 research. Since its latency times are comparatively high, it was not considered during the project.

Research question 4:

Does the implemented system fulfill the defined requirements?

In chapter 6, the validation experiments following both a centralized and decentralized approach outlined that the framework is capable of deriving decisions following various data-flow requirements with different services provided by the involved agents. It was also demonstrated that the different factory layouts with varying amounts of workpiece-carriers, workstations, and AGVs are supported, which leads to the desired flexibility. In chapter 3, requirements for the involved sub-systems were introduced. All of them are considered in the implementation of the framework which leads to the required standardization with as much flexibility as possible to address the different needs of control approaches.

The simulation model is following the principles 1 to 7 of a modular assembly system that were outlined in chapter 2.1. Principle 8 and 9 are not considered. This can be motivated for principle 8 with the assumption of process stability. Although a defect of a workstation can occur, which leads to downtime behavior, all malfunctioning parts are detected immediately. Therefore, further quality control loops are not implemented in the model. Since assembly workers are not implemented in the simulation model at all, varying skills and experience level of the workers cannot be used for alternating processing times in today’s implementation of the framework. However, these two principles have to be seen as opportunities that are not mandatory within the concept of modular assemblies. Therefore, the developed simulation model is still valid and acts aligned with the physical role-model.

7.2 Utilization of the framework

The implemented framework shall be used to evaluate the performance of a control approach with respect to a defined factory layout, skills provided by workstations,
assembly priority charts of products and released orders. The composition of the involved sub-systems enables a standardized evaluation based on KPIs, which can be further used for the comparison of multiple control approaches to select the most suitable one for specific conditions. Fig. 7.1 shows a typical order of adjustments to fully exploit the benefits of the framework.

First of all, the control logic has to be incorporated into SOA, which is built-up among the involved active agents. Following the patterns of subscribing and publishing, services for each agent type are created within the NodeRed-environment. Secondly, the user interface is used to prepare both the simulation model and the databases containing relevant data, such as APCs and orders. Further, the simulation model can be updated with relevant data, such as the buffer sizes of workstations. The user-interface also initiates the simulation run with the click on a button. These single-point adjustments have been proven to reduce the complexity of the framework during the validation process in chapter 6. During the simulation run, the Data-collector agent continuously receives the timestamps of occurring events. The third step in using the introduced framework is to execute MySQL-queries to determine the KPIs. The results can be used for further comparison with other control approaches while using the same primary conditions. Alternatively, it can also be used to evaluate the robustness of a control approach by varying the skill assignments, orders, etc.

Another use-case of the framework is to use the insights into the performance of the control approach for detailed improvements of the involved services.
7 Discussion on the developed framework

7.3 Limitations and challenges

As presented in the previous sections, the implemented framework is a powerful tool to increase the knowledge about the quality of production control approaches for the modular assembly. However, its functionality is only given with valid data about APC, orders, and skills. Therefore, these sets of input data have to be checked for consistency before initiating the simulation run.

Additionally, the implementation of the different agent services is time-consuming, since changes in one agent instance have to be duplicated to the remaining instances. In further implementations of control approaches, it can be the case that more detailed information from the current state of the simulation model is needed. This can result in more-complex data-flow requirements. This challenge can be addressed with adjusting the simulation model by inserting triggers to inform the external control system about its state. Therefore, the overall framework shall not be seen as a stiff tool-suite.
8 Conclusion and future work

The last chapter of this master thesis concludes the work and also contains a section with the future work that introduces the next steps of the developed framework and open research questions within the modular assembly.

8.1 Conclusion

This master thesis has aimed to conceptualize and develop a framework for VC of control approaches for the modular assembly. With the definition of requirements of such a tool suite, a standardized framework has been created that is following the principles of VC described by Zäh considering the sub-systems ‘Simulation model’, ‘Control logic’, and ‘Interface’. A PlantSim simulation model represents the hardware components of a production system such as workstations, workpiece-carriers, and AGVs. Its implemented event triggers are used to request decisions for job release or the assignment of operations and workstations for jobs. These corresponding decisions are derived from a MAS that consists of agents that communicate with each other. The agents are defined by software scripts being executed by a set of Raspberry Pis and can be adjusted to the control logic of both a centralized or decentralized control approach. The architecture is based on SOA-principles and leads to agent’s requests in services. Derived decisions are incorporated back into the simulation model to visualize the decisions and to update the production execution. The interface between the simulation model and the MAS has been established with TCP-connections.

The functionality of the framework has been validated with exemplified centralized and decentralized approaches. They have been integrated into the framework and have been performed in several simulation runs with alternating sets of parameter properties. The experiments have shown that the response time is sufficiently low.
enough to ensure decisions being sent back into the simulation model on time and
that the framework is flexible enough to allow an adaption to different layouts of the
factory. In both approaches, data losses did not occur. Therefore, the framework shall
be used for further analyses of sophisticated control approaches.

8.2 Future work

Having this framework installed as a fundamental tool for evaluating the perfor-
mance of control approaches, further implementations of control algorithms shall
be integrated into it. With a set of well-configured control systems, the factors of
influence shall be identified to understand the modular assembly behaviors. The
main question that needs to be answered is in which conditions what type of control
approach is outperforming the other one. Furthermore, the tool suite can be further
improved to suggest a specific control approach based on the parameter properties
of the production system and the produced products with their APCs. Additionally,
the tool suite could suggest further improvements, such as a modification of the
number of AGVs or workstations. Next to ongoing improvements, this increase in
transparency will also affect the level of acceptance of modular assembly systems
within responsible departments.

Besides optimizing the control approaches, further work should also include what
the right product for modular assemblies is. In today’s design departments, products
are optimized for automated assemblies. In many cases, this is achieved by sandwich
assemblies building up a product layer by layer. However, this procedure reduces the
extent of flexibility that a production system can address. Therefore, a methodology
to evaluate the characteristics of a product with respect to its suitability to modular
assemblies should be developed.
A Control approach implementations

For verifying the functionality of the developed framework, both a centralized and a decentralized control approach were implemented in chapter 6. The flows of the used agents and their corresponding services are outlined in the following:

A.1 Centralized approach

WPC-A

Figure A.1: Services of the centralized WPC-A.
CC-A

Figure A.2: Services of the centralized CC-A.
A.2 Decentralized approach

WPC-A

Figure A.3: Services of the decentralized WPC-A.
CC-A

Figure A.4: Services of the decentralized CC-A.

MID-A

Figure A.5: Services of the decentralized MID-A.
Trans-A

Figure A.6: Services of the decentralized Trans-A.
# List of Figures

1.1 Differences in processing times lead to waste in mixed-model assembly lines. .................................................. 2

2.1 Visualization of the layout of a modular assembly. ......................... 7
2.2 The exemplary product characterized by its APC(a) utilizing order and operation flexibility has four different routes through the production (b). Source: Adapted from [15] ........................................ 10
2.3 A global schedule has all jobs assigned to workstations and AGVs within a specific period of time. ......................... 12
2.4 Division of the life-cycle of production systems. ....................... 14
2.5 Auinger distinguishes three approaches of virtual commissioning. Source: Adapted from [29] ................................. 16
2.6 Within SOA, the hierarchical functions of the automation pyramid are distributed among entities and services. Source: Adapted from [45] . 20

3.1 Visualization of the material flow within a workstation used in modular assembly. ........................................... 23
3.2 Data-flow within centralized control approaches. ...................... 25
3.3 Data-flow within decentralized control approaches. ................. 26
3.4 Centralized parameter settings lead to an adjustment in all sub-systems. 28

4.1 Node-RED is a graphical programming environment for developing application flows. ........................................ 32
4.2 Overview of the MQTT principle of publishing and subscribing. Adapted from [60] ........................................... 34

5.1 The framework consists of a simulation model running on a computer, a MAS executed by a stack of Raspberry Pis and TCP-socket-connections in-between. ..................................................... 36
5.2 The network ‘Initializer’ deals with the coordination of the release of movable elements. ...................................... 37
5.3 The Transport-grid connects all workstations with tracks used by AGVs. 38
5.4 Workstations are representing the material flow of its physical role-model. 39
5.5 The network ‘Finisher’ consists of the dis-assembly of the workpiece from the workpiece-carrier. ............................... 40
5.6 In total, ten Raspberry Pis compose the MAS. ........................ 41
5.7 The CC-A consists of multiple data storages that are used within its services. .................................................. 42
5.8 The WPC-A digitally represents a workpiece-carrier inside the MAS that exists within the simulation model. .................. 43
5.9 Services from the MID-A can either be requested with a message topic consisting of the skill or the ID of the workstation. .... 43
5.10 The Trans-A assigns AGVs to transport jobs. ......................... 44
5.11 The UI-A can be used for adjusting settings of the framework centrally. 45
5.12 The DC-A stores the timestamps of relevant events within a mySQL-Database. ............................................. 46
5.13 All PlantSim-sockets are centrally stored within a network and accessible by methods. ........................................... 47
5.14 All MAS-sockets are centrally stored within a flow and forward incoming messages to the corresponding agents via MQTT. .... 48
5.15 In total, three different networks are used to enable a consistent data-flow in-between the involved sub-systems. ............... 50
6.1 The CC-A assigns workstations to jobs based on the global schedule on request. .................................................. 53
6.2 All experiments were performed successfully with response times varying from 0.5 to 1 s. ............................................. 55
6.3 The MID-As are participating in a bidding competition to collectively derive decisions. ................................................. 56
6.4 The response times primarily depend on the amount of involved agents, not on the type of flexibility. ............................... 59
7.1 The evaluation is performed with modifications within the MAS. . . 64
A.1 Services of the centralized WPC-A. ..................................... 68
A.2 Services of the centralized CC-A. ........................................ 69
A.3 Services of the decentralized WPC-A. .................................... 70
A.4 Services of the decentralized CC-A. ....................................... 71
A.5 Services of the decentralized MID-A. ..................................... 71
A.6 Services of the decentralized Trans-A. .................................... 72
List of Tables

2.1 Overview of the nine principles of modular assembly systems. . . . . 7
4.1 Comparison of SBC-alternatives. . . . . . . . . . . . . . . . . . . . . . . 31
6.1 Overview of the parameter properties for validating the centralized
approach. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 55
6.2 Overview of the parameter properties for validating the decentralized
approach. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 58
Acronyms

**AGV** Autonomous Guided Vehicle. 2, 3, 6–8, 10–12, 23–25, 27, 37–39, 43, 44, 49, 54, 55, 57, 58, 62, 63, 66, 67, 73, 74

**AOET** Actual Order Execution Time. 21, 22

**APC** Assembly Priority Chart. 3, 9, 10, 64, 65, 67, 73

**APT** Actual Production Time. 21, 22

**AQT** Actual Queueing Time. 21

**ATT** Actual Transportation Time. 21

**CC-A** ‘Central Coordinator’-Agent. 41, 42, 52–54, 57, 69, 71, 74

**CPPS** Cyber-Physical-Production-System. 19, 20, 40

**DC-A** ‘Data Collection’-Agent. 44–46, 49, 74

**DES** Discrete Event Simulation. 29, 37

**ERP** Enterprise Resources Planning. 20

**GUI** Graphical User Interface. 29

**ICT** Information- & communication technologies. 2

**ID** Identification number. 52, 54, 56, 57

**IDS** Intelligent Distributed System. 25–27, 29, 30, 34–36

**IoT** Internet of Things. 31, 32

**IP** Internet Protocol. 34, 35, 40, 46, 51

**JSON** Java Script Object Notation. 32, 41

**KPI** Key Performance Indicator. vii, 3, 6, 8, 20–22, 24, 45, 64
Acronyms

LAN  Local Access Network. 51

MAS  Multi-Agent-System. 34–37, 40, 41, 43, 44, 46–54, 57, 62, 64, 66, 73, 74

MES  Manufacturing Execution System. 17, 20

MID-A  ‘Workstation’–Agent. 41, 43, 49, 56, 57, 71, 74

MQTT  Message Queuing Telemetry Transport. 33, 34, 40, 48, 62, 73, 74

PlantSim  Technomatix PlantSim. 29, 30, 37, 46–48, 74

PLC  Programmable Logic Controller. 16

RAM  Random Access Memory. 30, 31

rnj  Request-next-job. 48

rns  Request-next-step. 48

ROS  Robot Operating System. 31–33

SBC  Single Board Computer. 30, 31, 40, 75

SOA  Service-Oriented-Architecture. 6, 19, 20, 26, 30, 40, 62, 64, 66, 73

SOP  Start-of-Production. 18

TCP  Transmission Control Protocol. 35, 46, 48, 49, 66

Trans-A  ‘Transport’–Agent. 41, 43, 44, 54, 57, 72, 74

UI-A  ‘User Interface’–Agent. 44, 45, 74

VC  Virtual Commissioning. 15–18, 23, 52, 61, 66

WLAN  Wireless Local Access Network. 26, 27, 32, 50, 51

WPC-A  ‘Workpiece Carrier’–Agent. 41–43, 47, 48, 53, 56, 57, 68, 70, 74
Bibliography


Bibliography


