DISSERTATION

DEVELOPMENT OF WORKING SCENARIOS AND STRATEGIES FOR SELF-ORGANIZING ASSEMBLY SYSTEMS

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Doktors der technischen Wissenschaften unter der Leitung von

Professor Dr.sc. Dr. mult. h.c. Prof. h.c. Branko Katalinic
E311
Institut für Fertigungstechnik und Hochlaserleitungstechnik

eingerecht an der Technischen Universität Wien
Fakultät für Maschinenbau

von

Dipl.-Ing. Ilya Kukushkin
Matr. Nr. 0828367
Gumpendorferstraße 39,232
1060 Wien

Wien, im Juli 2014

Abstract

This dissertation describes the research dealing with the development of working scenarios and strategies for self-organizing assembly systems. The thesis consists of theoretical and practical parts. Theoretical part analyses the meaning of assembly process in modern production and shows the development line of modern assembly systems.

Practical part includes three fields of investigation. The first field of investigation is the implementation of self-organizing Bionic Assembly System (BAS) for assembly of the product with the complexity level of electrical motors (0,5 - 1 kW). The existing assembly solution for same motors is realized as Flexible Assembly System (FAS). The FAS is used for comparisons, analysis and references during the development and verification of a new system. The research is focused on the layout reconfiguration, control structure adjustments and the development of new algorithm for assembly sequence planning.

The adjustment of control structure includes the second field of investigation: integration of cloud communication into control system of BAS.

The third field of investigation is the comparison of FAS and BAS performances. This includes BAS behavior simulation and comparison with FAS.

The result of the investigation confirms the suitability of proposed BAS concept for assembly of multiple families of the products with the complexity of electrical motors.

The cloud concept has been implemented as informational interface between sub-ordinated and self-organizing subsystems of BAS control structure. This gives the possibility to organize direct and simple horizontal communication on BAS shop-floor and vertical communication between the subsystems.

The mobile robot algorithm for the selection of the most suitable assembly operation-station combination, based on the shortest time is developed. The algorithm allows to find out sub-optimal trajectory for each robot, considering system states and conditions. Simulation shows that implementation of this algorithm allows to decrease assembly run time.

Simulation-based investigation of working scenarios and strategies of developed system gives the possibilities to determine the optimal BAS configuration(number and types of stations and robots), most suitable working scenario and run mix. It also supports the human operators in decision-making process for analysis and improvement of system behavior.

Acknowledgement

Firstly I would like to acknowledge the contribution of large number of people, who in different ways provided me invaluable support for completion of this thesis.

Most of all I would like to thank Professor Dr.sc. Dr. mult.h.c. Prof. h.c. Branko Katalinic for his scientific and personal guidance. It was a pleasure to be with such great scientist and person.

Special thanks to Professor Dr. sc. Valentin Pryanicnikov for the review and comments to my dissertation.

I would like to thank Mag. Agniezska Radziwon and Mag. David Cerna for the proof-reading of this dissertation and corrections suggestions.

I gratefully acknowledge the help of Dipl.-Ing. Albert Lidauer, who inspired me with his lectures as well as gave a possibility to visit MAGNA Exteriors and Interiors plants and get valuable experience and better understanding of modern industry.

Special thanks for my good friend and colleague Mag. mech Damir Haskovic for his support and understanding as well as nice discussions and time spent together in IMS group office.

This work is supported by Erasmus Mundus Multic program of European Union. I am really grateful for the financial support of my studies in Vienna.

Ilya Kukushkin

Contents

Chapter 1. Introduction **1**
1.1 Objective 2
1.2 Thesis Outline 2
1.3 Glossary of Abbreviations 5
Chapter 2. Assembly Systems **6**
2.1. Assembly systems parameters 8
2.2. Developing line of modern assembly systems 10
2.3. Self-organization in nature and technology 12
Chapter 3. Flexible Assembly System for AC motors 14
3.1 Product Analysis 15
3.2 Structure of FAS 22
3.3 FAS control system and scheduling 29
Chapter 4. Bionic Assembly System 33
4.1 BAS basic characteristics 33
4.2 BAS components and subsystems 36
4.3 BAS control structure and scheduling strategies 38
4.4 BAS reconfiguration 41
4.5 Summary 46
Chapter 5. BAS Adjustments 47
5.1. Layout reorganization 47
5.2. Control structure adjustment 50
5.3. BAS working scenario adjustment 63
5.4. Summary 66
Chapter 6. BAS Implementation **67**
6.1.BAS layout organization 67
6.2. BAS assembly flow organization 70
6.3. Summary 74

Chapter 7. BAS Simulation Organization 75
7.1 AnyLogic software and functions overview 77
7.2 AnyLogic model organization 82
7.3 Summary 94
Chapter 8. BAS Behavior Analysis **95**
8.1. System behavior analysis 95
8.2. Robot behavior analysis 102
8.3. System performance comparisons 105
8.4. Comparison of FAS and BAS 106
8.5. Summary 107
Conclusion **109**
References **111**
Appendix A. AnyLogic listing **116**
Appendix B. Author's Curriculum Vitae **135**

"Поехали!"(Let's go!)
Yuriy Gagarin

Chapter 1

Introduction

The beginning of 21st century is characterized with a rapid development of science and technology. It is possible to see the significant difference in development rate of different fields. The ideas, results and concepts from faster and more successfully developed fields disseminate to all the others.

Development of modern industry is very strongly supported by the progress of information technology. This developing trend has a target to build a new kind of industry, generally known as Industry 4.0.

This trend also influences further development of production systems. The efficiency of these systems depends directly on the machines productivity as well as system control and scheduling strategies. The trend is focused on four main improvements: increase of system flexibility and system intelligence and implementation of self organization and cloud computing.

Assembly systems are the specific type of production systems. They very strongly depend on the assembled product. Therefore it is impossible to design the universal assembly system for a wide range of different products. In comparison with other production systems they have higher complexity and lower level of automation.

The research described in this dissertation focuses on the implementation of Industry 4.0 elements in assembly systems domain.

1.1 Objective

The primary scientific aim of this dissertation is the description of the research dealing with the development of working scenarios and strategies for self-organizing assembly systems. The thesis consists of theoretical and practical parts. Theoretical part analyses the meaning of assembly process in modern production and shows the development line of modern assembly systems.

Practical part includes three fields of investigation:

1. Implementation of self-organizing Bionic Assembly System (BAS) for assembly of the products with a complexity level of electrical motor.
2. Integration of cloud communication into control system of Bionic Assembly System
3. Comparison and critical analysis of the proposed Bionic Assembly System based solution and existing assembly solution realized as Flexible Assembly System (FAS). This comparison has to include the layout, control system, mode of assembly, workers integration, reconfiguration

1.2 Thesis outline

Chapter 1. This chapter shows introduction, thesis objective and outline. This includes dissertation aim, fields of investigation and number of methods to research them.

Chapter 2 discusses modern trends in product development, the meaning of assembly in modern production, the time and money share of assembly in product value. Second part of the chapter shows the development line of assembly systems from manual till flexible and self-organizing assembly systems.

Chapter 3 focuses on the Flexible Assembly System for assembly of AC motors. This includes the product analysis, description of general FAS characteristics, its components and their functions. Second part of this chapter discusses FAS control system and scheduling and its working scenarios. Additional topics such material supply, integration in the factory and system self-organization are discussed at the last part of the chapter.

In Chapter 4 is proposed a whole concept of Bionic Assembly System (BAS), its elements, layout description, algorithms and working scenarios. Most of the attention given to explanation of self-organizing element of a system - a swarm of mobile robots and its algorithms. Second part of the chapter includes an explanation of a hybrid control structure of the system, scheduling and delivery organization as well as connection with a plant control system.

Practical part of this dissertation includes analysis of BAS implementation possibility. It is proposed to reorganize an electrical motor facility for assembling two families of electrical motors. The main goal of this project is apply BAS concept in order to design single system, which is able to cover all the abilities of installed FAS. A case decision is organized in 4 steps shown in Fig. 1.1.

Chapter 5 focuses on the description of goals, requirements as well as conditions, limitations and restrictions for BAS implementation possibility. The task is to design a BAS-based system, which covers the production range of FAS. The next part describes BAS adjustments. BAS concept is improved in order to fulfill analyzed conditions, restrictions and limitations. This part includes layout and control structure reorganization and adjustments of BAS working scenario.

Chapter 6 represents a new facility layout and its elements for a combined assembly system. At this step, a new system, which satisfies all the requirements is created according to a BAS concept. Additionally the assembly flow for both products is developed.

In Chapter 7, the concept of the system is verified. In order to perform the testing in the best possible manner, a suitable simulation tool is chosen and all the working scenarios and algorithms are translated into a model state charts and functions. For this purpose a special simulation tool was created using Anylogic software. The chapter presents modeling environment, translation of algorithms to the JAVA programming language, realization of working algorithms, reconfiguration abilities and simulation of different working modes of the system. Additional model listing is attached to this dissertation as an Appendix A.

Chapter 8 includes a number of experiments to check working characteristics of the system. The system is tested in different working modes, under different conditions. The results are discussed and compared with the target goals and characteristics of initial systems.

The conclusion discusses key outputs of this thesis as well as future research.

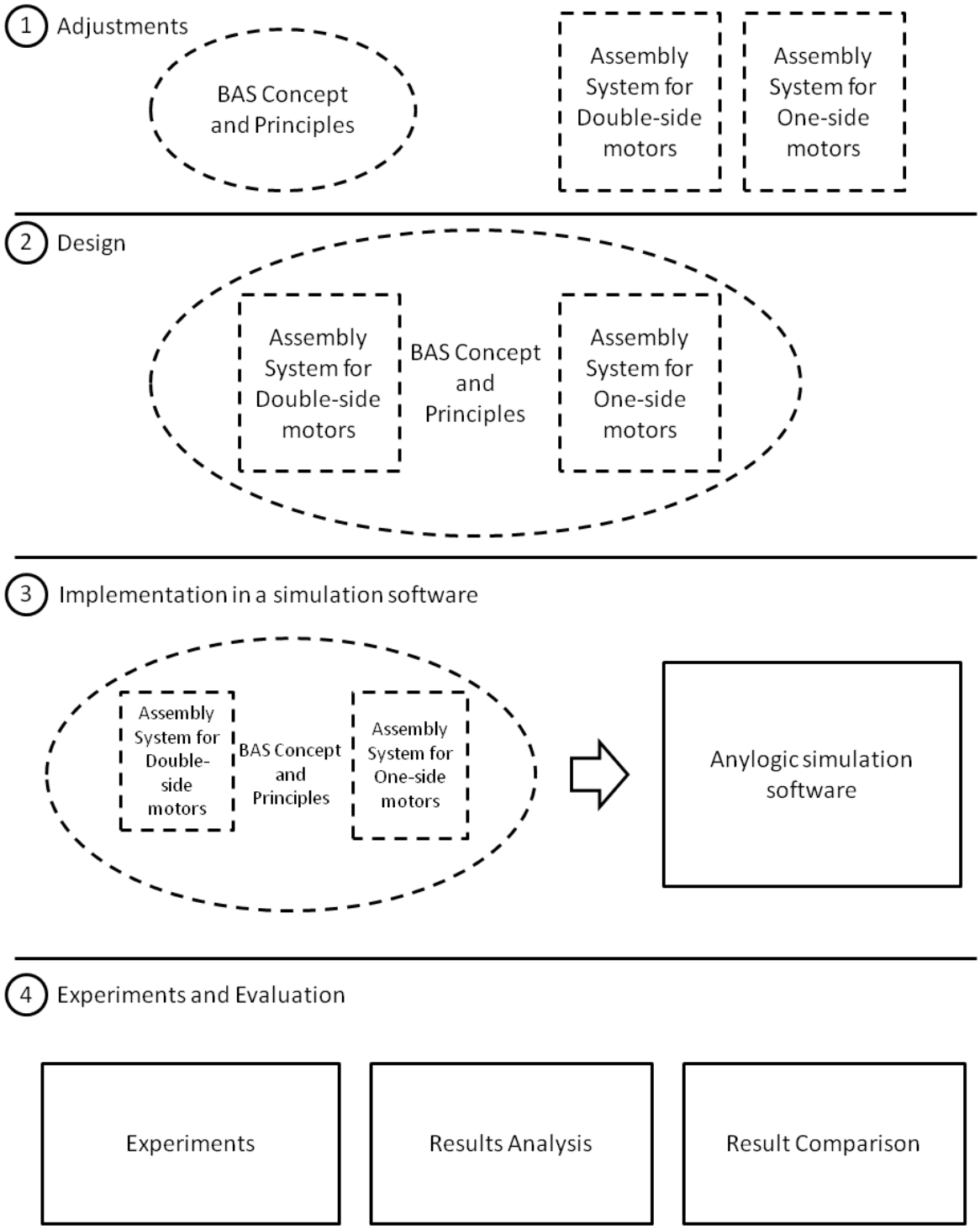


Fig. 1.1 Practical part flowchart

1.3 Glossary of abbreviations

AB	Agent-Based
AC	Alternative Current
BAS	Bionic Assembly System
BD1/2	Balancing Disc Station - 1/2
C1/C2	Capacitor Assembly Station - 1/2
DE	Discrete Event
ERP	Enterprise Resource Planning
ES1	1st End shield Assembly Station
ES2/ES	2nd End-shield / End-shield Assembly Station
FAC	Flexible Assembly Cell
FAS	Flexible Assembly System
FIX1	Fixture of Motor and Electrical Components - 1
FIX2	Fixture of Motor and Electrical Components -2
FP-AS	Fixing plate Assembly Station
GUM1	Rubber buffers control station
GUM2	Rubber buffers screw station
HEAT	Heating Station
IAM	Intelligent Advisor Module
ID	Identity Document
JIT	Just-In-Time
OO	Operator Order
ORO	Operator Repair Order
OSO	Operator Setup Order
PAK	Unloading/Packing Station
POOL-P	Pool of Pallets
POOL-R	Pool of Robots
QC	Quality Control Station
RAO	Robot Assembly Order
REP	Repair Station
Rot	Station Rotation
SAO	Station Assembly Order
SD	System Dynamics
STATOR	Stator Assembly Station
TSO	Transport System Order
V1	Fan Assembly Station

Chapter 2

Assembly process

Technical products are made out of the parts. The number of the parts varies from one to millions. Complex products, which are build out of more parts can perform additional functions, comparing to the simpler products. One of the trends in modern technology is the increase of the number product functions. Therefore number of the parts within the product increases and product becomes more complex.

The production process of one complex product consists of a number of sub-processes, as shown in Fig. 2.1.

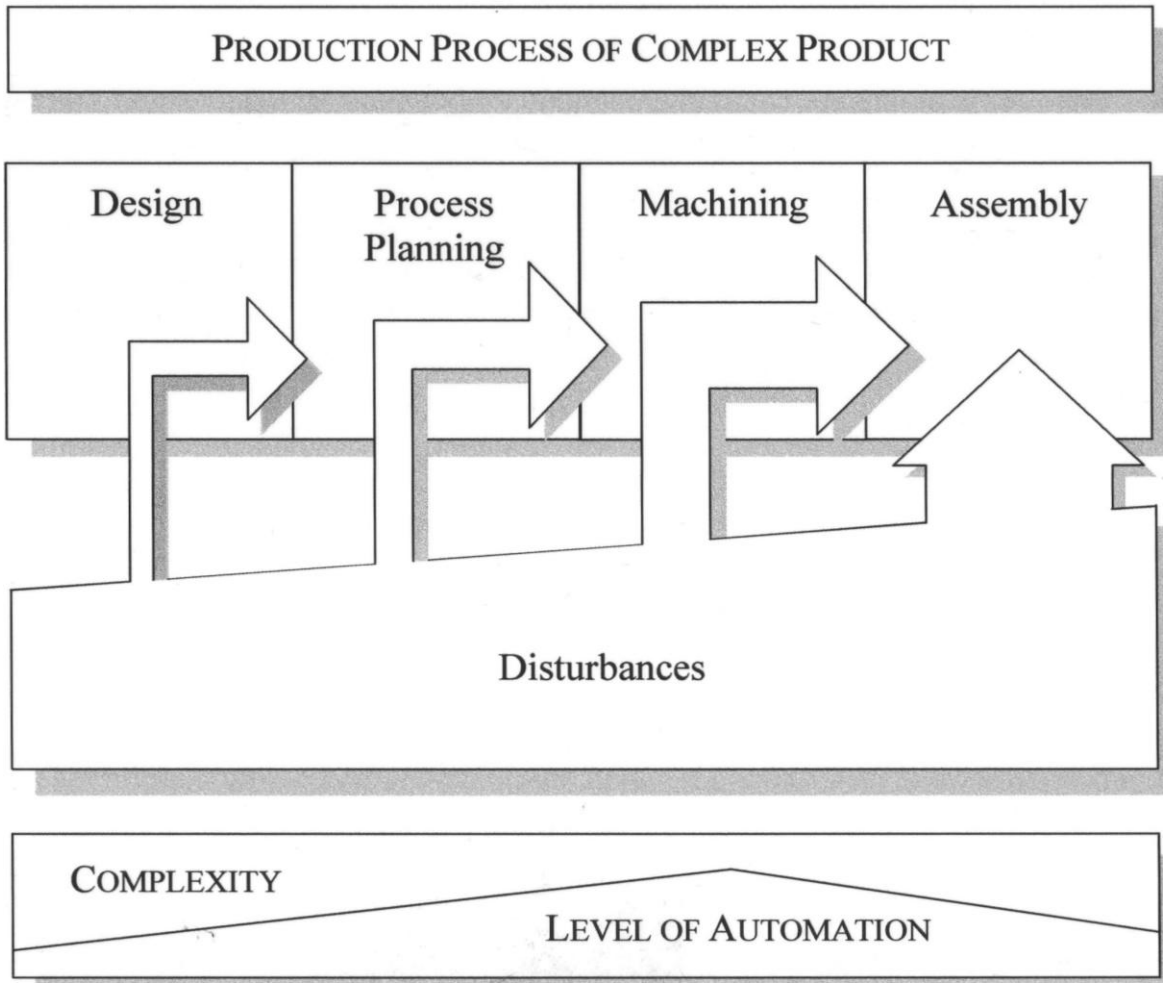


Fig. 2.1. Production process of complex product (VDI, 1990)

- **Design.** it's a process of constructing the action plan from the pure idea. Within design process all the product parts, features and characteristics must be planned. The start of this process is a concept phase, where the product concept is proposed. Then is development phase, where product is created according to the concept. After that, product is verified and prepared for serial production.
- **Process planning.** This process determines, which processes are suitable for the parts production and the sequence in which the processes should be performed to produce the designed product.
- **Machining.** it's a production process, where machine tools are used for the parts production (Groover, 2001). The result of this process is the set of the parts, which are ready for assembly.
- **Assembly.** Assembly is a process of integration. From one side it brings the parts together into one product. From the other side it brings people, departments, companies together. Assembly is connected with production at different levels. For the business context assembly influences production volume, model mix, product variety, customization, etc. (Ulrich & Probst, 1995). At the system level the way of assembly influences on assembly sequences, assembly quality, system layout, people involvement etc. (Cochran, 2002). At the product level assembly process depends on individual part joining, parts quality, part preparations, product complexity etc. Assembly it is a process, where the product comes to life (Whitney, 2004).

Each of these sub-processes has the factors, which impact the final product. The main factors influencing into the final product are the disturbances, process complexity, level of automation and resources consumption.

1. **Process disturbances.** Disturbances and mistakes accumulate within production process, as shown in Fig. 2.1. All the mistakes and disturbances sum up from process to process. Therefore, assembly as the last process of production chain has the biggest number of mistakes and disturbances.
2. **Process complexity and level of automation.** Modern machining systems produce the "universal" parts, which are quite independent from the final product (Lazinica & Katalinic, 2007). Therefore this process has the highest level of automation and the lowest complexity level. Assembly process consists of the range of different parts and operations. Therefore assembly is stated as the most complex process with the lowest level of automation.
3. **Process time and money consumption.** Key values in the resources consumption are time and money. Assembly claimed according to the product 15 to 85% of the total production time and costs (Lotter & Wiendahl, 2006). In the machines construction the assembly time between 20 and 45 %, depending on the complexity of the object. In the automotive industry the assembly time is 52-56%. The entertainment

electronics and PCs requires 82-85% of time and money resources for assembly process.

According to these factors assembly plays a key role in the production of technical products. Improvement of assembly process will lead to reduction of production costs, increase of the product quality, overall system productivity etc.

2.1 Assembly systems classification

According to Spath (Spath at al., 2007) there are three main criteria for determining the initial assembly system design. They are volume of production, range of products and initial investment, as shown in Fig. 2.2. There is an inverse correlation between product range and production volume in terms of factory operations.

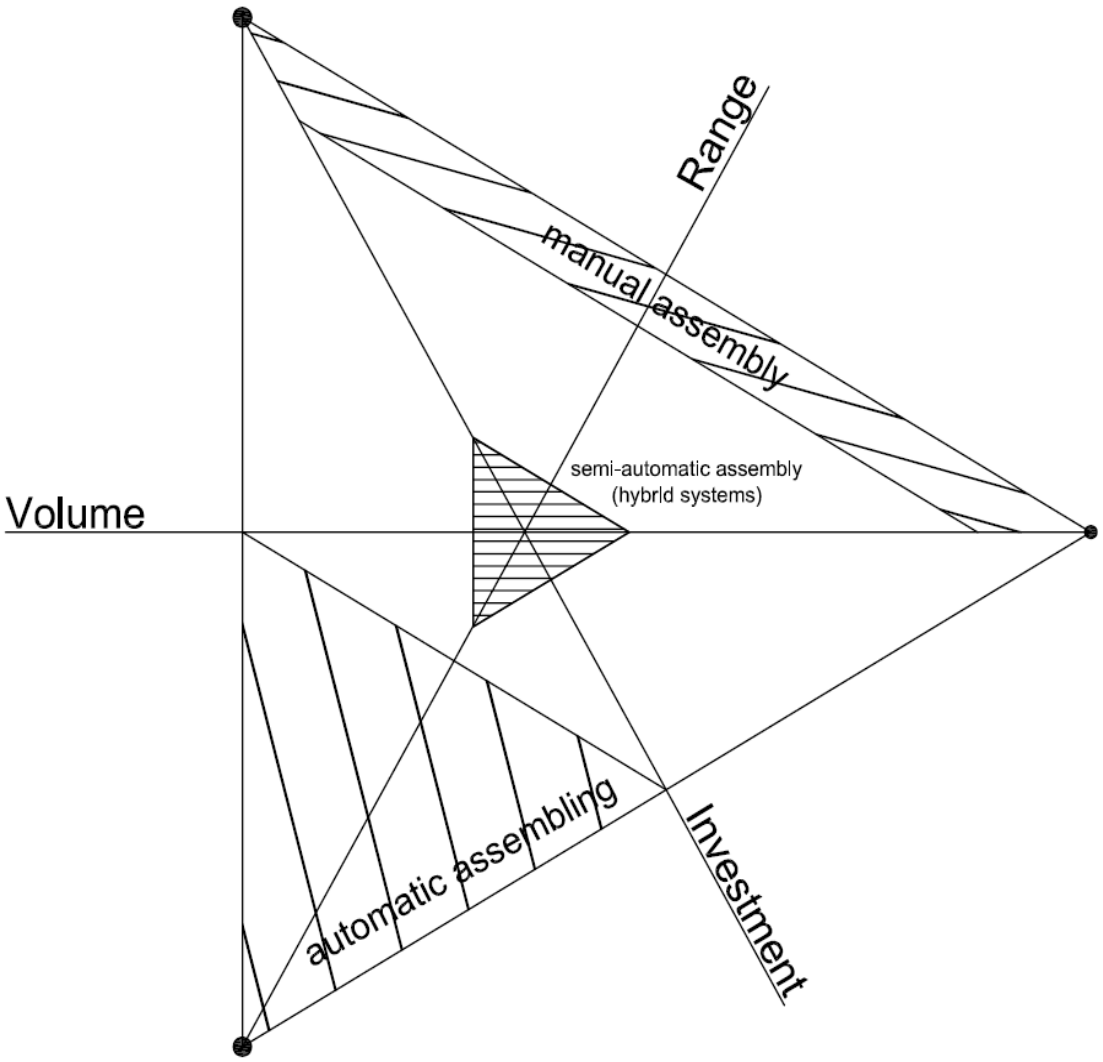


Fig. 2.2. Criteria for assembly system design

When product volume is high, range of produced products will be low and vice versa. This criteria lead to the classification into three big groups: manual, automatic and hybrid.

- **Manual assembly system.** Manual assembly gives the highest flexibility with lowest initial investments (Lotter, 2006). But the volume of production will be lower because of manual assembly operations. A typical example of manual assembly system is a job shop. Job shops must are developed to deal with the wide range of parts and product variations (Groover, 2007).
- **Automatic assembly system.** Automatic assembly gives the highest volumes of production, but requires high investments. Additionally the automatic systems are inflexible and not robust for the internal or external disturbances. A typical example of high-automated system is an assembly line. Assembly line consists of a number of stations, which are placed in predefined sequence. The parts flow from the beginning till the end of the line to complete the product.
- **Hybrid assembly system.** Semi-automatic or hybrid assembly allows to combine people and machines in one system. That gives higher flexibility with lower investments and middle volume of production. This systems have number of examples. One of this examples is Flexible Assembly System (FAS). The main difference between FAS and automated assembly line is FAS ability to assemble more than one product family within the system. FAS can also deal with changes in product mix and production schedule and could be reconfigured in any time.

Each of the strategies could be used for the different assembly cases. The comparison of these three examples is shown in Tab. 2.1.

Table 2.1. Assembly Line vs. Job-Shop

Criteria	**Assembly Line**	**Job Shop**	**FAS**
Investments	High	Low	High
Equipment utilization	High	Low	Medium
Flexibility	Low	High	Medium
Transport time	High	Low	Medium
Destruction point	Yes	No	Yes
Work-In-Progress	Low	High	Medium
Workers friendly	No	Yes	No

2.2 Developing line of modern assembly systems

Development of technology in XX century changed an original system dramatically. Implementation of computer control systems, Enterprise Resource Planning (ERP) systems, as well as lean techniques changed a companies from just mass-production and pushing it to a market to a complex systems for producing the variety of products with lower volumes (Matt, 2006), (Matt, 2008).

In early 60th the development of the computers leads to the development of the new assembly system class - Computer Integrated Systems(CIM). However, the first CIM were not able to assemble the variety of products and integrate the workers. Therefore, the next generation was focused on increase of flexibility. This systems were called Flexible Assembly Systems (FAS).

The flexibility can be defined several ways. The main flexibility criteria for assembly systems (De Toni & Tonchia, 1998) are:

- Range flexibility. It is a system's ability to operate a different part mix
- Operational flexibility. It is a system's ability to operate a variety of ways for product assembly
- Volume flexibility. It is a system's ability of run at different production rate.
- Routing flexibility. It is a system's ability to organize a different routing in case of machine failure.

Each assembly system type has its own level of flexibility. Flexibility could be compared for different system configurations. The comparison table is shown in the Tab 2.2.

Tab. 2.2. Flexibility criteria applied to the three types of assembly systems

System type	Range flexibility	Operational flexibility	Volume flexibility	Routing Flexibility
Single station	Low	Low	High	No
FAC	Middle	Low	Middle	No
FAS	High	Middle	Middle	Middle

- Single assembly cell / station - single manual or automatic station for a complete assembly process. Normally used for a low volume and range assembly. A single

assembly cell consists of one assembly station combined with own storage for parts and assembly means.

- Flexible assembly cell (FAC) consists of two or three manual or automatic stations for assembly process and loading/unloading station. The stations are connected with handling subsystem. The handling subsystem usually includes a limited parts storage capacity.
- Flexible assembly system (FAS). This system has four or more assembly stations connected in two levels. Mechanical connection is organized by a handling system and electronic connection by a central computer. Another difference is that the FAS generally includes non-assembly stations that support the assembly process. These other stations include pallets storages, repair stations, measuring and control stations and so on. The FAS is overlooked by complex control system, which has special diagnostics tools, sensors and other monitoring means. Typical FAS control structure is top-down hierarchical structure. A more appropriate term for FAS would be flexible automated assembly systems. This type of systems is developed to combine the features of manual assembly systems, which are flexible, as well as assembly lines, which are automated.

There are limitations to the parts and products range which can be assembled on FAS. The FAS as a hybrid system is most effective for the mid-volume production range with the average product variations. In other words, an FAS is capable to assemble a single product family or a limited range of product families.

The research of FAS in 1990-2000 showed number of weak points in this technology. The main disadvantages according to Whitney and Katalinic, (Whitney, 2004), (Katalinic, 2000):

- FAS is too complex, which makes it hard to produce and control
- FAS is effective when applied for assembly with limited aims, complexity and scope
- Too many tools are needed to make an additional number of operations
- Problems of scheduling and dispatching tools were not anticipated. The assembly lines behavior makes this systems dependant on the tact time and opposes the routing flexibility.
- FAS did not achieve claimed productivity. One of the reasons is routing inflexibility and scheduling problems.
- FAS were too expensive when compared with assembly lines or job-shops for the same range of products

Further development of assembly systems was made in the direction reconfigurable assembly systems inspired by following ideas:

- flexibility (Suarez at al, 1991), (De Toni & Tonchia, 1998)
- agility (Yusuf et al., 1999)
- self-organization (Katalinic, 2002).
- changeability (Wiendahl & Heger, 2003), (Zaeh at al. ,2005),
- mutability (Spath & Scholz, 2007)
- evolution and learning (Monostori at al., 2006), (Bi at al, 2008).
- reconfigurability (Ueda, 2014)

To address these requirements of the assembly domain, new assembly system concept, based on self-organization phenomena was proposed by Katalinic. A concept of Bionic Assembly System (BAS), as a self-organizing system was developed based on real industrial demand to significantly reduce the production costs in mass production (Katalinic, 2002).

2.3 Self-organization in nature and technology

Self-organization – it's a term, which cannot be explicitly defined. Connected to engineering, the definition of self-organization is the ability for the number of working elements to behave as a system without any leading element or external control. Examples from biology, physics, and economics can prove that such a system can exist.

The term was introduced in science in 1947 by W. Ross Ashby. Specifically, self-organization refers to systems in which the internal dynamics tend to increase order (Ashby, 1962).

To get the definition of self-organization, the definition of organization must be considered.

First of all, organization is an "act or process of putting the different parts of something in a certain order so that they can be found or used easily". Second – "the way in which the different parts of something (such as a company) are arranged" (**, 2014).

The synonyms of the organization are *assembling, construction, coordination, design, formation, grouping, harmony, management, methodology, organism, pattern, plan, planning, regulation, standard, standardization, structure, structuring, symmetry, system, unity, whole* (**, 2014).

Using this definition with prefix self-, we'll get a general definition of self-organization: the way in which the parts are arranged into a system, without any external control. There are a lot of definitions in the different spheres of science; main of them will be shown below.

Another description is given by S. Camazin: "Self-organization is a process whereby pattern at the global level of a system emerges solely from interactions among the lower-level components of the system. The rules specifying the interactions among the system's

components are executed using only local information, without reference to the global pattern." (S. Camazin, J.-L. Deneubourg, 2001).

In other words, self-organization is the ability of the system to make an order out of chaos without any external exposure.

2.3.1 Examples in nature and science

There is a number of examples of self-organization in different sciences (Haken, 2006), (Eigen, 1978)

- **Physics.** Self-organization is usually described for instance in laser phenomena: at certain wavelengths, bouncing light forms into the structured form of a laser. Or an example of ferromagnetic material, which becomes spontaneously magnetic when the temperature is below the certain level.
- **Biology.** In biological systems self-organization is a process, in which pattern at the global level of a system emerges solely from numerous interactions among the lower-level components of the system. Moreover, the rules specifying interactions among the system's components are executed using only local information, without reference to the global pattern (Camazine, 2003). Examples of the self-organization in nature can be bird flocking or ants colonies.
- **Economics.** In economics self-organization can be seen in the Alan Smith's law, or the theory of invisible hand.
- **Informational Technology.** In IT the world-famous program and protocol torrent use the principles of the self-organization. BitTorrent speeds up the distribution of large files over the net by breaking information down to smaller packets, and splitting up the burden of bandwidth among those seeking to download the same file. In BitTorrent, unlike other programs complete use of the bandwidth available. DHT (Distributed hash table) – protocol, which allows bittorrent clients to find each other without using external data or tracker. Clients with DHT organize their own network and support each other with the search for the needed information.

2.3.2 Self-organization in assembly domain

Usage of self-organization phenomena in a field of assembly systems brings a number of advantages. It helps to increase system flexibility with increase of a number of working shop floor elements. Also it helps to distribute system intelligence to whole system. Therefore the control system complexity will decrease.

The main challenge of self-organizing systems introduction is a conflict between sub-ordinated control subsystem and self-organizing shop floor. Introduction of new between the sub-systems will help to overcome this challenge.

Chapter 3

Flexible Assembly System for AC motors

This chapter describes facility for AC motors assembly, realized as flexible assembly system (FAS). FAS is a hybrid system in which most of assembly stations are linked together by a material-handling system and the system units behavior is controlled by a central computer. A flexible assembly system is designed to assemble products within a defined range of products with different sizes, forms and processes (Rosati et al., 2011).

Analyzed FAS placed in the facility deals with assembling of electrical motors. This is a type of electric motor, which runs on alternating current (AC). AC motors are more commonly used in industry as well as in home appliances. The assembled motors are used for home appliances (e.g. dishwashers or washing machines).

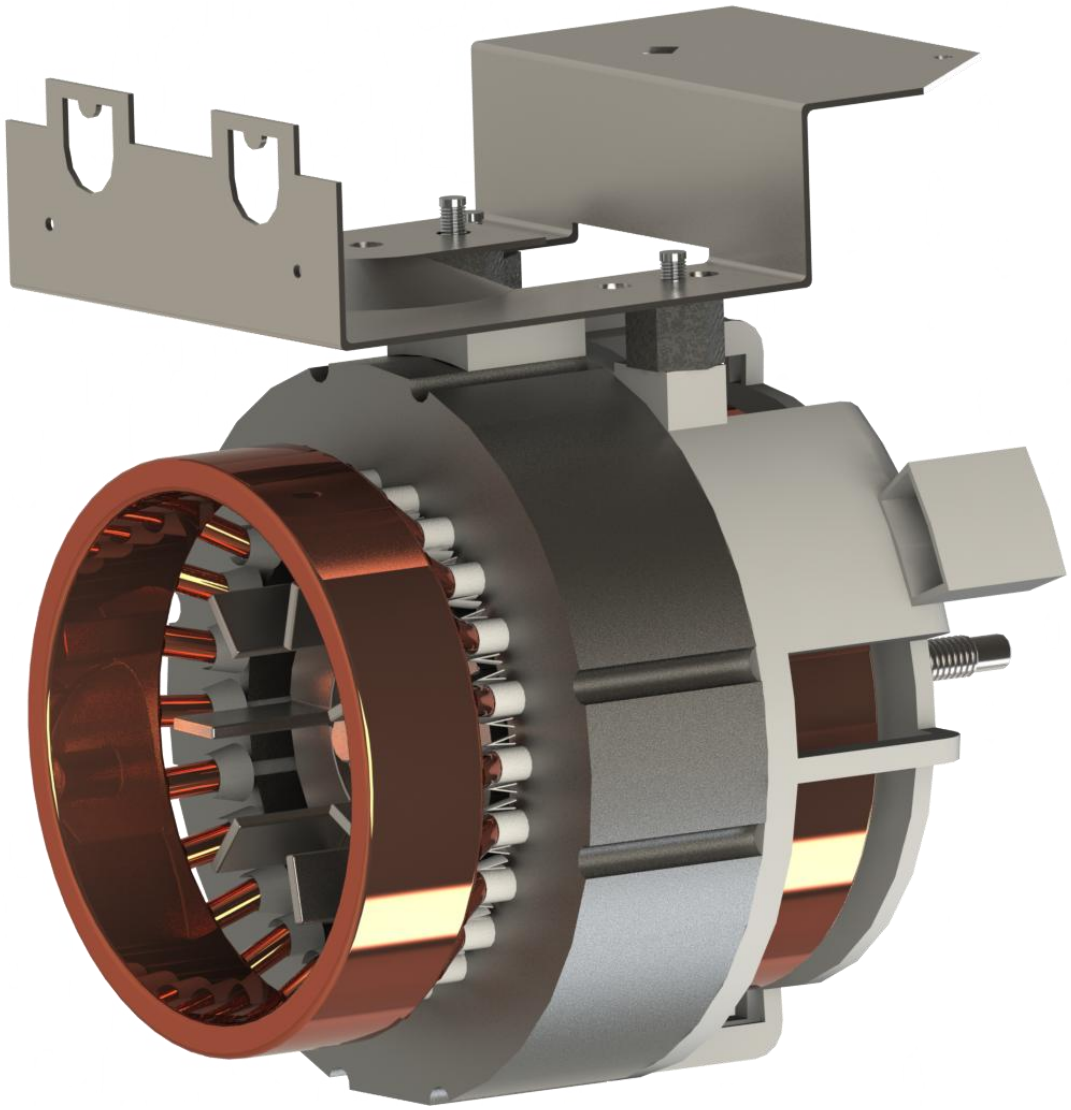
Complex analysis of FAS includes the product analysis and the description of FAS structure, subsystems, components and their functions.

3.1 Product Analysis

FAS assembles the AC motors of two families: one-sided and double-sided AC motors. The system covers assembly of 340 types of motors within double-sided motor family and 60 types of one-sided motor family. The type differences could be observed in a parts design, size, and presence of additional elements. The most complex assembled motor is shown at Fig. 3.1.

A life time of one motor type is about 2 years. After this period of time the assembly of this motor type is reduced or cancelled. The facility is also able to assemble new product types within these two motor families. That means that the range of electrical motors is permanently changing.

a)



b)

Fig. 3.1. Assembled AC motors: a) double-sided AC motor with capacitor and fan; b) one-sided AC motor with fixing plate

In order to keep the price low, the parts and components of motors, which are bought at market, have not 100% guaranteed quality. Acceptable number of defective parts varies in the relation 1 bad to 1000 good parts. Some mistakes also occur during the assembly process. For this reasons there is always a number of defective motors. The acceptable mistake rate for this facility is 2% (Katalinic, 1999). Some of these motors are repaired and some of them have to be produced additionally to cover the irreparable ones.

3.1.1 Double-sided AC motor structure

Structural parts of double-sided motors are shown in a Fig. 3.2. The example shows double-sided AC motor with capacitor and fan. It consists of 14 basic parts, some of them are pre-assembled.

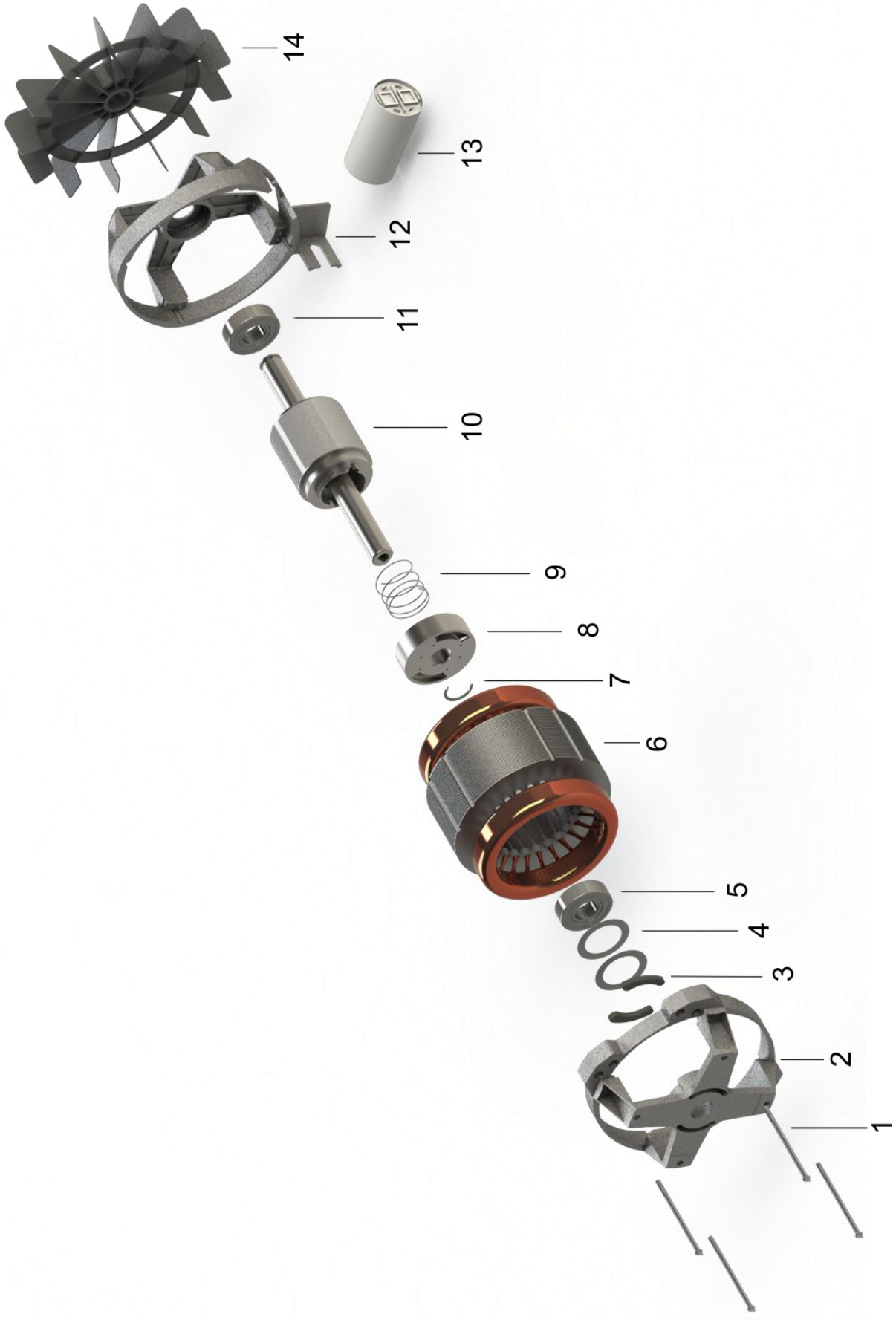


Fig. 3.2. Structure and parts of AC double-sided motor

Table 3.1 contains part names and descriptions. Some parts could have also different names in a literature. For proper identification German part names and synonym English names from literature are added to a description.

Part description is based on Glossary of Motor Terms from (AB, 2014) and Electric Motor & Machines Terminology from (****, 2014).

Table 3.1. Names and Description of AC double-sided motor parts (AB, 2014)

N	**Part name (English)**	**Part Name (German)**	**Part description**
1	Mounting Bolts	Befestigungsbolzen	Mounting Bolts are used to keep the motor parts together.
2	1st End Shield	Lager-schild	This part is also called end bell, cover plate, end backet etc. The part of the motor housing, which supports the bearing and acts as a protective guard to the electrical and rotating parts inside the motor. At one-side motor, a perforated cover may leave the rotor exposed, and only the stator windings are protected.
3	Brake Lining	Bremsstoff	Brake linings are used to apply a friction forces against the brake cover to stop a motor.
4	Leveling ring	Ausgleich-scheibe	This part is also called spacer, washer, balancing disc, etc. A small flat ring made of metal is used as a spacer to align a rotor and stator.
5	Ball-Bearing	Kugellager	A "ball" shaped component that is used to reduce friction and wear while supporting rotating elements. For a motor, this type of bearing provides a relatively rigid support for the output shaft. The bearing acts as the connection point between the rotating and stationary elements of a motor.
6	Stator	Stator	That part of an induction motor’s magnetic structure, which does not rotate. It usually contains the primary winding. The stator is made up of laminations with a large hole in the center in which the rotor can turn; there are slots in the stator in which the windings for the coils are inserted.
7	Safety	Sicherheits	Safety rings could be used for keeping ball-bearings

	Ring	ring	and brake construction together. Normally two rings placed on 180 degrees to each other are used.
8	Brake Cover	Bremstopf	Brake is an external device or accessory that brings a running motor to a standstill and/or holds a load. Can be added to a motor or incorporated. Brake cover overlays a breaks and it rubs on a brake linings to stop the motor.
9	Spring Coil	Feder	A spring coil pushes the brake cover to brake linings to stop the motor.
10	Rotor	Rotor	The rotating member of an induction motor with a shaft. Current is normally induced in the rotor, which reacts with the magnetic field produced by the stator to produce torque and rotation.
11	Ball-Bearing	Kugellager	see 5
12	2nd End Shield	Lagershild	see 2
13	Capacitor	Konden-sator	It is a device which, when connected in an alternating current circuit, causes the current to lead the voltage in time phase. The peak of the current wave is reached ahead of the voltage wave. This is the result of the successive storage and discharge of electric energy. Capacitor value and voltage rating are essential to the proper motor operation.
14	Fan	Ventilator	During a normal work, motors generate a heat. The fan is the cooling device which is used to prevent the motor temperature to rise above a specified value.

3.1.2 One-sided AC motor structure

Structural parts of one-sided motors are shown in a Fig. 3.3. The example shows double-sided AC motor. It consists of 8 basic parts, some of them are pre-assembled. Table 3.2 contains part names and descriptions.

A main construction difference of two motor families is a number of end-shields. One-side motors have one end-shield. Because of this construction there is no need for the balancing discs and additional brakes. This motor is fixed vertically. Because of this construction, additional side fixing plate is used to fix this motor to the driven equipment. This motor

family also doesn't have additional parts, such as fan and capacitor. The rotor includes blades on one side and cools motor down by its own rotation.

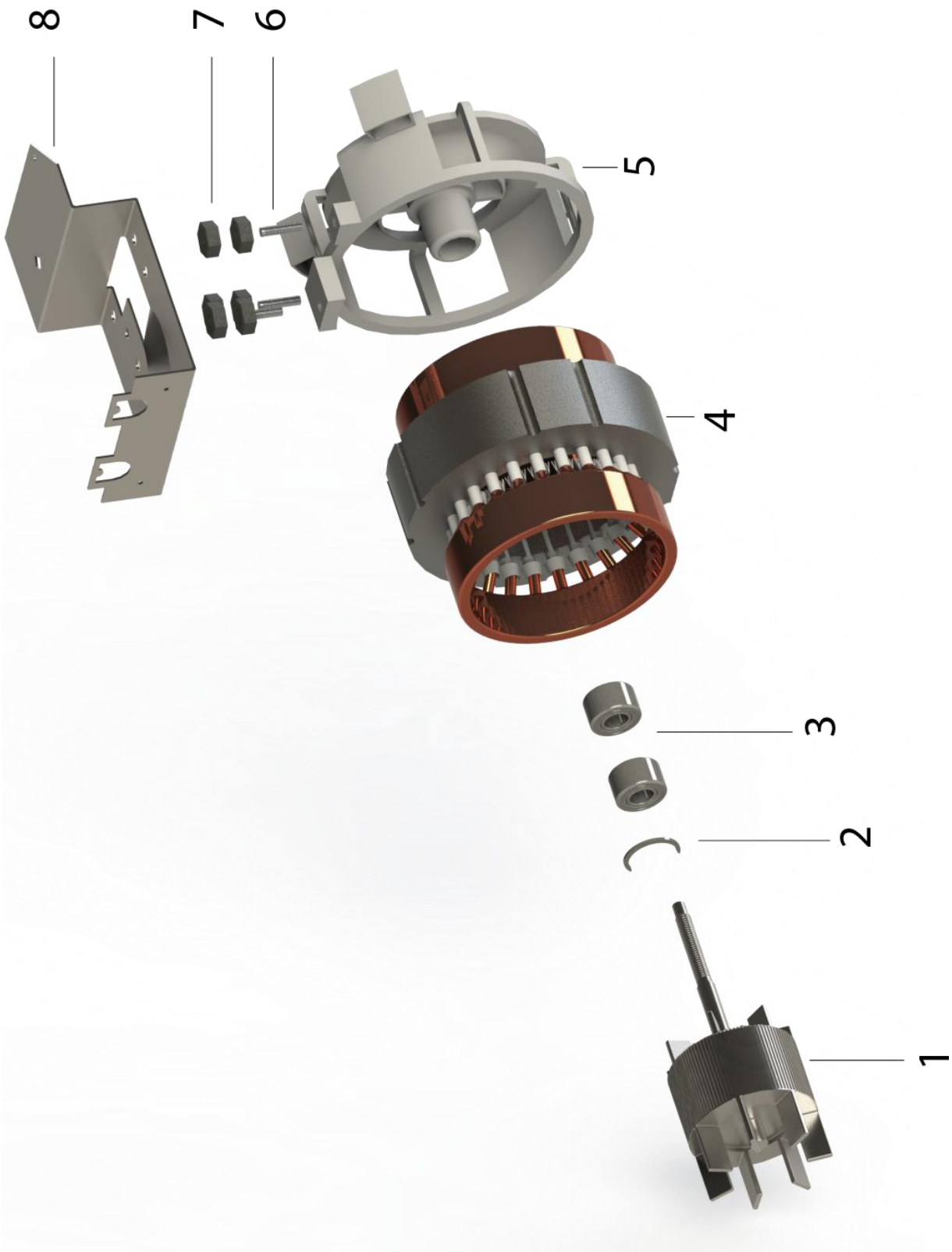


Fig. 3.3 Structure and parts of AC one-sided motor

Table 3.2. Names and Description of AC double-sided motor parts (AB, 2014)

N	Part name (English)	Part Name (German)	Part description
1	Rotor	Rotor	The rotating member of an induction motor with a shaft. Current is normally induced in the rotor, which reacts with the magnetic field produced by the stator to produce torque and rotation.
2	End Shield	Lager-schild	This part is also called end bell, cover plate, end backet etc. The part of the motor housing, which supports the bearing and acts as a protective guard to the electrical and rotating parts inside the motor. At one-side motor, a perforated cover may leave the rotor exposed, and only the stator windings are protected.
2	Safety Ring	Sicherheits ring	Safety rings could be used for keeping ball-bearings and brake construction together. Normally two rings placed on 180 degrees to each other are used.
3	Ball-Bearing	Kugellager	A "ball" shaped component that is used to reduce friction and wear while supporting rotating elements. For a motor, this type of bearing provides a relatively rigid support for the output shaft. The bearing acts as the connection point between the rotating and stationary elements of a motor.
4	Stator	Stator	That part of an induction motor's magnetic structure, which does not rotate. It usually contains the primary winding. The stator is made up of laminations with a large hole in the center in which the rotor can turn; there are slots in the stator in which the windings for the coils are inserted.
6	Mounting Bolts	Befestigun gsbolzen	Mounting bolts are used to keep the motor parts together.
7	Rubber buffer	Gummipuff er	This element is used for the proper placement of fixing plate and overcome the vibrations.
8	Fixing Plate	Befestigun gsplatte	This element is also called mounting plate or surface. It is machined to standard dimensions, with holes to allow easy, secure side mounting to driven equipment.

3.1 Structure of FAS

FAS is composed from three main subsystems: core, supplementary and control subsystems as shown in Fig. 3.4.

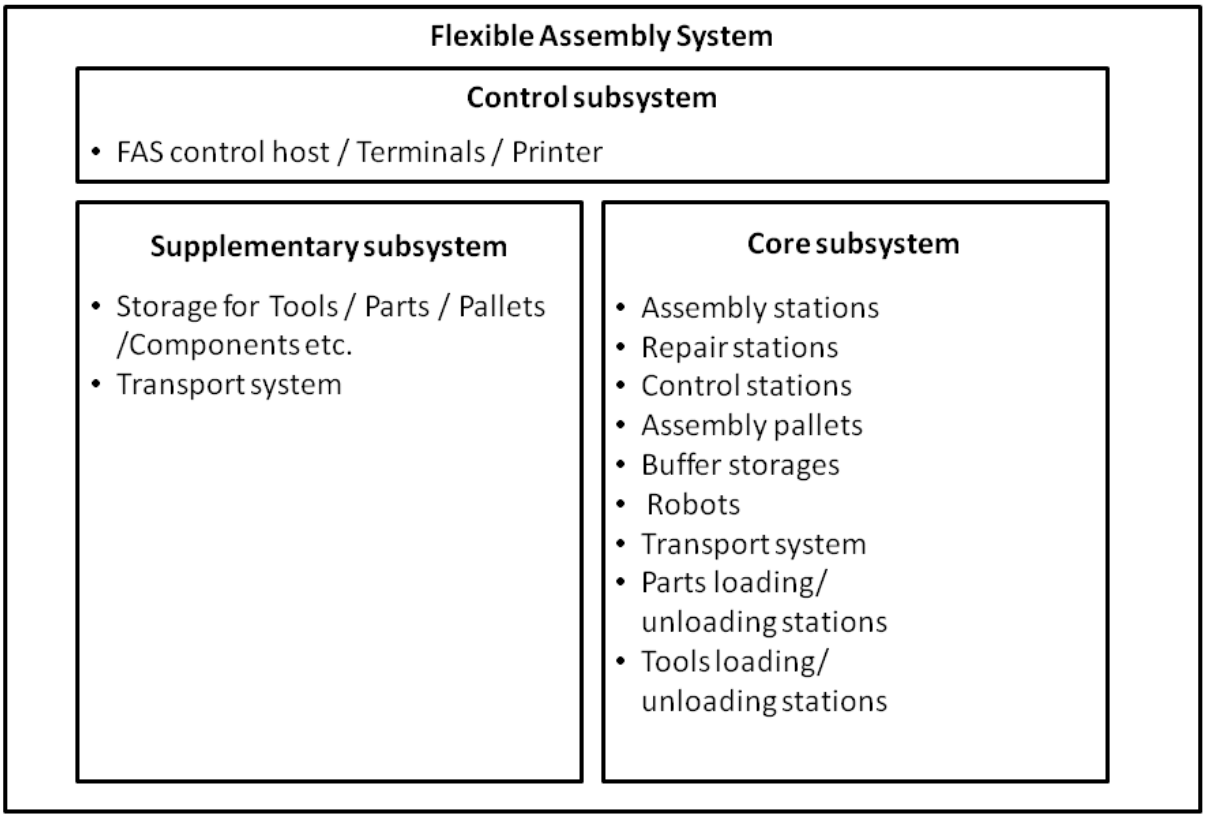


Fig. 3.4. FAS subsystems and structure

3.1.1. FAS core subsystem

FAS core subsystem includes two parts for one- and double-sided motors assembly. They are realized as separate systems within one core subsystem with shared material supply.

Material flow. Whole material flow (parts, tools, products) is organized with pallets. There are two types of material flow: internal and external. Internal material flow is going between the stations on assembly pallet. External flow is organized between core and supplementary subsystems to ensure the stations supply with parts and other means. Other function of external flow is taking assembled products out of the system.

Operators. Work of operators is required for the normal FAS functioning. Typical operator tasks are: setups, ensuring and supporting material flow, correction of interruptions, operating of central FAS computer.

Buffers. The core subsystem of FAS includes number of buffers, which are used for temporary storage of parts, work-in-progress inventory and other means.

Assembly of double-sided motors family. A layout of assembly system for double-sided motors family is shown in Fig. 3.5. It includes 24 stations including assembly stations, quality control stations, packing stations and pallet storage, connected with transport system. A system tact time for double-sided assembly is 24 seconds and 9 seconds for quality control. Assembly sequence is shown in Table 3.3.

Both families are assembled on FAS according to Just-in-Time principles. It means that the exact number of motors must be prepared at fix date and time for delivery to one particular customer according to the specification. A size of assembly run depends on a customer needs and could vary from fifty to several thousand motors. Maximal production of this facility is 4000 motors per daily. System productivity strongly depends on the work of operators. Stations setups as well as some assembly operations are done manually. Therefore, maximal production could be achieved only with two working shifts per day.

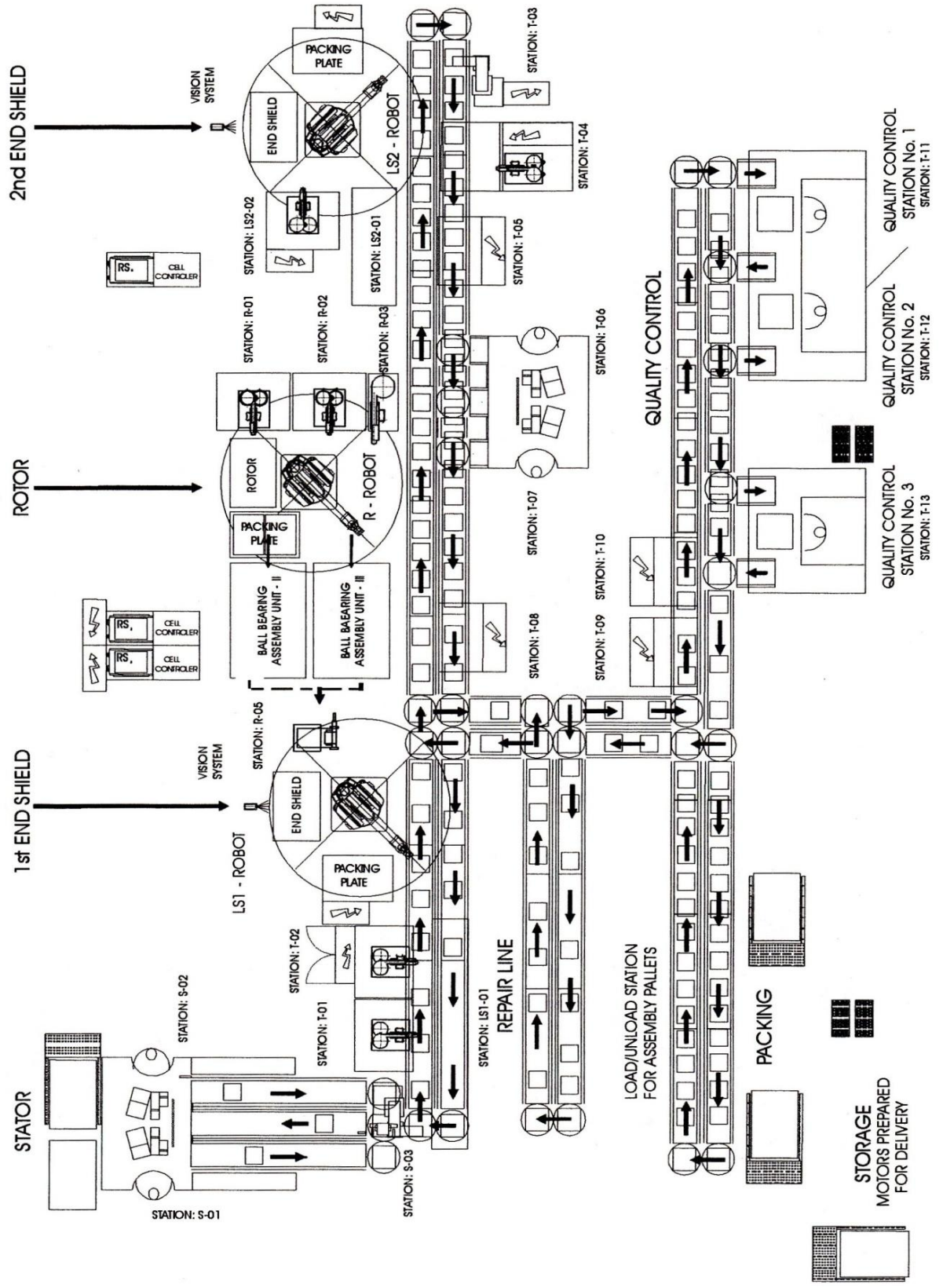


Fig. 3.5. Layout of assembly system for AC double-sided motors family (Katalinic, 1999)

Table 3.3. Assembly steps for AC double-sided motors family

N	Name of the Station	Function Description
1	Pallet Storage	Pallet Loading / Unloading. Pallet is assigned to a mobile robot according to the assembly order and taken out, when robot order is completed
2	1st End shield Station	The 1st End shield is placed into an empty pallet according to a Motor Type
3	Heating Station	Tunnel with a hot air. Pallet with an End Shield comes to a tunnel, where it is pre-heated with a temperature from 40 to 80C, depending on the motor type. Due to a thermal expansion end shield gets an optimal size for an assembly
4	Stator Pre-assembly Station	Stator is manually pre-assembled
5	Stator Measurement and Assembly Station	Pre-assembled stator is measured by height and put into a pre-heated End shield
6	Balancing Disc Station -1	Depending on a stator size a needed number of balancing discs are inserted. On some type of motors there is a need only for balancing discs from station 1, only station 2 or from both of the stations.
7	Balancing Disc Station - 2	
8	Rotor Pre-Assembly Station	Rotor is pre-assembled. Rotor is assembled with spring, disc brake, safety rings (180 degrees to each other, mounted together), ball bearings are inserted
9	Rotor Assembly Station	Pre-assembled rotor is put into a End shield with a stator and balancing discs
10	2nd End shield Pre Assembly Station	2nd End shield is pre-heated with hot air 40-80C, 2 brake lining are pressed into an end shield with a force (5-50 kg) depending on a motor type.
11	2nd End shield Assembly Station	Preheated and pre-assembled 2nd End shield is placed to pre-assembled motor
12	Rotating Station	Motor is turned upside down for a screwing procedure
13	Fixture of Motor and Electrical Components -1.	Bolts are sorted separated from each other. Loctite glue is inserted to a thread of a bolt.

	Preparation of bolts	Bolts are inserted to the holes in the end shield
14	Fixture of Motor and Electrical Components - 2	Motor is fixed, pressed together with a force (50-1000kg) depending on a motor type and screwed
15	Condenser Assembly - 1	Assembly of condenser, depending on a motor type - parallel stations
16	Condenser Assembly - 2	
17	Ventilator Assembly	Assembly of ventilator, depending on a motor type
18	Quality Control -1.1	Examination of axial clearance and measurement of connection dimensions. These stations work as parallel stations
19	Quality Control -1.2	
20	Quality Control - 2.1	Examination of electrical parameters of the motor
21	Quality Control - 2.2	Examination of electrical parameters of the motor
22	Quality Control - 3	Grinding brake linings
23	Unloading/Packaging Station	Completed motor is taken out of a pallet and get packed
24	Repair Line	This line is responsible for motor repairs. If it is not possible to repair a motor, it removes motor and restart of assembly pallet

Assembly of one-sided motors family. A layout of assembly system for one-sided motors family is shown in Fig. 3.6. Because of differences in a motor families construction there are differences in assembly process. In a system layout of one-sided motor assembly system there are 22 stations including assembly stations, quality control stations, packing stations and pallet storage.

System tact time for assembly is 18 seconds and 9 seconds for quality controls. Assembly sequence is shown in Table 3.4.

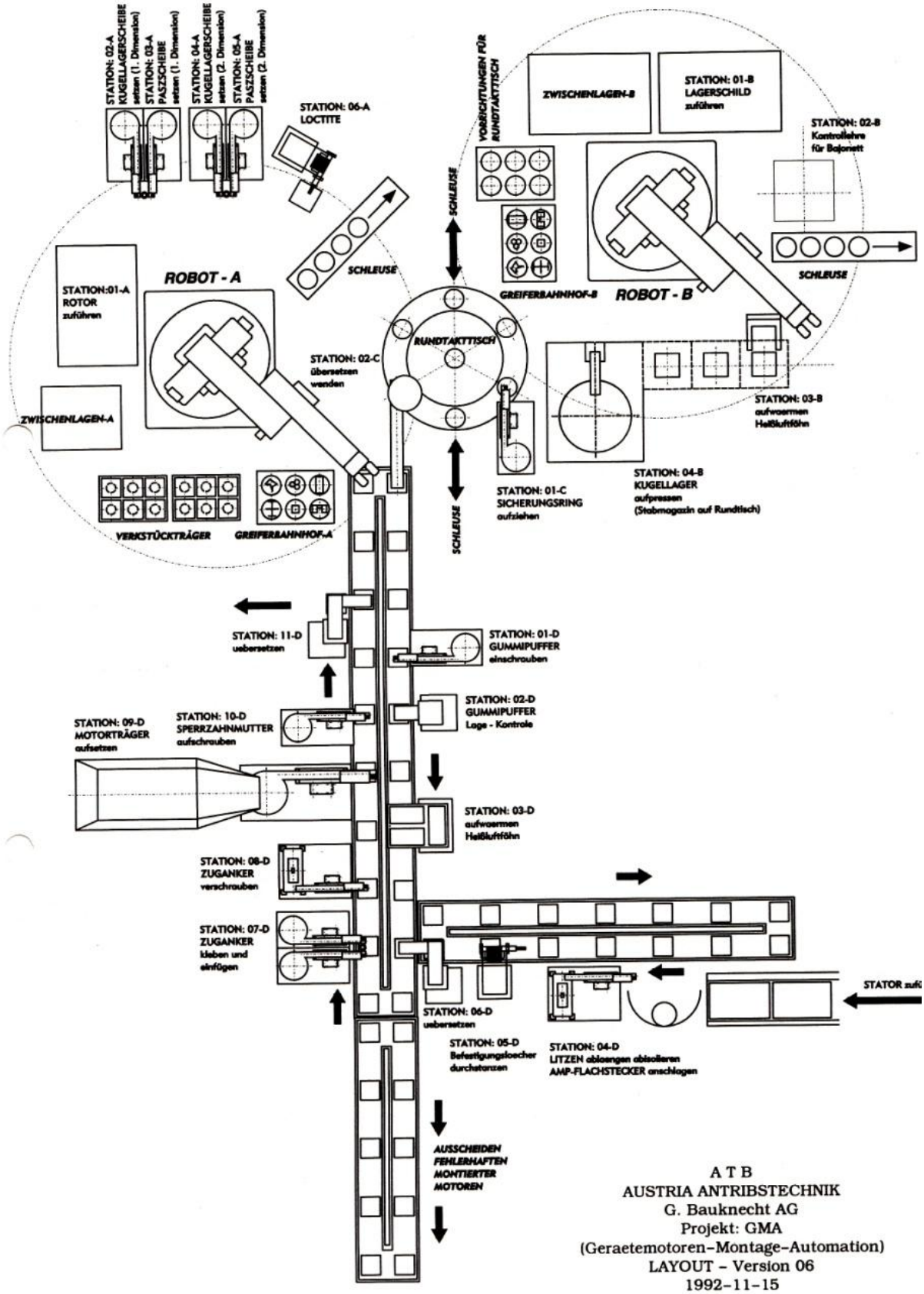


Fig. 3.6. Layout of assembly system for one-sided AC motors family (Katalinic, 1992)

Table 3.4. Assembly Steps for one-sided AC motors family

N	Name of the Station	Function Description
1	Pallet storage	Pallet loading / unloading. Pallet is assigned to a mobile robot according to the assembly order and taken out from a finished robot
2	End shield station	The end shield is placed into an empty pallet according to a motor type
3	Heating station	Tunnel with a hot air. Pallet with an end shield comes to a tunnel, where it is pre-heated with a temperature from 40 to 80c, depending on the motor type. Due to a thermal expansion end shield gets an optimal size for an assembly
4	Ball bearing pressing	Ball-bearing is pressed to a preheated stator
5	Ball bearing safety ring	Ball bearings are fixed with a safety ring
6	Rotor pre-assembly station	Rotor is pre-assembled Rotor is assembled with spring, disc brake, safety rings (180 degrees to each other, mounted together)
7	Rotor assembly station	Pre-assembled rotor put into a end shield
8	Rubber buffers screw	Bolts are sorted separated from each other. Bolts are inserted to the holes in the end shield Rubber buffers are screwed with the end shield
9	Rubber buffers control	Rubber buffers are controlled
10	Heating station	Tunnel with a hot air. Pallet with an end shield comes to a tunnel, where it is pre-heated with a temperature from 40 to 80c, depending on the motor type. Due to a thermal expansion end shield gets an optimal size for an assembly
11	Stator pre-assembly station	Stator is manually pre-assembled
12	Stator measurement and assembly station	Pre-assembled stator is put into a pre-heated end shield
13	Fixture of motor and electrical components - 1. Preparation of bolts	Bolts are sorted separated from each other. Loctite glue is inserted to a thread of a bolt. Bolts are inserted to the holes in the end shield
14	Fixture of motor and electrical components - 2	Motor is fixed, pressed together with a force (50-1000kg) depending on a motor type and screwed
15	Fixing plate preparation	Fixing plate preparation

16	Fixing plate assembly	Assembly of fixing plate to a motor
17	Quality control -1.1	Examination of axial clearance and connection dimensions
18	Quality control -1.2	Examination of axial clearance and connection dimensions
19	Quality control - 2.1	Examination of electrical parameters of motor
20	Quality control - 2.2	Examination of electrical parameters of motor
21	Unloading/packaging station	Controlled motor is taken out of a pallet and get packed
22	Repair line	Repairing of motors if not possible removing of motor and restart of assembly pallet

Analyzed systems have a number of similar stations. Systems analysis shows that there are also similarities in the assembly processes for the both families. Technologically it is possible to combine both systems into one by adjusting the similar stations and adding missing stations for a two family's assembly.

3.3 FAS control system and scheduling

FAS control structure is shown in Fig. 3.7. The design aim of FAS is minimum amount of lost station time (Katalinic, 1992). That means the highest stations utilization within assembly run. This aim is realized by fulfilment of three sub-aims:

1. **Maximal stations utilization.** All the parts, facilities, operators etc. must be present when required.
2. **Minimal stations setup time.** Assembly sequences must be planned in a way to minimize the time needed for setup.
3. **Minimal stations breakdown time**. If any station stopped working for any reason, it has to be brought back as soon as possible. This has to be organized in a way to keep the minimum of idle stations.

FAS Planning. Whole FAS planning is organized in order to maximize system productivity during the assembly of one product run. Maximal productivity means to assemble maximal number of products during one particular period of time. The planning has to take into consideration number of conditions, such as:

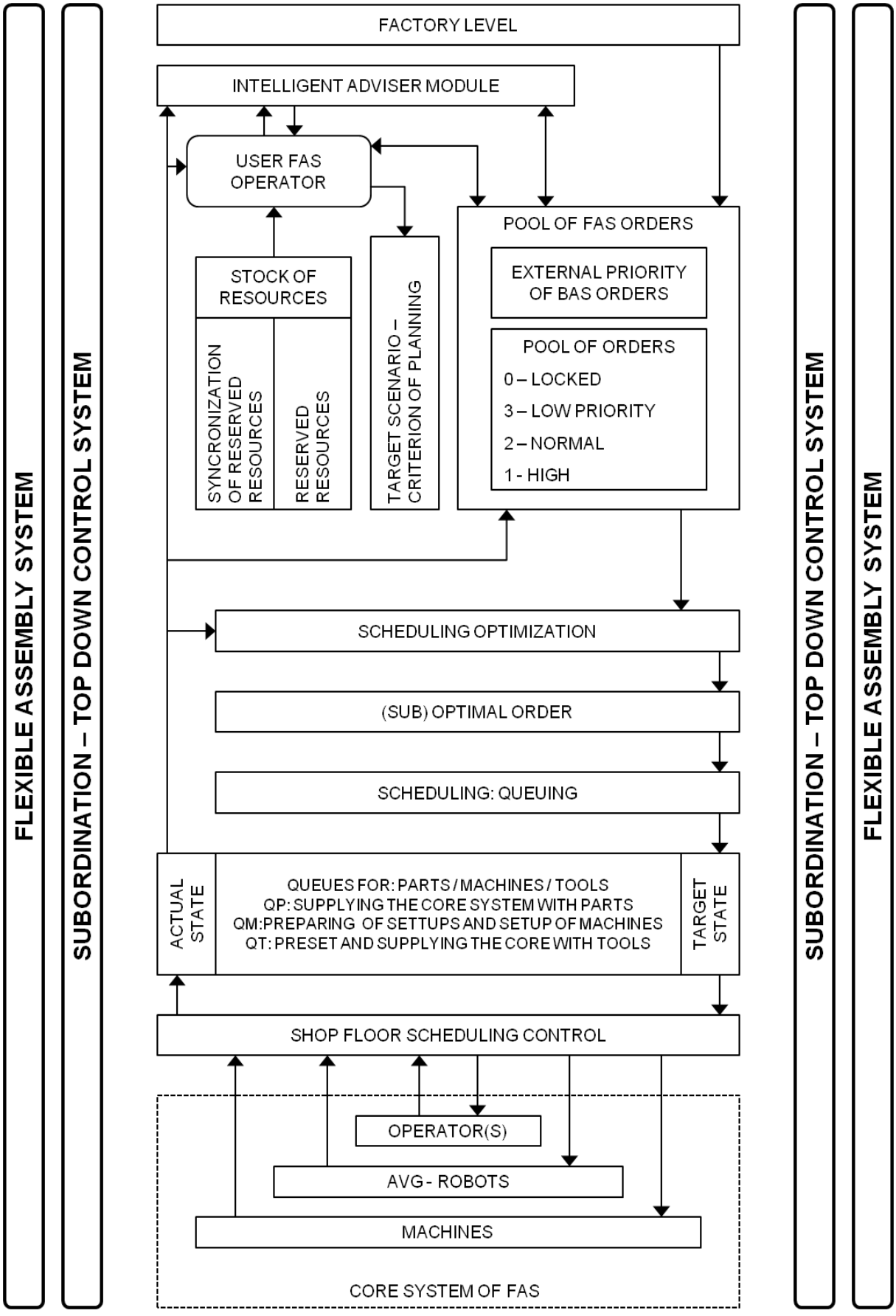


Fig. 3.7. FAS control structure (Katalinic, 1992)

- external priority of FAS orders
- system bottlenecks
- limitations in the number of production facilities
- limited capacity of assembly station, etc.

It is necessary to carry out of all above mentioned activities in as short a time as possible to achieve the above-mentioned aim. The work of all core subsystem elements: assembly stations, robots and operators has to be synchronized according to FAS working scenario. All planned activities of assembly stations, automatic transport system and operators will be arranged during this planning process.

Just-In-Time output. Delivery of assembled products is organized according to Just-in-Time principle. The Finished-Product is produced and delivered to one particular customer according to his specification of delivery target date (YYYY-MM-DD-HH-MM), type and volume.

Factory Level. The production planning system at factory level is responsible for the highest level planning. That means volume of production, mix of the product run and the time frame. The time frame could differ from a day to the year, depending on the order size. The main task of planning system is to decide what FAS has to produce in next given period of time. There are many methods, criteria and solutions for this high-level planning. FAS planning is done according to the predetermined aims. This aim could be:

- maximal profit
- maximal stations utilization
- minimal work-in-progress (WIP)
- a maximal production volume

When the aims are opposed to each other, the system searches for a compromise solution.

Pool of FAS orders. This pool acts as an interface between factory planning system and FAS. To assemble the products to the pre-planned time, the orders have to be prioritized. Therefore each FAS order has its own an external. The priority scale has 3 points, from 0 to 3. Priority 0 is given to the orders, which are locked for assembly. The normal urgency is priority 2, a very urgent order is priority 1, and a not urgent order is priority 3.

Scheduling optimization module. Scheduling optimization module is responsible for the orders allocation. It has search for the most suitable FAS order from the pool of orders considering a number of conditions, such as: target scenario, criterion of planning, actual state of FAS, free and reserved resources of system during the time planed. In the optimization procedure the (sub)optimal (sequence of) order(s) is determined.

(Sub)optimal order. This order is a result of optimization process. It is build in virtual scenario of FAS in the case of simulation or in working scenario of FAS in the case scheduling planning. The results of this planning are used for queues of orders formation. This queues

determine the operation sequences on assembly stations, the flow of pieces through the FAS, tool flow and set-up activities schedule.

Intelligent Adviser Module. For efficient scheduling of FAS, the human system operator as a user of FAS has to take a number of decisions to solve the complex situations in a right time. The speed of reaction and quality of decisions influence directly the efficiency of FAS. The intelligent adviser module is designed as interactive support for FAS system operator (Katalinic 1997), (Nanasi, 1996). This module includes:

- FAS specific knowledge and experience for the past work
- Evaluation of sensors data
- Evaluation of actual FAS data
- FAS simulation data

The operator will be supported during:

- Diagnostics: assembly stations diagnostics, equipment diagnostics, orders execution diagnostics, tool and system diagnostics, etc.
- Optimization during the scheduling planning
- Projection of alternative virtual scenarios
- Optimization during the scheduling of FAS and search for solutions in complex conflict and chaotic situation, etc.

The necessary data containing the information on the state of the FAS system can be transferred directly over information-flow structure of FAS. Specific data can be given to the module in one interactive dialogue with the operator or over intelligent sensor system.

Chapter 4

Bionic Assembly System

Bionic Assembly System (BAS) represents an assembly system from the next generation. An initial concept of this system was developed and described by Katalinic (2001). BAS initial layout is shown in a Fig. 4.1. The main task of this system is to facilitate assembly of a family of different products. The complexity level of the products is limited to a complexity of AC electrical motor. During the assembly process a product could have the following states: *Product-In-Assembly, Finished-Product, Product-In-Repair*.

Product Orders come from different customers and a delivery of Finished-Products has to be done according to Just-in-Time principle. The Finished-Product has to be delivered to one particular customer according to his specification of delivery target date (YYYY-MM-DD-HH-MM), type and volume.

Following ideas presented in this chapter are discussed in recently published articles:

- The description of BAS working scenarios and strategies (Katalinic et al, 2012)
- Possible reconfigurations within the system (Kukushkin et al, 2011)
- BAS Hybrid control structure (Katalinic et al., 2013)

4.1 BAS basic characteristics

BAS is developed with a goal to upgrade the traditional FAS. Therefore BAS has to answer a number of FAS requirements, discussed in previous chapter. Following BAS characteristics are designed to cover those requirements:

- **Wide range of products.** The system is able to assemble open range of product types. This range can be adjusted anytime.
- **Product lifetime.** The system is suitable for assembly of products with a short lifetime. It means that the product lifetime is limited and shorter than the lifetime of the assembly system.

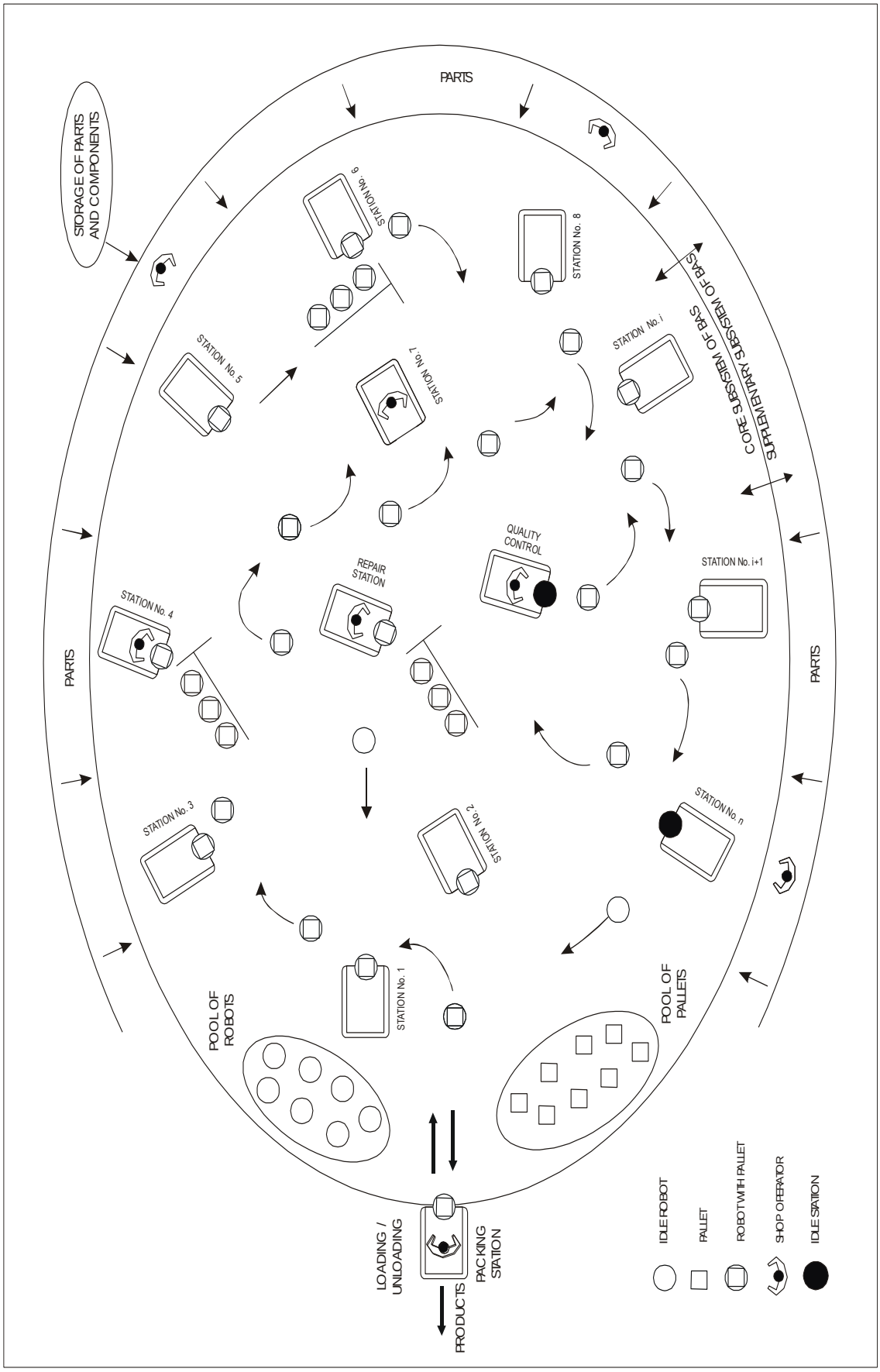


Fig. 4.1. Layout of a Bionic Assembly System (Katalinic, 2001)

- **Assembly run size.** The system covers assembly runs from one to thousands of products.
- **Working mode.** The system supports continuous working mode during longer period of time as well as work in shifts.
- **Incoming quality control**. The system rejects the parts, which do not fulfill the quality standard. Therefore, only the parts which do, are assembled into a product. This ability of the system allows buying the parts from suppliers at lower price.
- **In-process quality control:** High Finished-Product quality is achieved only if all assembly operations are successfully completed. The possibility of assembling the product increases by applying in-process quality control. That implies a quality check after each assembly operation. If in-process quality control passes, assembly process continues. Otherwise, the Product-In-Assembly goes to repair station.
- **Quality standard:** During the assembly of product, faults can occur in some percentage of products in the run. The system must organize the continuous correction of mistakes in order to get the exact number of guaranteed high-quality Finished-Products
- **Integration of workers:** The system has to be able to easily integrate workers. There should be possibility to use workers as operators for the normal assembly tasks and special tasks (repair, inspection, supply of parts, set up, etc).
- **Stations variation:** The number and type of assembly stations has to vary. The set-up of stations and material / parts supply can be carried out, at least partly, manually.
- **System autonomy:** The disturbances that come from the outside have a limited negative influence on the efficiency and function of the system.
- **Diagnostic:** The system has to have a very detailed and broad diagnostic. That allows the possibility to track the system state, orders and resources. Another function of diagnostic is to identify and report problems within the system.
- **Prognosis:** The system has to have the possibility to develop alternative production scenarios, based on the actual state of the system, for a period of time in advance. The period of time can be chosen freely.
- **Internal reserves and reaction of the system:** Reserves have to be accumulated in the system and they have to be used for reducing the negative influence of outer disturbances. The system reacts on the disturbances flexibly and efficiently under normal and non-typical conditions.
- **Interface between BAS and the environment:** The information and material flow between BAS and its environment is organized in a way that each transaction can be reconstructed.
- **Optimization of production process:** In the optimization of BAS's work, the system has the possibility of choosing the most suitable working scenario.

4.2 BAS components and subsystems

BAS layout consists of two subsystems: the core subsystem and the supplementary subsystem. Elements of BAS subsystems are shown in a Table 4.1.

Tab. 4.1. Elements of BAS subsystems

Core subsystem	Supplementary subsystem
Assembly Stations	Stations Storage
Mobile robots	Mobile Robot Storage
Operators	Operators Rooms
Unloading station	Finished-Product Storage
Packing station	Packing material Storage
Quality control station	Tools Storage
Repair station	Parts Storage

Dominating activities in the core subsystem are assembly, quality control, repair of defective Products-In-Process and packaging of Finished-Products. The supplementary subsystem surrounds the core subsystem. The main activity in this subsystem is the storage of parts and components. There is an intensive material exchange between these subsystems. BAS transport system is responsible for this flow realization. The flow includes parts, tools, setup components etc. The functions of elements, which take part in assembly operations, are shown below.

Operators. Operators are the shop floor workers. They are responsible for stations set ups, ensuring and supporting the material flow in the system and correction of interruptions during production.

Manual Assembly stations. In some cases operators could take part in assembly process. In this case they act as manual assembly station.

Assembly stations. Assembly stations are the machines with a wide range of flexibility. They can complete one or more assembly operations on one or more different products. Also there can be more than one assembly station dedicated to one particular assembly operation. Additionally assembly stations are suitable for one or more types of mobile robots.

Assembly pallet. The assembly pallet is the basic carrier for the product during assembly cycles. Assembly pallets are stored in a pool of pallets and moved by the mobile robots. Each

pallet has an information tag, containing Pallet-ID and quality state of pallet. Pallet ID consists of pallet type and number. When the pallet is loaded on the robot, the tag gets scanned. Robot can read Pallet ID and quality state information from this tag.

Mobile robots. Mobile robots are autonomous machines, which transport assembly pallets between assembly stations. There are three possible robot states:

- Switched-off - robot is not participating in assembly operations before it is switched on by shop floor scheduling control unit
- Stand-by - robot is ready to participate in assembly operations, but has no active assembly order
- Active - robot has active assembly order

Switched-off robots are located at robot storage in supplementary subsystem. Standby robots are located in the pool of robots. They constantly check if there is a new robot assembly order. Robot assembly order represents a task to assemble one product. When robot gets an assembly order it activates and drives out to take pallet of needed type. Robot can fulfil this order using different assembly trajectories. It is possible, that one assembly operation can be completed by alternative assembly stations. To select the most suitable station for the next assembly operation the robot follows "the smallest time resistance" criteria for the next assembly operation.

Repair station. Not every assembly operation could be completed successfully. In order to continue assembly it is necessary to correct occurred errors, if possible. Repair work is done by shop floor operators.

Quality Control Station. This station performs a final Finished-Product quality control. This control is based on the measurement of mechanical and electrical parameters of the Finished-Product. After quality control, the Finished-Product gets a positive or negative quality state. Mobile robot drives it to unloading station.

Unloading station. Main function of this station is unloading a Finished-Product from an assembly pallet. When Finished-Product quality state is negative, it goes to scrap. If it is positive, then it is transferred to the packing station.

Packing Station. After unloading Finished-Product gets packed and prepared for a delivery.

4.3. BAS control structure and scheduling strategies

BAS has a hybrid control structure. This structure combines two basic control principles: hierarchy and heterarchy. Fig. 4.2. shows basic control principles: hierarchy, heterarchy and hybrid.

Hierarchical control is based on subordination, where only one leading/control element exist. This concept is very common for control of FAS.

Heterarchical control concept is present in the nature, but almost not used in the assembly domain. This concept is based on self-organization and therefore has no leading element and decentralized control.

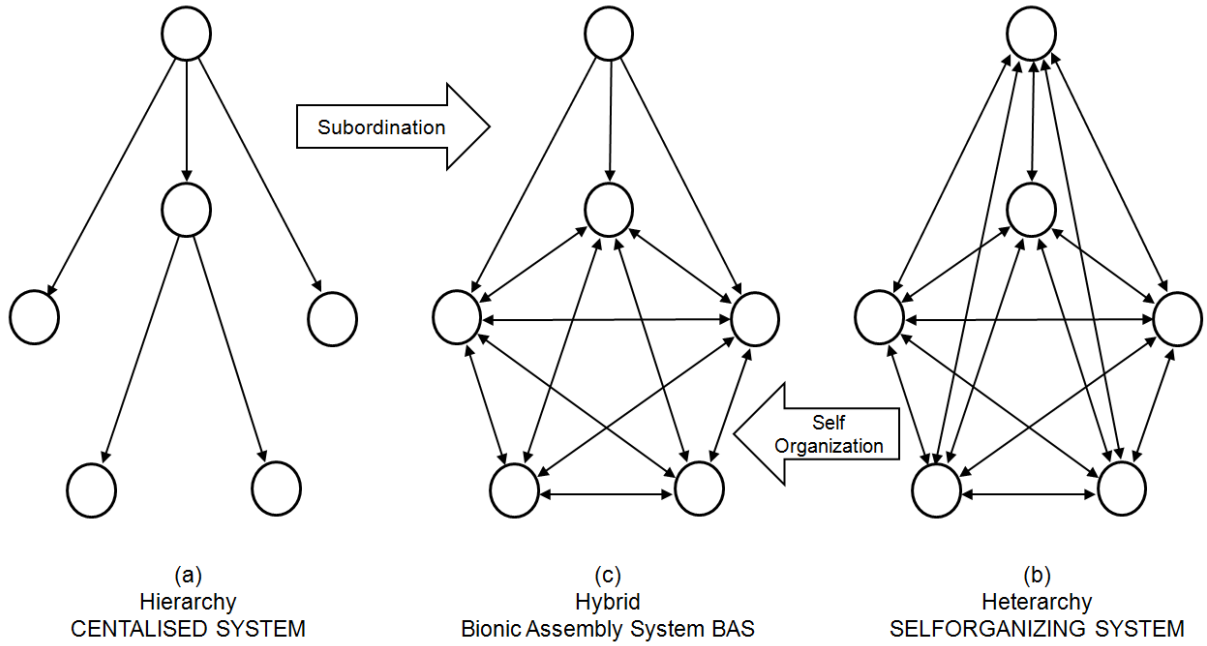


Fig. 4.2. Basic control principles: hierarchy, heterarchy and hybrid (Katalinic at al, 2012)

Combination of those two concepts brings out the hybrid system. Hybrid control has firm structure on the upper level of planning and flexibility on a lower level of realization. Therefore BAS control system is planned according to hybrid concept, as shown in Fig. 4.3.

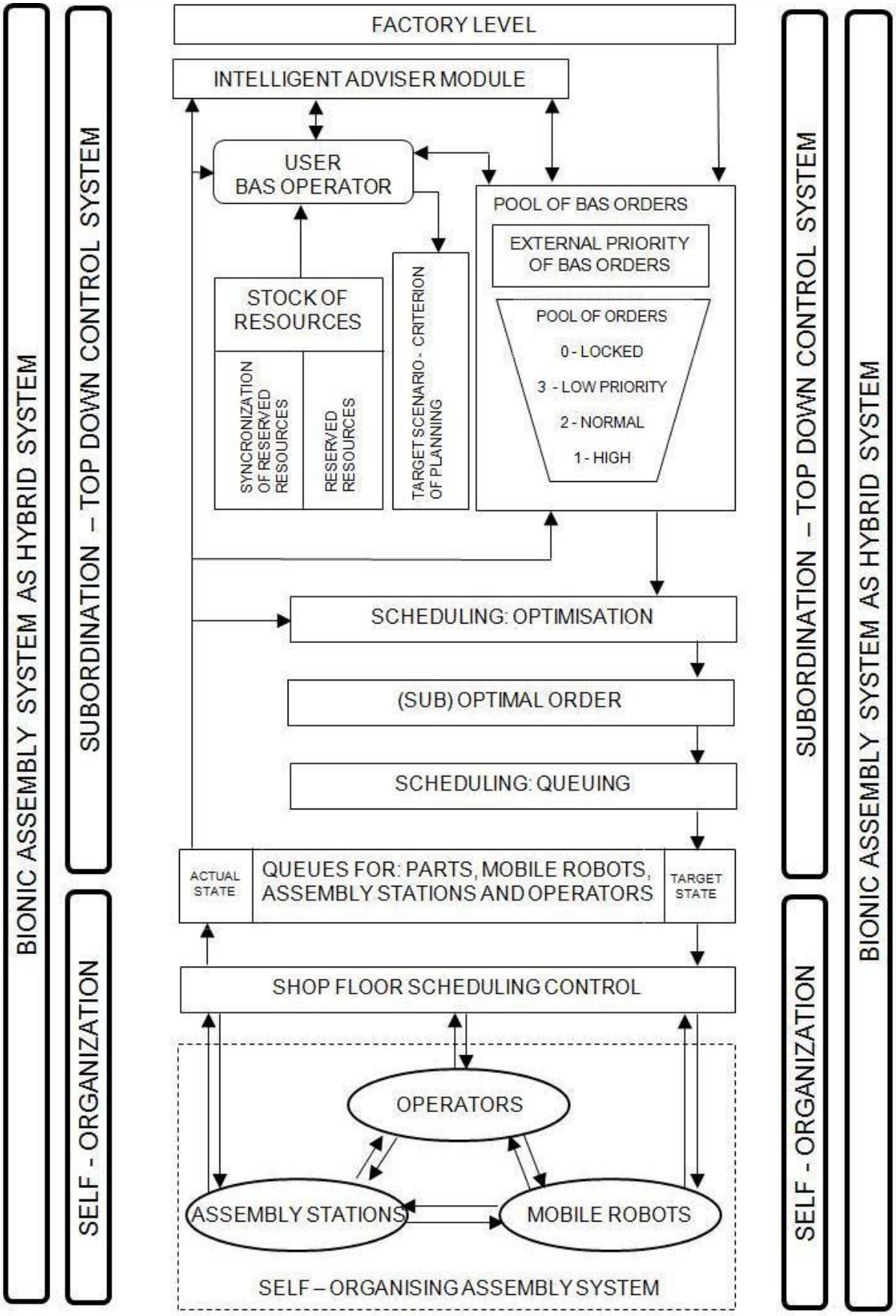


Fig. 4.3. BAS hybrid control structure (Katalinic at al, 2012)

BAS hybrid control structure combines two parts: top-down control system, based on subordination and heterarhical shop floor, based on self-organization. Upper system is similar with the discussed in previous chapter control system of FAS.

The planning activities happen in the following way. The first step in the production planning at the factory level is orders combination. The orders from different customers are combined in order to organize the scheduling plan, which satisfies customer wishes.

This plan is called the system order. It shows which products (product types and their runs) have to be assembled during the next period of time. According to this data it sets the amount and the priority of the orders. All unlocked orders in the pool of the orders are parts of the system order. Internal structure of the system order is shown in Fig. 4.4.

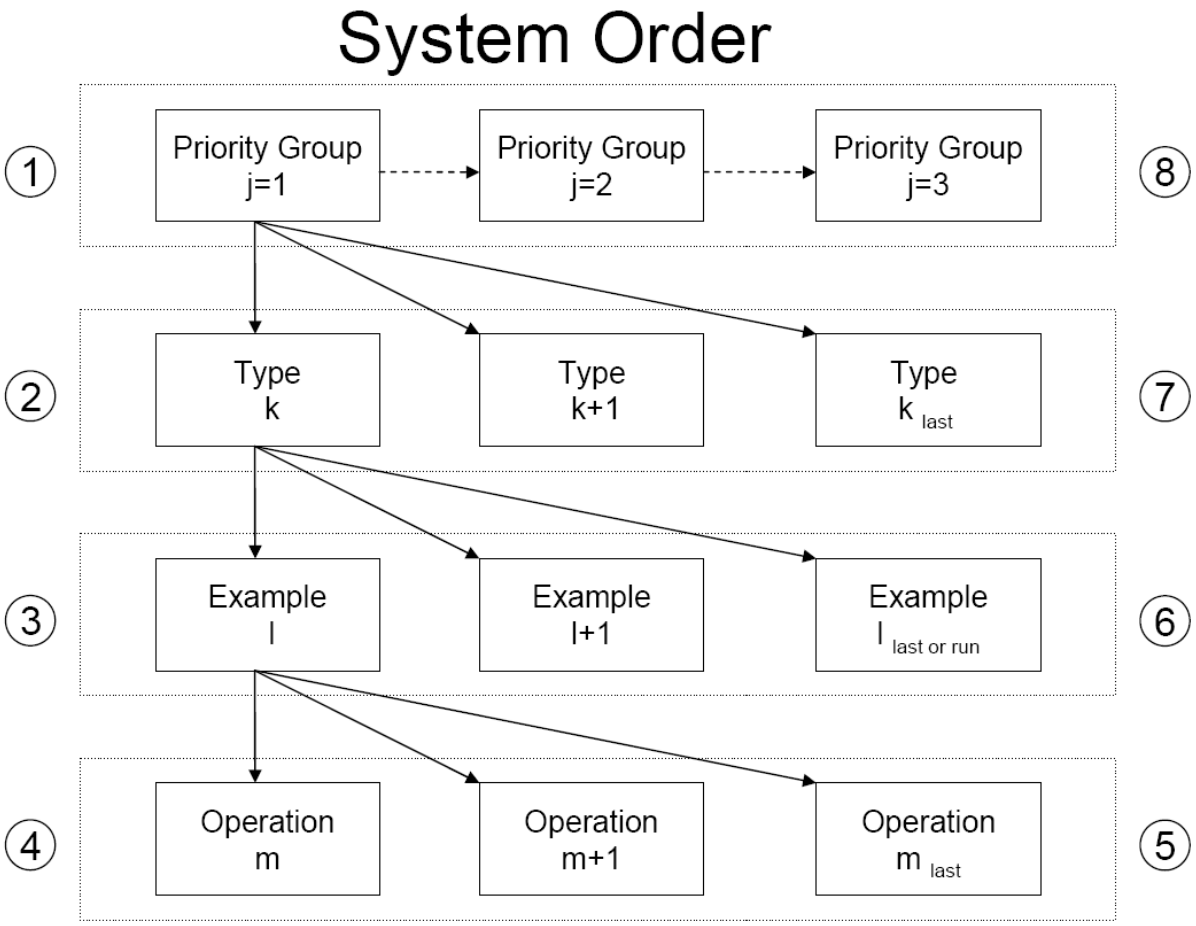


Fig. 4.4 Internal structure of system order

1. The group of assembly orders with the highest priority is selected from the system order.
2. From these groups, the first product type is selected
3. The first piece in the run of that product type will be assembled.

4. Mobile robot is getting order to assemble that piece. This assembly process is happening in the shop floor and follows basic principles of self-organization. Participants in self-organizing process are mobile robots, assembly stations and shop-floor operators. This part is shown at the bottom of the Fig. 4.3.
5. The procedures 3, 4 are repeating till the last piece of the run is assembled.
6. The procedure is repeating for the next product type in the priority group.
7. When the last product type in the priority group is assembled the whole procedure from step 2 till 6 is repeating for the next priority group.
8. System order is completed when the last piece in the run of the last product type in the lowest priority group is finished.

4.4 BAS Reconfiguration

The purpose of the reconfiguration is making assembly process more efficient, through the ability of assembly system to change current configuration. BAS Reconfiguration introduces the following abilities:

- Quicker adaptation to external disturbances
- Optimize assembly run time
- Optimize assembly step time
- Handle the internal disturbances
- Change number of elements in a system

There are two main examples of BAS reconfiguration. They are layout reconfiguration and queue rearrangement. The layout reconfiguration is performed by replacement of assembly stations and the queues rearrangement is organized by the movement of by mobile robots.

4.4.1 BAS layout reconfiguration.

All BAS elements are located on a shop floor in one order. This order is called BAS layout. There are movable and non-movable elements on a layout. Mobile robots are the elements that constantly move.

Some assembly stations in BAS can also be movable. Movable stations can change their position and orientation on BAS layout. In case that a station is too heavy or not easy to reconfigure it is called a non-movable BAS station. A change of system layout in this dissertation is called “layout reconfiguration”.

BAS concept allows the parallel assembly of multiple product families. Sometimes it happens that layout configuration is not optimal for these particular products, because of the long robot transport time. In this case the system layout must be reconfigured. Example of layout reconfiguration is shown in Fig. 4.5.

Movable assembly stations number 1, 2, 3 and 7, 8 are taking part in assembly of product family 1. Remaining movable stations 4, 5, 6, 9, 10 are taking part in assembly of product family 2. For this assembly run this layout is not optimal. Travel time of mobile robots can be minimized by layout reconfiguration. Stations 8, 7 are moving closer to the 1, 2, 3 and 9, 10 are moved closer to 4, 5 and 6. During this process stations are forming two groups within a new system layout.

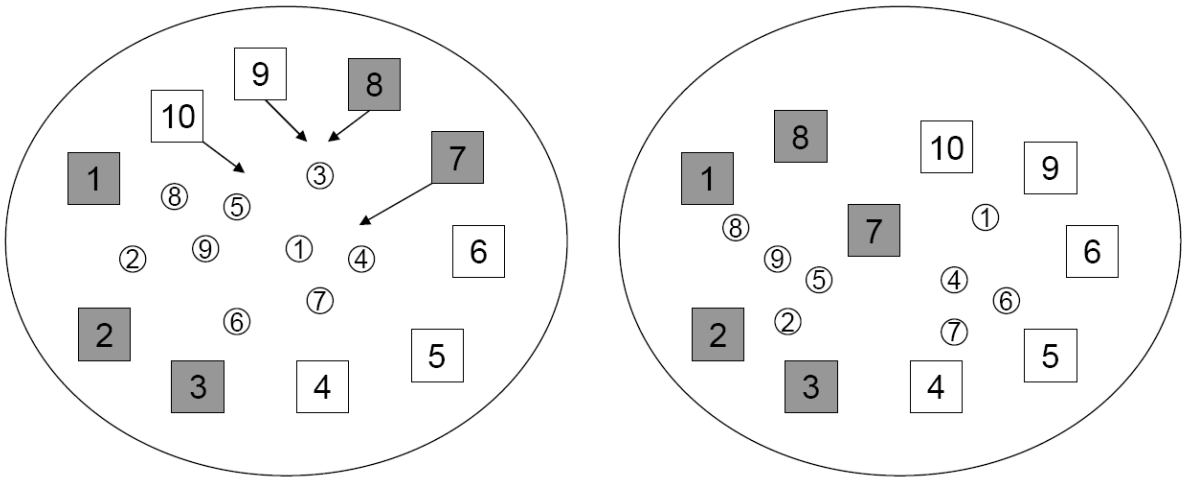


Fig. 4.5. Example of layout reconfiguration for two independent product families

Layout reconfiguration gives a big advantage to BAS when compared with classical assembly systems. It allows minimizing an assembly time of a product run. The other form of BAS reconfiguration is a queues rearrangement.

4.4.2 BAS working modes and queues rearrangement

BAS working mode represents execution of sequence of assembly orders. This sequence is formed by sub-ordinated subsystem. Each mobile robot gets a robot assembly order. It means to assemble one product. To complete this order it moves within the assembly stations.

It is typical for assembly stations to have a queue of waiting robots in front, as shown in Fig. 4.6. This mode is called normal working mode. The queue structure is shown in (4.1)

$$ {}^{O_i}_{P_m}S_1 : \blacksquare\, {}^{O_i}_{P_m}R_1 \ \blacksquare\, {}^{O_i}_{P_m}R_2 \ \dots \ \blacksquare\, {}^{O_i}_{P_m}R_{last} \qquad (4.1) $$

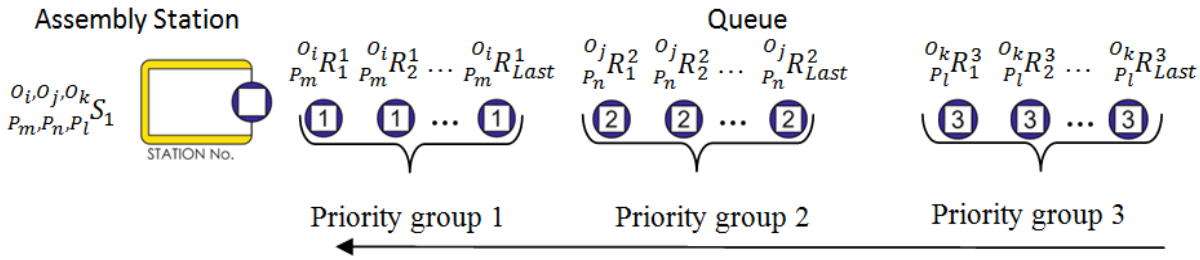


Fig. 4.6. BAS assembly station queue structure

In front of the station S number one for operation i-th on m-th product robots, numbered from one till the last, are waiting on i-th operation in the assembling of m-th product

There are 3 priority groups of orders (1 – high, 2 – normal, 3 – low). Typical situation in front of the one station is described in (4.2) and shown at Fig.3.7.

$$ {}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_1: \blacksquare\, {}^{O_i}_{P_m}R^1_1 \blacksquare\, {}^{O_i}_{P_m}R^1_2 \ldots \blacksquare\, {}^{O_i}_{P_m}R^1_{Last} \blacksquare\, {}^{O_j}_{P_n}R^2_1 \blacksquare\, {}^{O_j}_{P_n}R^2_2 \ldots \blacksquare\, {}^{O_j}_{P_n}R^2_{Last} \blacksquare\, {}^{O_k}_{P_l}R^3_1 \blacksquare\, {}^{O_k}_{P_l}R^3_2 \ \ldots \ \blacksquare\, {}^{O_k}_{P_l}R^3_{Last} \quad (4.2) $$

In station S it is possible to make i-th operation on m-th product, j-th operation n-th product and k-th operation on l-th product. The queues of the robots in front of the station, respect priorities form the following sequence.

In front of the station S number one for i-th operation on m-th product, j-th operation n-th product and k-th operation on l-th product, are waiting robots on i-th operation in the assembling of m-th product, with the first priority, numbered from one till the last. Then, following robots on j-th operation in the assembling of n-th product, with the second priority, numbered from one till the last. The last in the queue are robots on k-th operation in the assembling of n-th product, with the third priority numbered from one till the last.

Working mode after introduction of new alternative station. In the case of new station introduction it is necessary to rearrange the queue from the robots, waiting in front of the other station, as shown in (4.3).

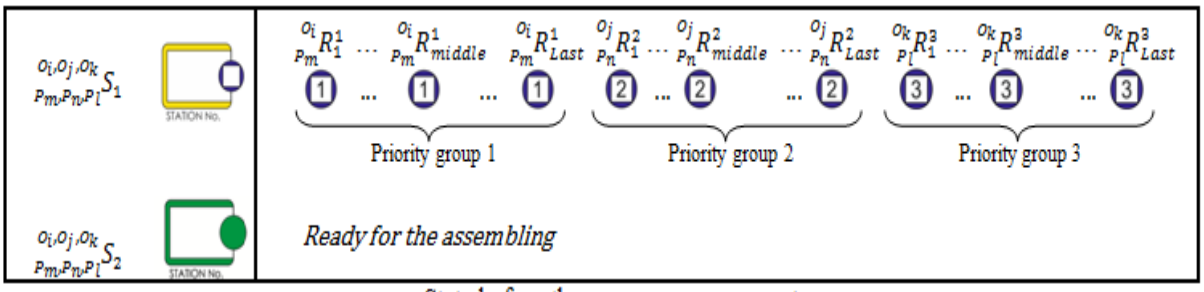
$$ \begin{aligned} &{}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_1: \blacksquare\, {}^{O_i}_{P_m}R^1_1 \blacksquare\, {}^{O_i}_{P_m}R^1_2 \ldots \blacksquare\, {}^{O_i}_{P_m}R^1_{middle} \ \ldots \ \blacksquare\, {}^{O_i}_{P_m}R^1_{last} \ \blacksquare\, {}^{O_j}_{P_n}R^2_1 \blacksquare\, {}^{O_j}_{P_n}R^2_2 \ldots \blacksquare\, {}^{O_j}_{P_n}R^2_{middle} \ldots \\ &\blacksquare\, {}^{O_j}_{P_n}R^2_{last} \ \blacksquare\, {}^{O_k}_{P_l}R^3_1 \blacksquare\, {}^{O_k}_{P_l}R^3_2 \ \ldots \ \blacksquare\, {}^{O_k}_{P_l}R^3_{middle} \ldots \blacksquare\, {}^{O_k}_{P_l}R^3_{last} \\ &{}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_2: Ready\ for\ the\ assembling \end{aligned} \quad (4.3) $$

In front of the station number one for i-th, j-th and k-th operation on m-th, n-th and l-th product are waiting robots on i-th operation in the assembling of m-th product, with the first priority, numbered from one till the middle. Then, following robots on j-th operation in the

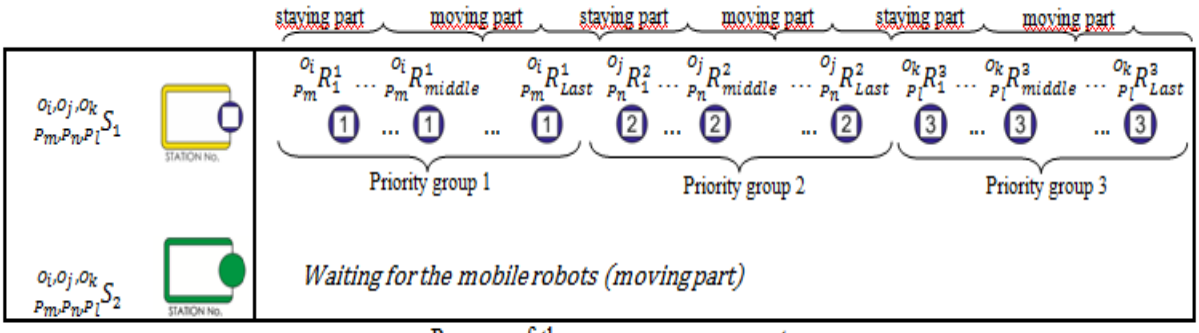
assembling of n-th product, with the second priority, numbered from one till the middle. The last in the queue are robots on k-th operation in the assembling of n-th product, with the the third priority numbered from one till the middle.

In front of the station number two for i-th, j-th and k-th operation on m-th, n-th and l-th product are waiting robots on i-th operation in the assembling of m-th product, with the first priority, numbered from middle+1 till the end. Then, following robots on j-th operation in the assembling of n-th product, with the second priority, numbered from middle+1 till the end. The last in the queue are robots on k-th operation in the assembling of n-th product, with the the third priority numbered from middle+1 till the end, as shown in (4.4) and at Fig. 4.7. The result of the queues rearrangement is shown in Fig. 4.8.

$$
\begin{aligned}
&{}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_1\colon \blacksquare\, {}^{O_i}_{P_m}R^1_1\ \blacksquare\, {}^{O_i}_{P_m}R^1_2 \dots \blacksquare\, {}^{O_i}_{P_m}R^1_{middle}\ \blacksquare\, {}^{O_j}_{P_n}R^2_1\ \blacksquare\, {}^{O_j}_{P_n}R^2_2 \dots \blacksquare\, {}^{O_j}_{P_n}R^2_{middle}\ \blacksquare\, {}^{O_k}_{P_l}R^3_1\ \blacksquare\, {}^{O_k}_{P_l}R^3_2 \dots \\
&\blacksquare\, {}^{O_k}_{P_l}R^3_{middle} \\
&{}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_2\colon \blacksquare\, {}^{O_i}_{P_m}R^1_{middle+1} \dots \blacksquare\, {}^{O_i}_{P_m}R^1_{last}\ \blacksquare\, {}^{O_j}_{P_n}R^2_{middle+1} \dots \blacksquare\, {}^{O_j}_{P_n}R^2_{last}\ \blacksquare\, {}^{O_k}_{P_l}R^3_{middle+1} \dots \\
&\dots \blacksquare\, {}^{O_k}_{P_l}R^3_{last}
\end{aligned}
\tag{4.4}
$$



State before the queue rearrangement



Process of the queue rearrangement

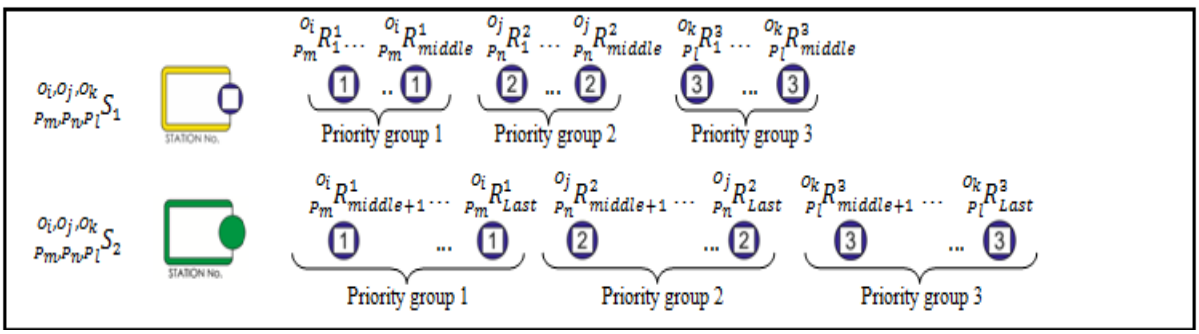


Fig. 4.7. Process of queues rearrangement

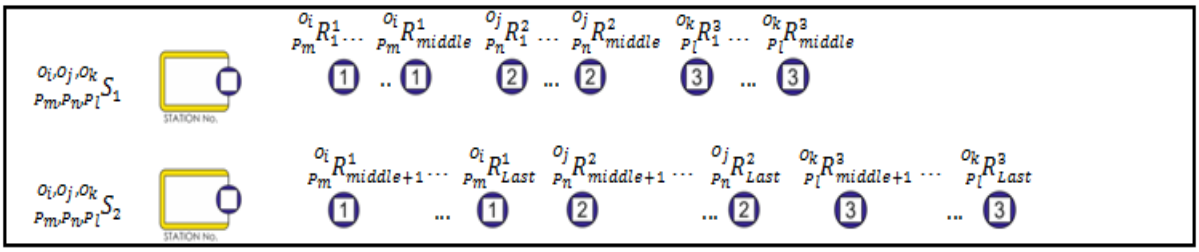


Fig. 4.8. Queues state after rearrangement process

Working mode after failure of one alternative station. In front of the stations number one and two for i-th, j-th and k-th operation on m-th, n-th and l-th product are waiting robots on i-th operation in the assembling of m-th product, with the first priority, numbered from one till the last. Then, following robots on j-th operation in the assembling of n-th product, with the second priority, numbered from one till the last.

The last in the queue are robots on k-th operation in the assembling of n-th product, with the third priority numbered from one till the last as shown in (4.5). In case of failure of the station number 2 mobile robots move to the station 1 in the following way:

$$\begin{matrix}{}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_1\colon \blacksquare\,{}^{O_i}_{P_m}R^1_1\ \blacksquare\,{}^{O_i}_{P_m}R^1_2\ \dots\ \blacksquare\,{}^{O_i}_{P_m}R^1_{last}\ \blacksquare\,{}^{O_j}_{P_n}R^2_1\ \blacksquare\,{}^{O_j}_{P_n}R^2_2\ \dots\ \blacksquare\,{}^{O_j}_{P_n}R^2_{last}\blacksquare\,{}^{O_k}_{P_l}R^3_1\ \blacksquare\,{}^{O_k}_{P_l}R^3_2\ \dots \\ \dots\ \blacksquare\,{}^{O_k}_{P_l}R^3_{last} \\ {}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_2\colon \blacksquare\,{}^{O_i}_{P_m}R^1_1\ \blacksquare\,{}^{O_i}_{P_m}R^1_2\ \dots\ \blacksquare\,{}^{O_i}_{P_m}R^1_{last}\blacksquare\,{}^{O_j}_{P_n}R^2_1\ \blacksquare\,{}^{O_j}_{P_n}R^2_2\ \dots\ \blacksquare\,{}^{O_j}_{P_n}R^2_{last}\blacksquare\,{}^{O_k}_{P_l}R^3_1\ \blacksquare\,{}^{O_k}_{P_l}R^3_2\ \dots \\ \dots\ \blacksquare\,{}^{O_k}_{P_l}R^3_{last}\end{matrix} \quad (4.5)$$

The rearrangement of queues in the case of failure of one alternative station is shown in (4.6) and at Fig. 4.9.

$$\begin{matrix}{}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_1\colon (\blacksquare\,{}^{O_i}_{P_m}R^1_1\ \blacksquare\,{}^{O_i}_{P_m}R^1_2\ \dots\ \blacksquare\,{}^{O_i}_{P_m}R^1_{last})\ (\blacksquare\,{}^{O_i}_{P_m}R^1_1\ \blacksquare\,{}^{O_i}_{P_m}R^1_2\ \dots\ \blacksquare\,{}^{O_i}_{P_m}R^1_{last})(\blacksquare\,{}^{O_j}_{P_n}R^2_1\ \blacksquare\,{}^{O_j}_{P_n}R^2_2\ \dots \\ \blacksquare\,{}^{O_j}_{P_n}R^2_{last})\ (\blacksquare\,{}^{O_j}_{P_n}R^2_1\ \blacksquare\,{}^{O_j}_{P_n}R^2_2\ \dots\ \blacksquare\,{}^{O_j}_{P_n}R^2_{last})\ (\blacksquare\,{}^{O_k}_{P_l}R^3_1\ \blacksquare\,{}^{O_k}_{P_l}R^3_2\ \dots\ \blacksquare\,{}^{O_k}_{P_l}R^3_{last})\ (\blacksquare\,{}^{O_k}_{P_l}R^3_1\ \blacksquare\,{}^{O_k}_{P_l}R^3_2\ \dots \\ \blacksquare\,{}^{O_k}_{P_l}R^3_{last}) \\ {}^{O_i,O_j,O_k}_{P_m,P_n,P_l}S_2\colon Out\ of\ function\end{matrix} \quad (4.6)$$

Robots on i-th operation in the assembling of m-th product, with the first priority, numbered from one till the last are coming to the end of the queue of i-th operation in the assembling of m-th product, with the first priority, on the station two. Then, following robots on j-th operation in the assembling of n-th product, with the second priority, numbered from one till the last are rearranged with the same rule. The last is the rearrangement in the queue of

robots on k-th operation in the assembling of n-th product, with the third priority numbered from one till the last.

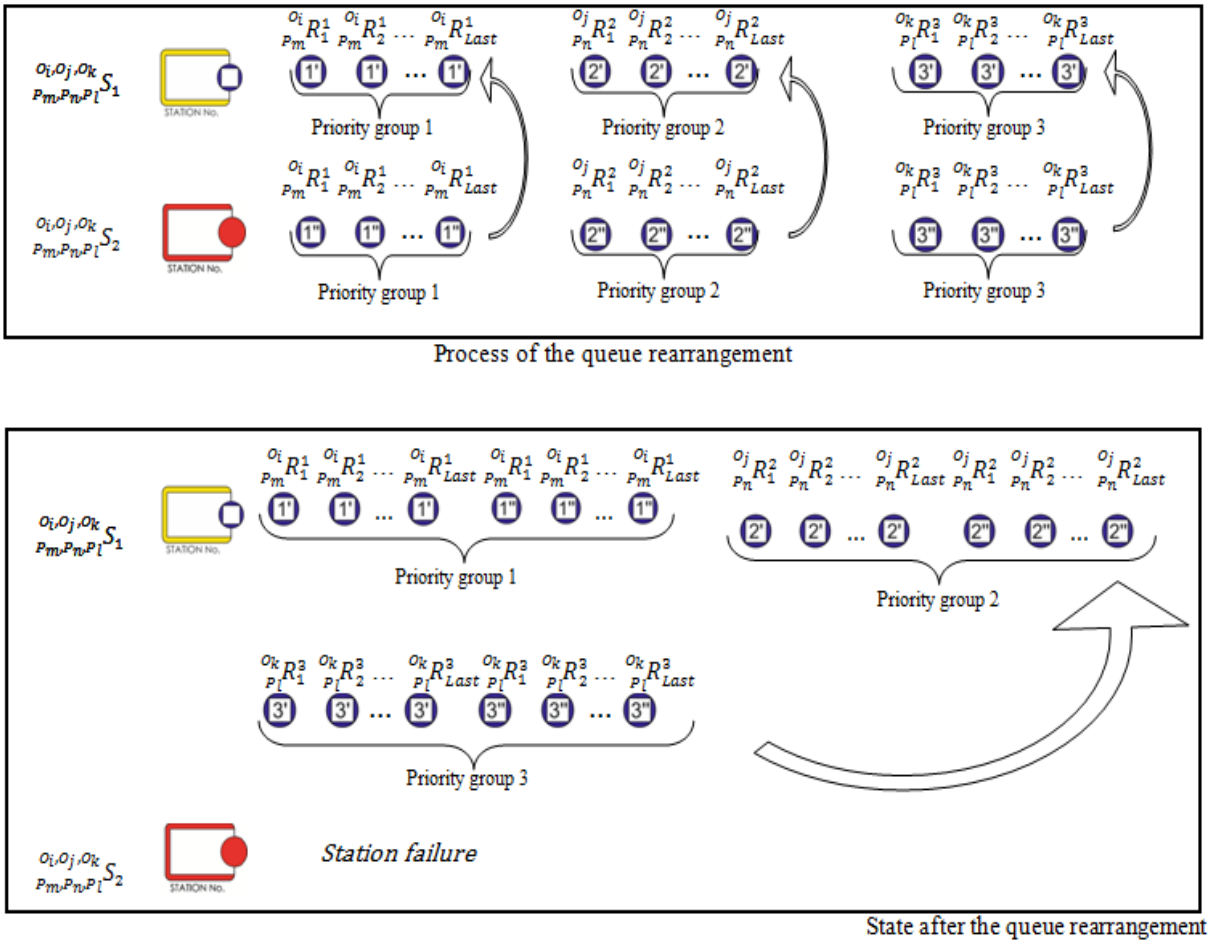


Fig. 4.9. Changes of the queues after failure of one alternative station

4.5. Summary

This chapter described general concept of BAS. It represents the assembly system from the next generation of flexible assembly systems. It is robust, worker-friendly assembly system, which allows to assembly wide range of products with smaller volumes.

BAS hybrid control structure combines top-down control and self-organizing subsystems. This combination provides a structure to scheduling and flexibility in the assembly process. BAS layout organization allows many possibilities for space optimizations, component replacements, system scalability and reconfiguration. BAS reconfiguration is organized at two levels: queue and layout.

Concept of BAS is promising for an implementation in industry for assembling products with a complexity level of electrical motor. A case study of BAS concept implementation for an electrical motor facility reorganization is shown in chapter 5.

Chapter 5

BAS Adjustments

The complex analysis of FAS and BAS systems shows that there is a possibility to implement BAS concept to cover the assembly range of studied FAS. However, both FAS assembly sub-systems as well as BAS concept have to be adjusted according to following conditions, restrictions and limitations.

1. **No changes of motors and parts design are allowed.** A facility has to produce the motors according to a customer specification without any change of motor and parts design.
2. **Keep Just-in-time concept of output.** The exact number of motors must be prepared at the fix date and time for delivery to one particular customer according to the specification.
3. **Incoming parts quality has to remain the same.** That means in average 1 part from 1000 will be defective. This condition has to be kept in order to keep the motor prices low.
4. **New system has to replace the old one in step by step mode.** That means that new system deployment must be organized in a smooth way and special stations, assembly stations, and subsystems from existing system has to be integrated in new one.

This chapter focuses on the BAS and FAS adjustments. It includes FAS layout reorganization, BAS control structure adjustments and the description of working scenario. Points, which don't satisfy the above, mentioned conditions are redesigned. The chapter presents the extended discussion of results, earlier published in (Katalinic et al., 2014a) and (Katalinic et al., 2014b).

5.1 Layout reorganization

Main FAS adjustment is the task to form BAS layout. All FAS stations from both subsystems must be brought into one layout at one shop floor. According to the restriction, new system

has to replace the old ones step by step. Therefore, extra stations and old system elements would be stored in a supplementary subsystem.

There is an intensive material exchange between these subsystems. Therefore, BAS transport system must be included for this flow realization. The flow includes parts, tools, setup components etc. Transport tasks can be made manually at the start point and then smoothly changed to an automatic delivery.

The main difference in core subsystem will be a change of current transport systems to a swarm of mobile robots. These robots will transport assembly pallets between assembly stations as a replacement to conveyor belts. Switched-off robots will be located at robot storage in supplementary subsystem. Standby and active robots must be in a core subsystem. Storage for these robots, the pool of robots, must be constructed additionally. The main pool of robots task is to keep, charge and maintain stand-by and active robots.

When robot gets an assembly order it activates and drives out of the pool to take pallet of needed type. Therefore pallet storage must be placed nearby in a core subsystem. This storage is called pool of pallets. In order to be compatible with robot docking slots, must be organized at each station.

Robot can complete assembly order using different assembly trajectories. It is possible, that one assembly operation can be completed by alternative assembly stations. In order to keep all initial assembly stations some of them will be placed as alternative stations, another ones will be reformed.

Some preassembly operations could be performed within one group of stations. After that, preassembled parts will be mounted into the Product-In-Assembly on assembly pallet. This will reduce robot docking and travelling time. At the next steps, these groups will have one docking slot and will act as one assembly station. Station groups are shown in Table 5.1

Table 5.1. Organization of assembly station groups

Name of the station group	**Motor families**	**Names of the Station**	**Function Description**
Stator	1 / 2	• Stator Pre-assembly Station • Stator Measurement and Assembly Station	Stator is preassembled and measured inside the group. After this procedure stator is assembled to a product-in-assembly.
Rotor	1 / 2	• Rotor Pre-Assembly Station • Rotor Assembly Station • Ball bearings pressing	Rotors is preassembled and measured inside the group. Rotor is assembled with spring, disc brake, safety rings (180 degrees to each other, mounted together) and ball bearing. After this procedure rotor is

		• Ball bearing safety ring	assembled to a product-in-assembly.
2nd End-shield / End-shield	1 / 2	• 2nd End shield Pre Assembly Station • Heating Station • 2nd End shield Assembly Station • End shield Assembly Station(One-sided motor family) • Ball bearings pressing • Ball bearing safety ring	2nd End shield is pre-heated with hot air 40-80C, ball-bearings are pressed into an End shield with a force (5-50 kg) depending on a motor type. After this procedure 2nd End shield is assembled to a product-in-assembly. This station will be combined with an End-shield station for assembly of motor family of type 2. In case of overload this station will be doubled with a similar one from family 2.
Fixing plate	2	• Fixing Plate Preparation • Fixing Plate Assembly	Fixing Plate is prepared and pre-assembled. After this procedure Fixing Plate is assembled to a product-in-assembly
Quality control-1	1/2	• Quality Control -1 (One-sided motor family) • Quality Control -1 (Double-sided motor family)	Examination of axial clearance and Connection dimensions. These equal stations will be combined for both motor families.
Quality control-2	1/2	• Quality Control -2 (One-sided motor family) • Quality Control -2 (Double- sided motor family)	Examination of electrical parameters of motor. These equal stations will be combined for both motor families.

Minimal number of stations with docking slots for the stations, which cover assembly operations of two product families is 23. This includes two storages - pool of pallets and robots, repair station, packing stations and three stations for quality control.

A minimal number of stations, which perform assembly operations, is 16. All stations used in assembly process in a new layout are shown in Table 5.2.

Table 5.2. Stations of combined assembly system

N	Name of the station	Double-sided motor family	One-sided motor family
1	1st End shield	X	

2	2nd End-shield / End-shield	X	X
3	Balancing Disc Station - 2	X	
4	Balancing Disc Station -1	X	
5	Condenser Assembly - 1	X	
6	Condenser Assembly - 2	X	
7	Fan Assembly	X	
8	Fixing plate		X
9	Fixture of Motor and Electrical Components - 2	X	X
10	Fixture of Motor and Electrical Components -1	X	X
11	Heating Station	X	X
12	Rotating Station	X	
13	Rotor	X	X
14	Rubber buffers control		X
15	Rubber buffers screw		X
16	Stator	X	X
17	Unloading/Packing	X	X
18	Quality control-1	X	X
19	Quality control-2	X	X
20	Quality control-3	X	
21	Repair	X	X
22	Pool of Pallets	X	X
23	Pool of Robots	X	X

5.2 Control structure adjustment

Top-down control structures of initial FAS is based on hierarchy. Therefore this structure doesn't support self-organizing shop floor. A new control structure must combine top-down control system with vertical communication for upper part and self-organizing heterarchical control system with shop floor horizontal communication.

BAS control structure is hybrid and allows combining these two structures. Usage of BAS control structure will allow keeping the same customer interface and JIT delivery. From another side it will allow to control self-organizing subsystems. However deeper analysis of

BAS control structure and simulation of BAS working scenarios (Kukushkin, 2012) shown that for the further BAS Hybrid control structure implementation it is necessary to deal with following weak points:

- Strong interaction between subordinating and self-organizing subsystems. There is no smooth and clear interface
- Shop floor facilities cannot function without permanent and active role of the control system
- Absence of independent Log data. Data accessibility is limited on working time of the control system
- Communication between active components follows the concept “Everyone-to-Everyone”
- Communication has no standardized protocol
- If only one element is active, it communicates with nobody. If there are only independent irrelevant active elements, system cannot function. They can communicate, but cannot make progress in production.
- Start of production strongly depends on the shutdown sequence (switching off order of the stations). Special problem is when a new start is made by stations, which were not used during the last work period.

BAS control structure adjustment is organized through the implementation of cloud communication concept to the hybrid control structure of BAS. Cloud communication / computing is one of emerging IT phenomena. Different variation of cloud computing are applied in different fields of modern science. Applications of cloud computing in IT environment are shown in (Davenport, 2013), dataflow computing in automation applications (Panfilov & Salibekyan, 2014), improvement of ERP systems (Gastermann, 2014).

Google Docs, Dropbox, Skydrive are examples of cloud computing in home and office usage (Blau & Avner, 2009). Usage of cloud helps organize better communication and increase the collaboration within the company (Sun et al., 2014).

The ideas of cloud communication and its infrastructure are used to create a new cloud element inside the control structure of BAS. The original concept is published at (Katalinic et al., 2014a) and BAS cloud functions and working scenario in (Katalinic et al., 2014b)

BAS Cloud is an additional element, which is introduced into BAS hybrid control structure. It acts as an informational interface between subordinating and self-organizing subsystems. Therefore, it connects shop floor scheduling control unit of subordinating subsystem with all active elements of self-organizing subsystems as shown in Fig. 5.1. Main functions of BAS Cloud are:

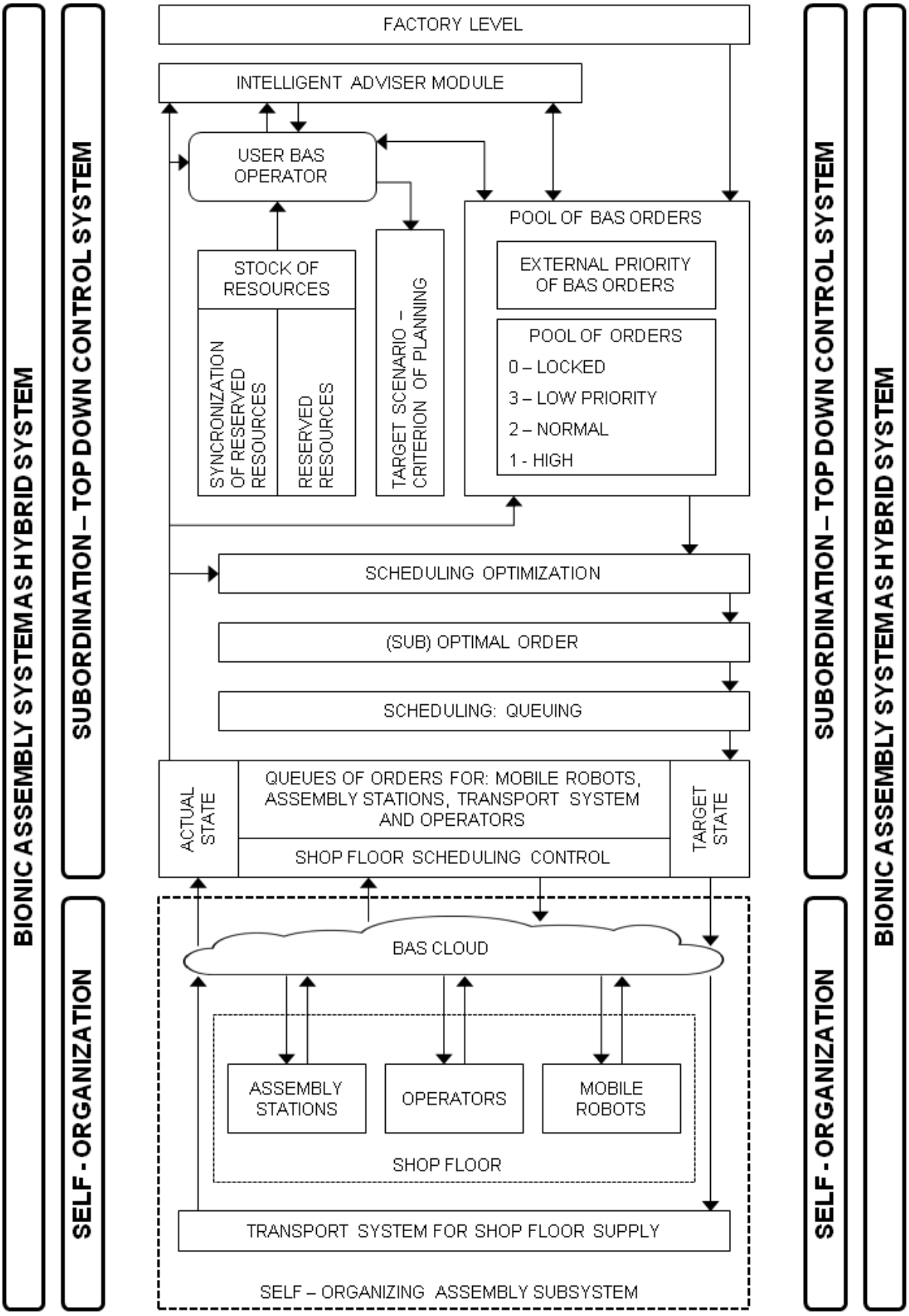

BIONIC ASSEMBLY SYSTEM AS HYBRID SYSTEM
SUBORDINATION – TOP DOWN CONTROL SYSTEM
SELF - ORGANIZATION
FACTORY LEVEL
INTELLIGENT ADVISER MODULE
USER BAS OPERATOR
POOL OF BAS ORDERS
EXTERNAL PRIORITY OF BAS ORDERS
POOL OF ORDERS
0 – LOCKED
3 – LOW PRIORITY
2 – NORMAL
1 - HIGH
STOCK OF RESOURCES
SYNCRONIZATION OF RESERVED RESOURCES
RESERVED RESOURCES
TARGET SCENARIO – CRITERION OF PLANNING
SCHEDULING OPTIMIZATION
(SUB) OPTIMAL ORDER
SCHEDULING: QUEUING
ACTUAL STATE
QUEUES OF ORDERS FOR: MOBILE ROBOTS, ASSEMBLY STATIONS, TRANSPORT SYSTEM AND OPERATORS
SHOP FLOOR SCHEDULING CONTROL
TARGET STATE
BAS CLOUD
ASSEMBLY STATIONS
OPERATORS
MOBILE ROBOTS
SHOP FLOOR
TRANSPORT SYSTEM FOR SHOP FLOOR SUPPLY
SELF – ORGANIZING ASSEMBLY SUBSYSTEM
SUBORDINATION – TOP DOWN CONTROL SYSTEM
SELF - ORGANIZATION
BIONIC ASSEMBLY SYSTEM AS HYBRID SYSTEM

Fig. 5.1. Cloud-based BAS hybrid control structure

1) Connection of self-organizing and subordinating parts. BAS Cloud allows combining vertical and horizontal communication. Exchange of information between a cloud and subsystems goes through communication channels. In this dissertation flow of information from shop floor scheduling control unit to the cloud is defined as vertical upload and in the opposite way vertical download. Flow of information from self-organizing sub-system components to the cloud are defined as horizontal upload and in the opposite way horizontal download as shown in Fig. 5.2.

2) Internal horizontal communication organization. Cloud has a two-way communication with the shop floor elements: assembly stations, operators and mobile robots. Introduction of a cloud changes the “everyone to everyone” communication to a direct “element - cloud” interactions, as shown in Fig. 5.2.

3) Storage of system technical data. Cloud stores products, stations, mobile robots and transport system technical data. Product technical data includes a number and types of operations for product assembly and suitable pallet types. Stations data includes abilities for assembly operations, station size, ability to move etc.

4) *Memory storage.* Cloud is a passive storage element. It only stores the data, which comes from other active elements and doesn't react on system changes. It contains technological, control and system state data.

5.2.1 BAS cloud internal structure

BAS Cloud stores the information in number of different files: files for vertical and horizontal communications organization, log files and technical data files. All this files are used by different subsystems and elements. Main cloud elements are shown in Fig. 5.2. They are divided into two groups.

- **Queues**. Upper group consists of queues of orders. These queues are formed according to logic of working cycles discussed in Chapter 3. As a result of this process queues of orders for mobile robots, transport system and operators come to the target state module of the shop floor scheduling control unit. This module vertically uploads these queues to the cloud. The queue of station assembly orders is formed dynamically from mobile robots, and uploaded horizontally. The states of the queues are tracked and vertically downloaded by actual state module of shop floor scheduling control unit.
- **Orders**. Each queue consists of number of single orders. That means that each single order has its own cloud file. This orders are horizontally downloaded by self-organizing subsystem elements. All the state changes in orders are horizontally uploaded to the cloud by self-organizing subsystem elements.

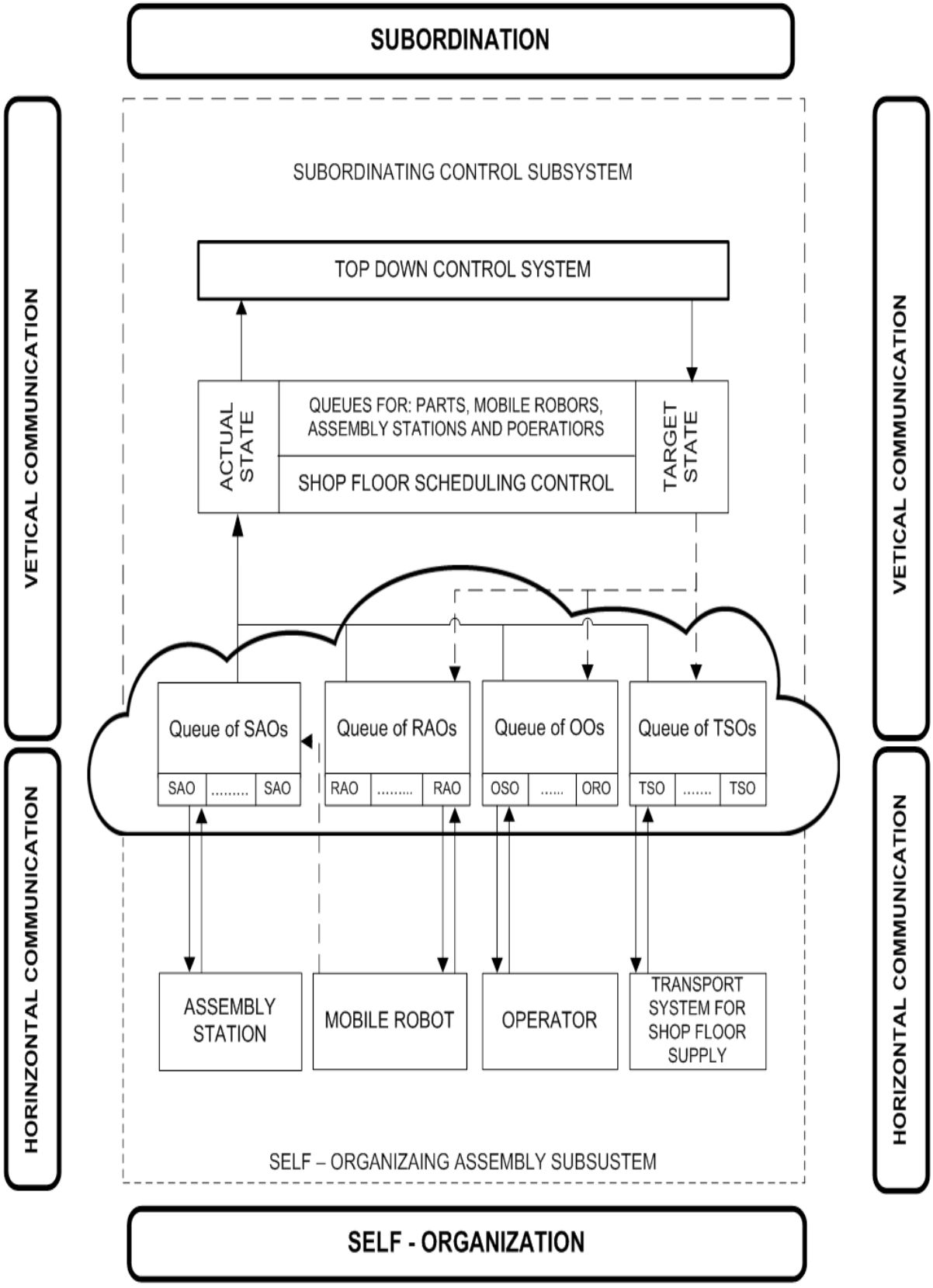


Fig. 5.2. Cloud-based communication in BAS

Queue of Robot Assembly Orders is shown in Table. 5.3. Queue includes an order creation time in YYMMDDHHMMSS format, assembly order ID, product ID, order priority and status. When order is created it has a status 'new'. When mobile robot finds a new order it reserves it by uploading its ID into Robot ID field of corresponding RAO. After that it reads Pallet ID and goes to pool of pallets to get a pallet of needed type. When order is completed its status is changed to *Completed*. Changes in a queue of RAOs are vertically downloaded to an actual state module of shop floor scheduling control unit.

Table. 5.3 Queue of Robot Assembly Orders

Queue of Robot Assembly Orders						
Order creation time	**Assembly Order ID**	**Product ID**	**Priority**	**Robot ID**	**Pallet ID**	**Order Status**
YYMMDDHHMMSS	RAO ID-Timestamp	M1	1	R35-01	P15-003	Completed
YYMMDDHHMMSS	....	....	....	....	....	....
YYMMDDHHMMSS		Mn	3	R35-06	P15-008	In Process
YYMMDDHHMMSS	....	....	....	....	....	....
...		Mlast	1	0	P8/P12	New

Robot Assembly Order Data File stores complete information about RAO, as shown in the Table 5.4. RAO ID and Product type are assigned by Shop Floor scheduling control unit according to the logic of working cycles. Pallet type and Number of Assembly Steps (NAS) are filled according to technological data. Robot and pallet IDs, Pallet Quality State and Product Quality State are horizontally uploaded by mobile robot to RAO Data File. When Robot Assembly Order Data is completed robot is ready for assembly.

To assemble one product robot has to make a number of assembly operations. Some of these operations depend on an assembly state of the product; some could be done only in a strict order. This number of assembly operations is called assembly sequence. To complete an assembly sequence, robot has to make a fixed number of assembly steps (NAS), one operation per step. When robot chooses assembly station for the next assembly step, it horizontally uploads station and operation ID to the corresponding assembly step field in RAO Data File.

Table 5.4. Cloud-stored Robot Assembly Order Data File

Robot Assembly Order Data				
Timestamp	**Data**	**Value**		
YYMMDDHHMMSS	RAO ID	lkhji-YYMMDDHHMMSS		
YYMMDDHHMMSS	Product Type	Number according to technological data		
YYMMDDHHMMSS	Pallet Type	Number according to technological data		
YYMMDDHHMMSS	Number of assembly steps(NAS)	According to technological data		
YYMMDDHHMMSS	Robot ID	Number according to technological data		
YYMMDDHHMMSS	Pallet ID	Number according to technological data (Type-ID)		
YYMMDDHHMMSS	Pallet Quality State	Positive / Negative		
YYMMDDHHMMSS	Product Quality State	Positive / Negative		
Assembly sequence Data				
Timestamp	**Step (1...NAS)**	**Station ID**	**Operation**	**Status**
YYMMDDHHMMSS	1	AS ID	OP ID	Waiting / In Transport /
YYMMDDHHMMSS	2	AS ID	OP ID	Completed / In Process /
YYMMDDHHMMSS	...	...	...	Failed / Repair
YYMMDDHHMMSS	NAS	AS ID	OP ID	

During an assembly sequence the RAO could have six states:

1. *In Transport* - when assembly pallet is transported by robot to a chosen assembly station.
2. *Waiting* - robot waits an assembly operation in a queue in front of chosen assembly station.
3. *In Process* - Assembly pallet is on a station for an assembly operation.
4. *Completed* - Assembly operation succesfully completed, quality states of product and pallet are positive.

5. *Failed* - Assembly operation is not succesfully completed or quality states of product or pallet is negative. Robot has to drive this pallet to a repair station.
6. *Repair* - Pallet is on a repair station, waiting for an operator.

When assembly sequence is successfully finished, packing station horizontally uploads order status Completed to the Queue of Robot Assembly Orders file in a cloud.

Station Assembly Orders Data File stores complete information about SAOs, as shown in the Table 5.5. It consists of Current SAO Data, Queue of SAOs, and SAO Log sections. Current SAO Data File includes Pallet ID, operation ID, Pallet Quality State, Product Quality State. Pallet and operation ID are horizontally uploaded by mobile robot to the queue of SAOs. SAO could have following statuses:

- Waiting - SAO is in a queue of SAOs, waiting for assembly operation. Queue of SAOs is formed from the RAOs of robots waiting in front of the station for assembly operation. RAOs are formed into three priority groups: High, Normal and Low. If there is new SAO, it will be placed to the corresponding priority group.
- Cancelled - SAO is cancelled by Robot. This SAO is moved to SAO Log section
- In Process - SAO is leaving a queue for assembly operation. This SAO is moved to Current SAO section. Assembly Station processing the SAOs according to the priority group and the First-In-First-Out principle within each group.
- Completed - Assembly operation is completed successfully. This SAO is moved to SAO Log section. After assembly operation station checks pallet and product quality state and horizontally uploads it to RAO Data file. If these states are positive, then Assembly Station horizontally uploads the *Completed* state to RAO Data file.
- Failed. Assembly operation is not completed successfully. This SAO is moved to SAO Log section. Assembly Station horizontally uploads this state to RAO Data file.

Table 5.5. Station Assembly Orders Data File

Current Station Assembly Order Data		
Timestamp	**Data**	**Value**
YYMMDDHHMMSS	Pallet ID	Number according to technological data (Type-ID)
YYMMDDHHMMSS	OP ID	Completed / In Process / Failed
YYMMDDHHMMSS	Pallet Quality State	Positive / Negative

YYMMDDHH MMSS	Product Quality State			Positive / Negative	
Queue of Station Assembly Orders					
Order creation time	**SO ID**	**Operation**	**Operation Time**	**Priority group**	**Order Status**
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	...	...
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	High	Waiting
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	Normal	Waiting
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	...	...
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	Normal	Waiting
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	Low	Waiting
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	...	...
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	Low	Waiting
Station Assembly Orders Log					
Order creation time	**SO ID**	**Operation**	**Operation Time**	**Priority group**	**Order Status**
YYMMDDHH MMSS	YYMMDDHH MMSS	YYMMDDHH MMSS	YYMMDDHH MMSS	YYMMDDHH MMSS	YYMMDDHH MMSS
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	High	Completed
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	Normal	Failed
YYMMDDHH MMSS	Type-Timestamp	OP ID	Time (sec)	High	Cancelled

Changes in SAO Data File are vertically downloaded by actual state module of shop floor scheduling control unit.

Queue of Transport System Orders (TSO) is shown in the Table. 5.6. The transport system is responsible for a material flow organization. Therefore TSOs could have two types:

- Input TSO to bring material from supplementary to a core subsystem
- Output TSO to bring material from core to supplementary subsystem

Queue of TSOs includes an order creation time in YYMMDDHHMMSS format, Transport System Order ID, Element ID, order priority and status. Element ID means ID of system element (station, robot), which is an origin in case of output TSO or destination in case of input TSO. There are four TSO Statuses:

- **Waiting.** When TSO is created it has a status waiting.
- **In Process.** When transport system operator finds a Waiting TSO it reserves it by uploading his ID into Operator ID field of corresponding TSO and starts to fulfil this order.
- **Completed**. When TSO is completed its status is changed to Completed.
- **Failed.** When TSO is failed operator changes its status to failed.

Table. 5.6. Queue of Transport System Orders

Queue of Transport System Orders						
Order creation time	**TSO ID**	**Element ID**	**Order type**	**Priority**	**Order Status**	**Operator ID**
YYMMDDHHMMSS	TSO ID-Timestamp	Element ID	Input / Output	1	Completed	Operator ID
YYMMDDHHMMSS	TSO ID-Timestamp	Element ID	Input / Output	....	Completed	Operator ID
YYMMDDHHMMSS	TSO ID-Timestamp	Element ID	Input / Output	3	In Process	Operator ID
YYMMDDHHMMSS	TSO ID-Timestamp	Element ID	Input / Output	....	Waiting	
YYMMDDHHMMSS	TSO ID-Timestamp	Element ID	Input / Output	1	Waiting	

The transport System Order Data File stores complete information about TSOs, as shown in the Table 5.7. It consists of TSO ID, TS operator ID, Element ID, transport properties and transport state. Transport property is shows what has to be transported (Parts / Tools / Elements), type of this property and a number of pieces of this property. Changes in a queue

of TSOs are vertically downloaded by actual state module of shop floor scheduling control unit.

Table 5.7. Transport System Order Data File

Transport System Order		
Timestamp	**Data**	**Value**
YYMMDDHHMMSS	TSO ID	TSO-Timestamp
YYMMDDHHMMSS	Operator ID	Operator ID
YYMMDDHHMMSS	Element ID	Element ID
YYMMDDHHMMSS	Transport property	Parts / Tools / Elements
YYMMDDHHMMSS	Transport property type	Type of Parts / Tools / Elements
YYMMDDHHMMSS	Transport property number	Number of Parts / Tools / Elements
YYMMDDHHMMSS	TSO State	Waiting / In Process /Completed / Failed

Queue of Operator Orders is shown in the Table. 5.8. Operators are responsible for assembly, transport, setup and repair orders. At first case they act as manual assembly station and get their SAOs as other stations. At the second case they act as transport systems operators and get the TSOs.

Queue of Operator Orders (OOs) includes Repair and Setup OOs. Other field of queue are: OOs creation time in YYMMDDHHMMSS format, OO ID, Station ID, order priority and status. Element ID means ID of system element, which needs setup. When order is created it has a status new. When operator finds a new order it reserves it by uploading his ID into Operator ID field of corresponding TSO. When order is completed its status is changed to Completed. Changes in a queue of OOs are vertically downloaded by actual state module of shop floor scheduling control unit. There are four states of OO.

- **Waiting.** When OO is created it has a status waiting.
- **In Process.** When transport system operator finds a Waiting OO it reserves it by uploading his ID into Operator ID field of corresponding OO and starts to fulfil this order.
- **Completed**. When OO is completed its status is changed to Completed.
- **Failed.** When OO is failed operator changes its status to failed.

Table. 5.8. Queue of Operator Orders

Queue of Operator Orders						
Order creation time	**OO ID**	**Station ID**	**Priority**	**Order type**	Order Status	**Operator ID**
YYMMDDHHMMSS	OO ID - Timestamp	Station ID	1	Repair	Completed	Operator ID
YYMMDDHHMMSS	OO ID - Timestamp	Station ID	2	Setup	In Process	Operator ID
...	...	...	...	...	...	...
YYMMDDHHMMSS	OO ID - Timestamp	Station ID	3	Repair	Waiting	

Operator Repair Order (ORO) Data File stores complete information about ORO, as shown in the Table 5.9.

Table 5.9. Operator Repair Order Data File

Operator Repair Order		
Timestamp	**Data**	**Value**
YYMMDDHHMMSS	ORO ID	ROO-Timestamp
YYMMDDHHMMSS	Operator ID	Operator ID
YYMMDDHHMMSS	AO ID	lkhji-YYMMDDHHMMSS
YYMMDDHHMMSS	Product Type	Number according to technological data
YYMMDDHHMMSS	Number of assembly operation	According to technological data
YYMMDDHHMMSS	Pallet ID	Number according to technological data
YYMMDDHHMMSS	Current operation ID	OP ID
YYMMDDHHMMSS	Pallet Quality State	Positive / Negative
YYMMDDHHMMSS	Product Quality State	Positive / Negative
YYMMDDHHMMSS	ORO State	Waiting /In Process /Completed / Failed

The first section consists of general ORO Data such as ORO ID, Operator ID, AO ID. The second part includes technical data of the product: Product Type, Number of assembly

operation, Pallet ID and current operation ID. The third section helps operator to identify a problem. It includes Pallet and Product Quality States. There are three possible results of repair. At first two cases ORO state will be changed to Completed, and at the last one to Failed.

1. **Complete current assembly step.** Operator repairs the mistake and completes the assembly step. After repair mobile robot goes for a next assembly step.
2. **Cancel current assembly step.** Operator cannot complete the assembly step. Operator repairs the mistake by removing the defective part added in current assembly step. After repair mobile robot repeats a current assembly step.
3. **Remove Irreparable Product-In-Assembly.** If operator is unable to complete or cancel the current assembly step, the Product-In-Assembly is moved to scrap. This happens in case of heavy damage of Product-In-Assembly.

Operator Setup Order (OSO) Data File stores complete information about OSO, as shown in the Table 5.10. The first section consists of general OSO Data such as OSO ID, Operator ID and Station ID. The second part includes setup properties. Setup property is shows the mean, which has to be set up to the station (Parts / Tools / Maintenance), type of this mean and a number of pieces of this mean. After the setup procedure OSO status is changed to Completed in case of successful setup or Failed in case of failure.

Table 5.10. Operator Setup Order Data File

Operator Setup Order		
Timestamp	**Data**	**Value**
YYMMDDHHMMSS	OSO ID	SOO-Timestamp
YYMMDDHHMMSS	Operator ID	Operator ID
YYMMDDHHMMSS	Station ID	Station ID
YYMMDDHHMMSS	Setup property	Parts / Tools / Maintenance
YYMMDDHHMMSS	Setup property type	Type of Parts / Tools / Maintenance
YYMMDDHHMMSS	Setup property number	Number of Parts / Tools/ Maintenance
YYMMDDHHMMSS	OSO State	Waiting /In Process /Completed / Failed

5.3 BAS working scenario adjustments

Introduction of BAS Cloud directly influences on BAS working scenario. In this part BAS normal working scenario is adjusted. This scenario means the execution of uninterrupted sequence of assembly orders. System start-ups, shut-downs and interruptions are not considered. A sequence of system orders is formed by sub-ordinated subsystem according to the logic of working cycles. As a result of this process queues of orders for mobile robots, assembly stations, transport system and operators come to the target state module of the shop floor scheduling control unit. An order for mobile robot is called robot assembly order (RAO).

Shop Floor Scheduling Control vertically uploads robot assembly orders to a cloud according to the first-in-first-out principle. For each RAO data file is created, as shown in Table. 5.4.

This file contains RAO ID and Product Type. According to this data, pallet type and number of assembly steps for this product are uploaded from the Product Technological Data File. After this, RAO is ready to be processed. Mobile robots are located in the pool of robots. There are three possible robot states: Switched-off, Stand-by and Active. A number of stand-by robots in a system are regulated by shop floor scheduling control unit. A robot working algorithm is shown in Fig. 5.3. It consists of 5 sections. These sections describe following functions of mobile robot:

1. Search of RAO
2. Reservation and preparation of RAO
3. Assembly sequence loop
4. "Best operation" search loop
5. Post-assembly actions

***Section 1*.** All Stand-by robots check if there are any available orders on the cloud. If there is an order waiting, it proceeds to section 2. If not, it repeats the procedure.

***Section 2*.** When an order is found, robot reserves it by horizontally uploading its own ID number into the cloud. This data comes to Robot ID field of the RAO data file (Table 5.4).

From this file robot downloads the required pallet type and number of assembly steps. After that it goes to the pool of pallets and gets a pallet of the specified type. Each pallet has an information tag, containing ID and quality state of Pallet. When the palette is loaded on the robot, the tag gets scanned.

Robot gets Pallet ID and quality state information from this tag and uploads it to the cloud. This data comes to Pallet ID and Pallet Quality State fields of the RAO data file. After this section the robot is ready to start the assembly sequence according to the AO data.

***Section 3*.** To assemble one product robot has to make a number of assembly operations. Some of these operations depend on an assembly state of the product; some could be done only in strict order. This number of assembly operations is called assembly sequence. To complete an assembly sequence, robot has to make a fixed number of assembly steps (NAS), one operation per step. NAS is stored in RAO Data File in the cloud. Efficiency of assembly system depends directly on ability to choose an assembly sequence based on the best operation for the next assembly step.

A process of the search of assembly operation is described in the Section 4. When assembly station for the next assembly step is chosen, robot horizontally uploads station and operation ID to the corresponding assembly step field in RAO Data File. The last operation of an assembly sequence is packaging. Robot drives to a packing station and leaves a product on it. When assembly sequence is completed, robot starts post-assembly actions (Section 5).

***Section 4*.** For each assembly step robot has to find the best station from available ones. Availability of an assembly operation depends on two factors: physical presence of the station for this operation and technological possibility to make this operation on this stage of assembly. A number of previously completed operations, required for the processing of the next one are called preconditions set. This set is available for each operation and stored in the technological data file of each product.

For each of assembly operations robot checks its availability by comparing preconditions set with an own list of completed operations. If all preconditions are satisfied and this operation was not completed yet, robot requires station numbers and operation times suitable for this assembly operation. A process of the best station search is based on the smallest time resistance criteria. That means that from all suitable stations robot would choose the one with the smallest assembly time (*Tas*). This time sums from the transport time, waiting time and operation time.

Before the "Best operation" search loop *Tasmin* is set to infinity. On each iteration of a loop, if operation time of a chosen station is smaller than *Tasmin,* the current value of *Tasmin* would be replaced with *Tas*. After the loop robot gets a number of the required station.

***Section 5*.** Robot gets an empty pallet from the packing station and checks if there is any available order for this pallet type on a cloud. If yes, robot reserves it by horizontally uploading its own ID number to the cloud. Then it horizontally uploads Pallet ID and Pallet Quality State fields of the RAO data file. If there are no available orders for this pallet, robot brings the pallet back to the pool of pallets. After that it gets a new state from the shop floor scheduling control unit. If it stays Standby, it goes to Section 1. If it gets Switched-off state, it drives to the pool of robots and shuts-down automatically.

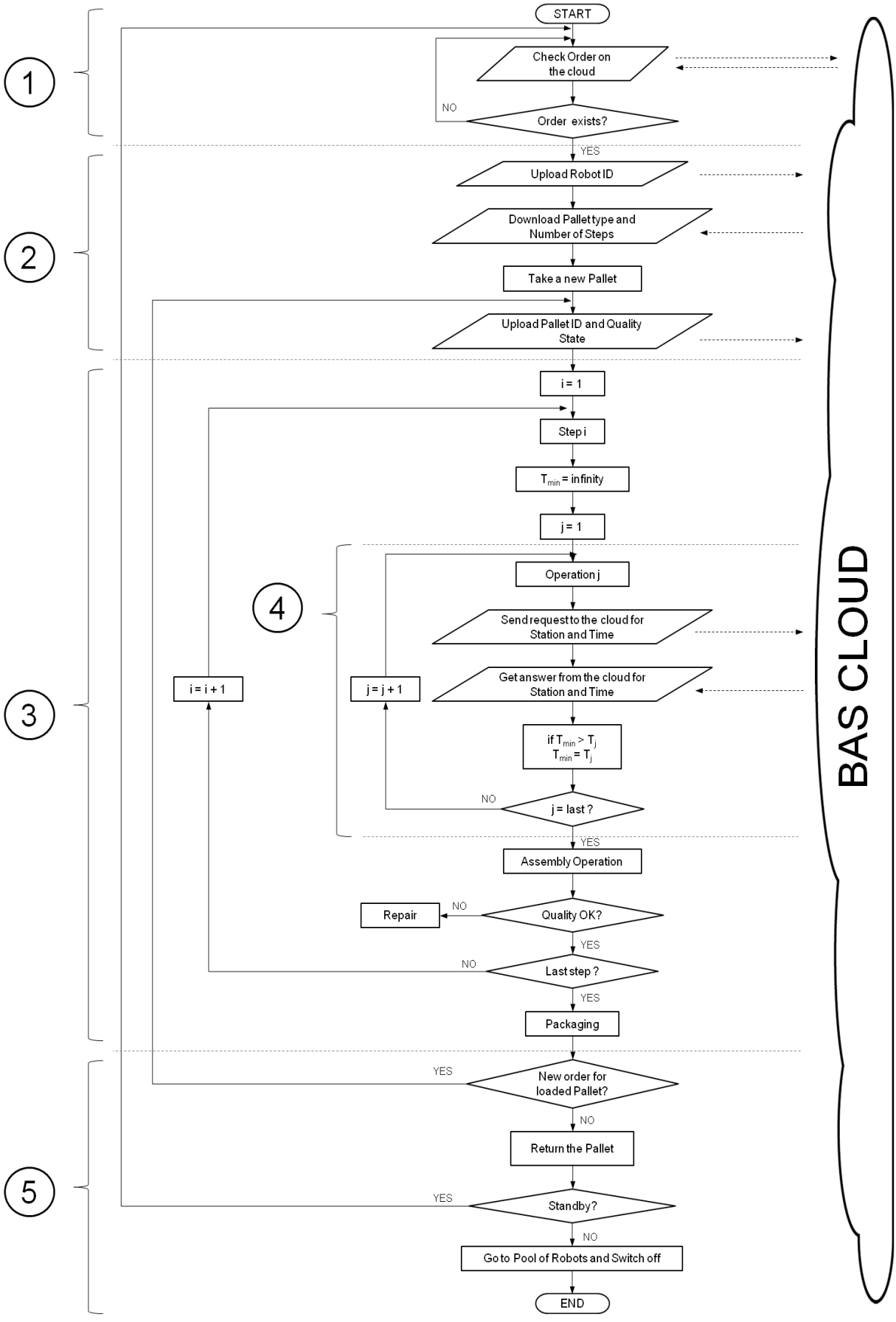


Fig. 5.3. Mobile robot working algorithm (Katalinic at al., 2014)

5.4 Summary

To organize further BAS implementation there is a need to make a number of adjustments of initial FAS and BAS concept. These improvements include:

- **Layout reorganization.** System layout is reorganized according to suit the BAS concept. The main adjustment is a change of current transport systems to a swarm of mobile robots and stations adjustments. Also the minimal number of stations with docking slots, which cover assembly operations of two product families is analysed.

  Reorganization a control structure. Initial FAS control structure and BAS hybrid control structure were adjusted for further implementation. Integration of cloud communication into the control structure of Bionic Assembly System (BAS) brings a number of advantages. In the comparison to the classical control structures this solution promises improvement of system performances and increase of system robustness. Introduction of a cloud helps to organize horizontal communication on BAS Shop-Floor in a direct and simple way. It changes the “everyone to everyone” communication to a direct “element - cloud” interactions.

- **System behavior adjustments.** BAS working scenario is adjusted according to the control structure changes. Therefore a new algorithm for mobile robot behavior is developed. This algorithm allows real time sequence planning considering system states. This is achieved by development of a new procedure, based on the selection of the best operation for the next assembly step.

These performed improvements allow the next step of BAS implementation for electrical motors facility reorganization.

Chapter 6

BAS Implementation

The main idea of this chapter is to describe a creation of new BAS layout for combined assembly system. This includes initial stations organization and placement. All system elements as well as control structure are adjusted for this new layout. Initial placement must be organized with minimal number of assembly stations to cover the assembly of two motor families. The second part of this chapter is focused on the initial flow organization. This includes the organization of the preconditions sets for the real-time robot sequence planning.

6.1 BAS layout organization

There are number of improvements in the layout design in comparison with original FAS. The most sufficient is a switch from the conveyor belts to the BAS mobile robots. The mobile robots realization may be organized in different ways. For example, AMUR mobile robots (Pryanichnikov at al., 2012).

Other improvements are introduction of the core and supplementary subsystems and introduction of the transport system. A new system layout is shown in a Fig. 6.1.

New system layout is reconfigurable. That means there is a possibility to change the total number of stations and the positions of movable stations. Number of stations is reduced to 26. This number includes 16 stations for assembly operations, 5 quality control stations, 2 pools for robots and pallets storage and 2 post-assembly stations - repair and unloading-packaging.

Extra stations, which did not participate in assembly operations, are stored in supplementary subsystem. The combined system uses 18 stations for double-sided motor family assembly and quality control and 13 stations for one-sided motor family. Description of system elements is shown in Table. 6.1. There are following improvements in assembly stations design:

- 2^{nd} End shield station is combined with End shield station for another product family. Both operations are made within 2ES-ES station group.

- Stator is preassembled and measured inside the group STATOR. After this procedure stator is assembled to a product-in-assembly.
- Rotors is preassembled and measured inside the group ROTOR. Rotor is assembled with spring, disc brake, safety rings and ball bearing. After this procedure rotor is assembled to a product-in-assembly.
- Fixing Plate is prepared and pre-assembled inside the group FP-AS. After this procedure Fixing Plate is assembled to product-in-assembly.
- Examination of axial clearance and Connection dimensions. These alternative stations are combined for both motor families. These stations are quality control stations QC-1-1/1-2 and QC-2-1/2-2. That means robots could choose either one or another from alternative stations for quality control operations.

Table 6.1. Layout elements description

Num	Abbreviation	Station / Group name
1	ES1	1st End shield
2	ES2 / ES	2nd End-shield / End-shield
3	BD1	Balancing Disc Station - 2
4	BD2	Balancing Disc Station -1
5	C1	Capacitor Assembly - 1
6	C2	Capacitor Assembly - 2
7	V1	Fan Assembly
8	FP-AS	Fixing plate
9	FIX1	Fixture of Motor and Electrical Components - 2
10	FIX2	Fixture of Motor and Electrical Components -1
11	HEAT1	Heating Station
12	ROTATE	Rotating Station
13	ROTOR	Rotor
14	GUM1	Rubber buffers control
15	GUM2	Rubber buffers screw
16	STATOR	Stator
17	PAK	Unloading/Packing
18	QC1-1	Quality control-1
19	QC1-2	Quality control-2
20	QC2-1	Quality control-1
21	QC2-2	Quality control-2
22	QC3	Quality control-3
23	REP	Repair Station
24	POOL-P	Pool of Pallets
25	POOL-R	Pool of Robots
26	HEAT2	Heating Station

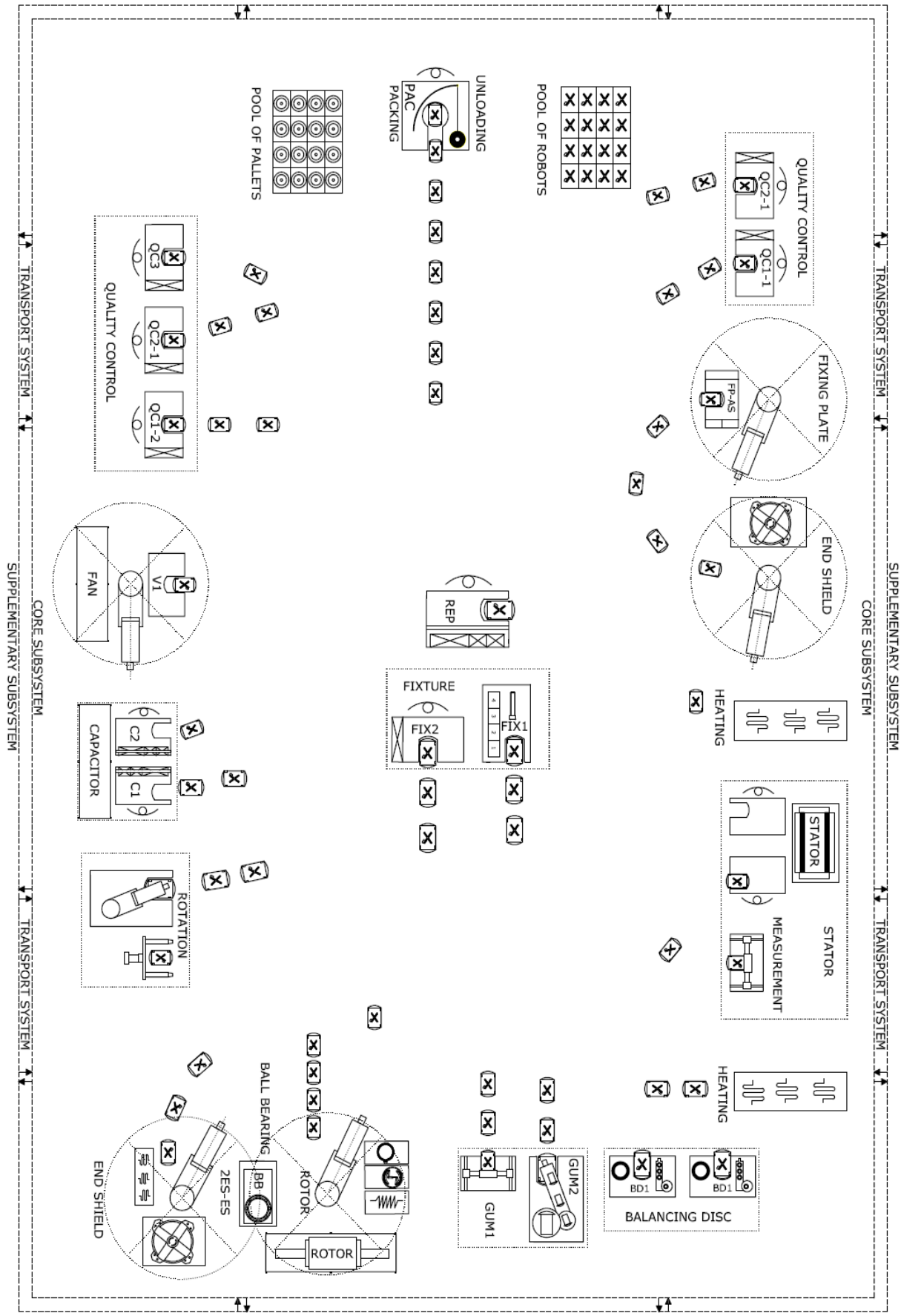


Fig. 6.1. Combined system layout

6.2 BAS assembly flow organization

Efficiency of assembly system depends directly on the ability to choose an assembly sequence based on the best operation for the next assembly step. Designed layout opens additional possibility for assembly sequence planning. This means that at some assembly step mobile robot could choose not only an alternative station for an assembly operation, but also alternative operation for current assembly step. Information about such operations is stored in technical data file called mobile robot assembly matrix, as shown in Tab. 6.2 and Tab.6.3. During the assembly process robot checks this file to see, which operations are possible to carry on with at his assembly step. Some operations could be done only in one strict sequence. To ensure this sequence, file includes a list of preconditions for each operation. Precondition is a set of operations, which must be previously made on product-in-assembly. If all preconditions are fulfilled, robot could make this operation on current assembly step. If not, it has to search for another operation.

It could happen that some operations have the same preconditions. These operations are called alternative operations. In Tab. 6.2 operations 12 and 14 on product family one are alternative ones. That means that technologically there is no difference to assemble fan or capacitor first into the product-in-assembly. When robot has an alternative operation, it has to choose one based on the "best" operation for the next assembly step algorithm. When assembly step is completed robot repeats search procedure again.

Tab.6.3. shows that there are three alternative operations with the same preconditions, however for operation 6 on product family two there is one linked operation. That means that it is technologically impossible to make other operations between these two operations. The reason for this case is that product-in-assembly must be preheated before the stator assembly. Therefore operation 7 is linked to operation 6 on product family two.

Example of assembly-flows for one- and double-sided motor families is shown at Fig. 6.2. Assembly sequence for double-sided motor family is shown with blue line and for one-sided motor family with red line.

Assembly sequences for both product families start from pool of pallets and finish at unloading/packing station. For double-sided motor family pairs of operations (12/14) and (16/17) are alternative. Moreover, operation 16, as well as 15 could be done at alternative assembly stations. That means at robot's 16^{th} assembly step it will choose between three stations: QC2-1, QC2-2 and QC-3.

For one-sided motor family operations (3/4/6) are alternative. Number of alternatives for the next assembly step will increase if robot will choose operation 3 or 4. If it will choose operation 6, the next linked operation will be 7. Operations 11 and 12 could be done at alternative assembly stations. With three alternative operations for assembly step one-sided motor family will have more variations of assembly sequences.

Tab. 6.2. Mobile robot assembly matrix for double-sided motor family

Operation	Station	Preconditions
Op1-1	POOL-P	
Op1-2	ES-1	Op1-1
Op1-3	HEAT	Op1-1 Op1-2
Op1-4	STATOR	Op1-1 Op1-2 Op1-3
Op1-5	BD1	Op1-1 Op1-2 Op1-3 Op1-4
Op1-6	BD2	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5
Op1-7	ROTOR	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6
Op1-8	2ES / ES	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7
Op1-9	ROTATE	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8
Op1-10	FIX-1	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9
Op1-11	FIX-2	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10
Op1-12*	C-1	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10 Op1-11
Op1-13	C-2	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10 Op1-11 Op1-12
Op1-14*	V-1	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10 Op1-11
Op1-15	QC1-1 / QC1-2	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10 Op1-11 Op1-12 Op1-13 Op1-14
Op1-16*	QC2-1 / QC2-2	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10 Op1-11 Op1-12 Op1-13 Op1-14 Op1-15
Op1-17*	QC-3	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10 Op1-11 Op1-12 Op1-13 Op1-14 Op1-15
Op1-18	PAK	Op1-1 Op1-2 Op1-3 Op1-4 Op1-5 Op1-6 Op1-7 Op1-8 Op1-9 Op1-10 Op1-11 Op1-12 Op1-13 Op1-14 Op1-15 Op1-16 Op1-17

* - Alternative assembly operations

Tab. 6.3. Mobile robot assembly matrix one-sided motor family

Operation	Station	Preconditions
Op2-1	POOL-P	
Op2-2	2ES/ES	Op2-1
Op2-3*	ROTOR	Op2-1 Op2-2
Op2-4*	GUM1	Op2-1 Op2-2
Op2-5	GUM2	Op2-1 Op2-2 Op2-4
Op2-6*	HEAT	Op2-1 Op2-2
Op2-7**	STATOR	Op2-1 Op2-2 Op2-6
Op2-8	FIX-1	Op2-1 Op2-2 Op2-3 Op2-4 Op2-5 Op2-6 Op2-7
Op2-9	FIX-2	Op2-1 Op2-2 Op2-3 Op2-4 Op2-5 Op2-6 Op2-7 Op2-8
Op2-10	FP-AS	Op2-1 Op2-2 Op2-3 Op2-4 Op2-5 Op2-6 Op2-7 Op2-8 Op2-9
Op2-11	QC1-1 / QC1-2	Op2-1 Op2-2 Op2-3 Op2-4 Op2-5 Op2-6 Op2-7 Op2-8 Op2-9 Op2-10
Op2-12	QC2-1 / QC2-2	Op2-1 Op2-2 Op2-3 Op2-4 Op2-5 Op2-6 Op2-7 Op2-8 Op2-9 Op2-10 Op2-11
Op2-13	PAK	Op2-1 Op2-2 Op2-3 Op2-4 Op2-5 Op2-6 Op2-7 Op2-8 Op2-9 Op2-10 Op2-11 Op2-12

* - Alternative assembly operations

**- Linked operation

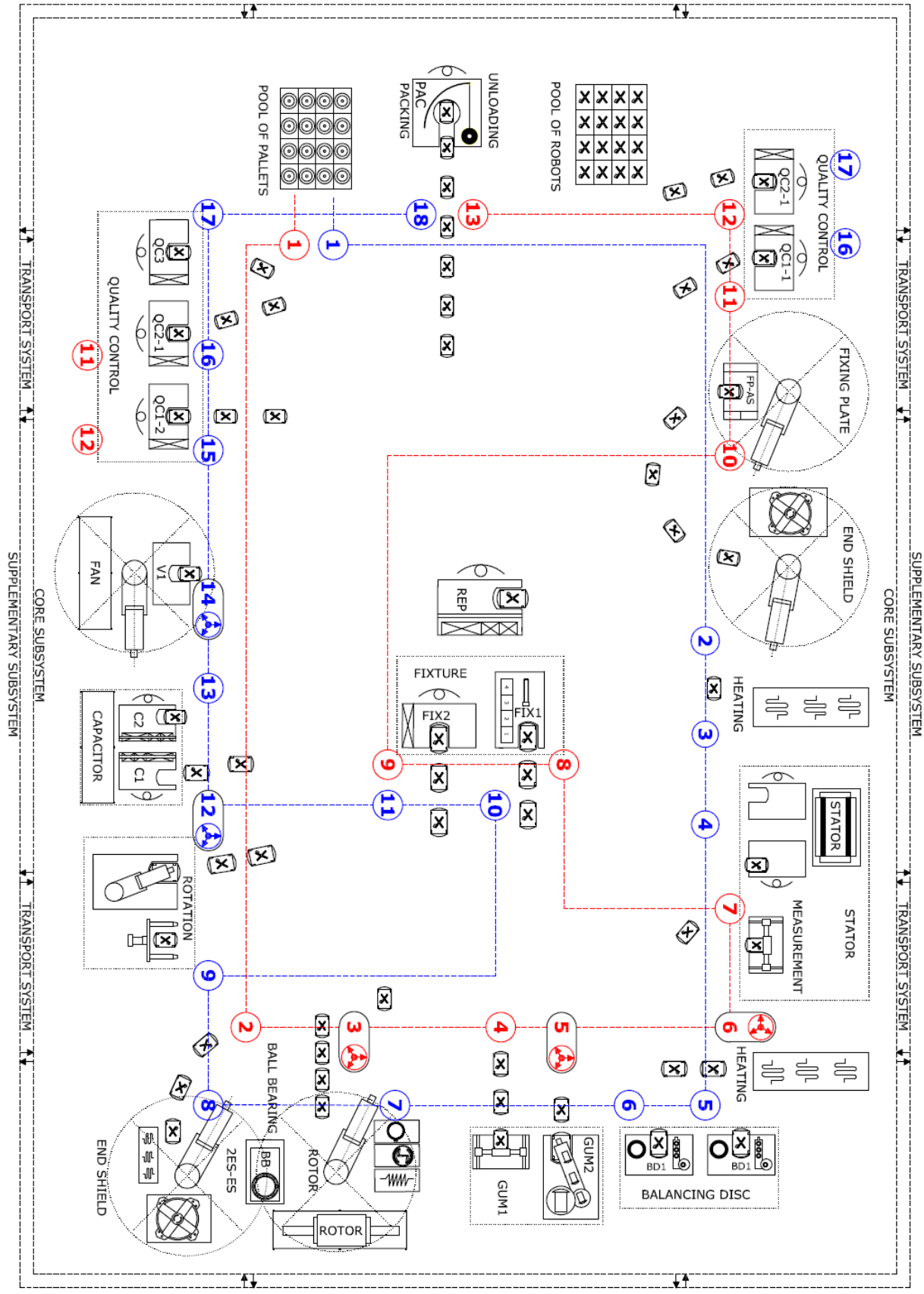


Fig. 6.2. Example of assembly-flows for one- and double-sided motor families.

6.3 Summary

Combined layout variability opens a possibility to solve the problem of stations underutilisation. Proposed robot algorithm coupled with possibility to choose alternative operations for assembly step could give stability to the system flow and cover the differences between stations assembly times. Possibilities to vary number of stations within a layout would help to solve bottlenecks problem.

Introduction and adjustment of BAS concept solves the requirements of stations setups and inclusion of human into the assembly. The workers are included as manual assembly stations.

Developed BAS algorithms and scenarios must be verified and simulated. Next section deals with these tasks.

The designed system will have a number of BAS features, which are absent in centralized FAS. These features are summarized in Table 6.4.

Table 6.4. Characteristics of FAS and BAS

Characteristic	**FAS**	**BAS**
Assembly stations	Expensive	Cheap
System information	Global	Global and Local
State data collection	Complex	Easy
Computing speed	Slow	Quick
System failure robustness	Low	High
Operation mode	Real time	Real time
Workers introduction	Hard	Easy

Chapter 7

BAS Simulation Organization

This chapter describes the tool for behavior and characteristics verification. Physical implementation of assembly system is not simple for a number of reasons (technological, financial, etc). Therefore other tool for observation and verification of the general system scenarios without its physical implementation is needed. The most suitable tool for these processes is simulation of system behavior and characteristics.

Simulation is defined by (Shannon, 1975) as "*the process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behavior of the system and /or evaluating various strategies for the operation of the system.*"

Simulation tool for designed system must help to understand its behavior and/or evaluate various strategies for the system control. There are different model parameters exist in literature. Main ones are shown in Tab.7.1.

Criteria	Parameters
Time influence	Static model
	Dynamic model
Changes in state of the system	Discrete model
	Continuous model
Input data	Deterministic model
	Stochastic model
Modeling approach	System Dynamic
	Discrete event modeling
	Agent based modeling

Tab. 7.1 Parameters of the model (Weigl, 2010)

Dynamic models depend on time and in static models the time plays no role. State of the continuous models changes, over time and in discrete ones, only at separated points of time. From the input data view, there are deterministic models, which use only pre-defined data, and stochastic ones, operating with a random data. Modeling approach is defined according to the input parameters of the system, number of elements within this system, and number of interconnections between them. There are three approaches to build dynamic simulation model:

- System Dynamics (SD)
- Process-centric (“Discrete Event”, DE) modeling
- Agent Based modeling (AB)

The first two were developed in the 1950-1960s and both use system-level top-down approaches. The agent based method is more recent. It uses bottom-up approach where the modeler focuses on the behavior of the individual objects. (Borshchev, 2014)

The system dynamics method is suitable for models with a high abstraction level and is used for problems at the highest or top level. Process-centric (“DE”) modeling is mainly used on level of operations and tactics, as shown in Fig. 7.1

Main difference of agent-based modeling approach is wide applicability range. The agents are all active elements inside the model. Example of agents could be people, companies, vehicles, cities, robots, products etc. To create the model modeler defines agents behavior (main reactions, drivers, states, etc), allocates them into a certain environment, establishes agents connections and organizes the simulation (Borshchev A & Filippov A, 2004).

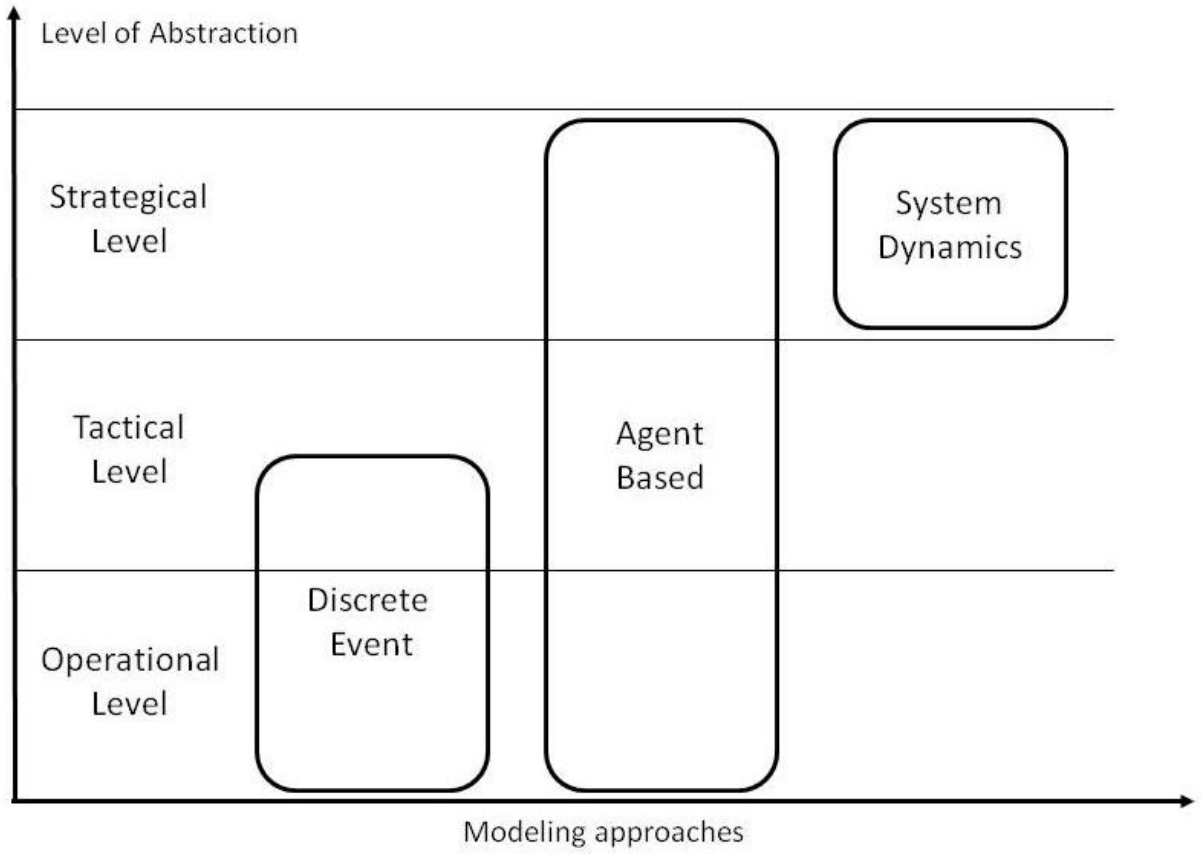


Fig. 7.1. Model approaches at different levels of abstraction (Siebers, 2014)

There are many simulation tools for modeling of assembly systems, but to model BAS hybrid control structure and system behavior both approaches must be combined.

Java-based AnyLogic tool is stated as *"the only tool that supports all the most common simulation methodologies in place today: System Dynamics, Process-centric (AKA Discrete Event), and Agent Based modeling."* (Anylogic, 2014).

It supports object-oriented model design, and therefore allows modular, hierarchical, and incremental construction of large models. Modularity gives it the possibility to connect the external databases; this feature allows constructing agents based on information stored in a database, or export/import data from the model. Other advantage of this software is functions scalability: needed functions and classes could be written by user at Java and integrated inside the program.

7.1 AnyLogic Software and Functions overview

This overview is limited to the special functions and elements, which are important for the BAS simulation. Functions and elements descriptions are based on Anylogic website (Anylogic, 2014a) and Anylogic help (Anylogic, 2014b) sections, available online. Additional information about Java programming language and its command syntax could be accessed from special literature, such as (Eckel, 2003).

The 7th version of Anylogic developed by XjTech company was realized in 2014. It has a number of versions for private, business, educational and research uses. University Edition is shown in Fig. 7.2. An interface consists of three parts project toolbar, main desktop and settings.

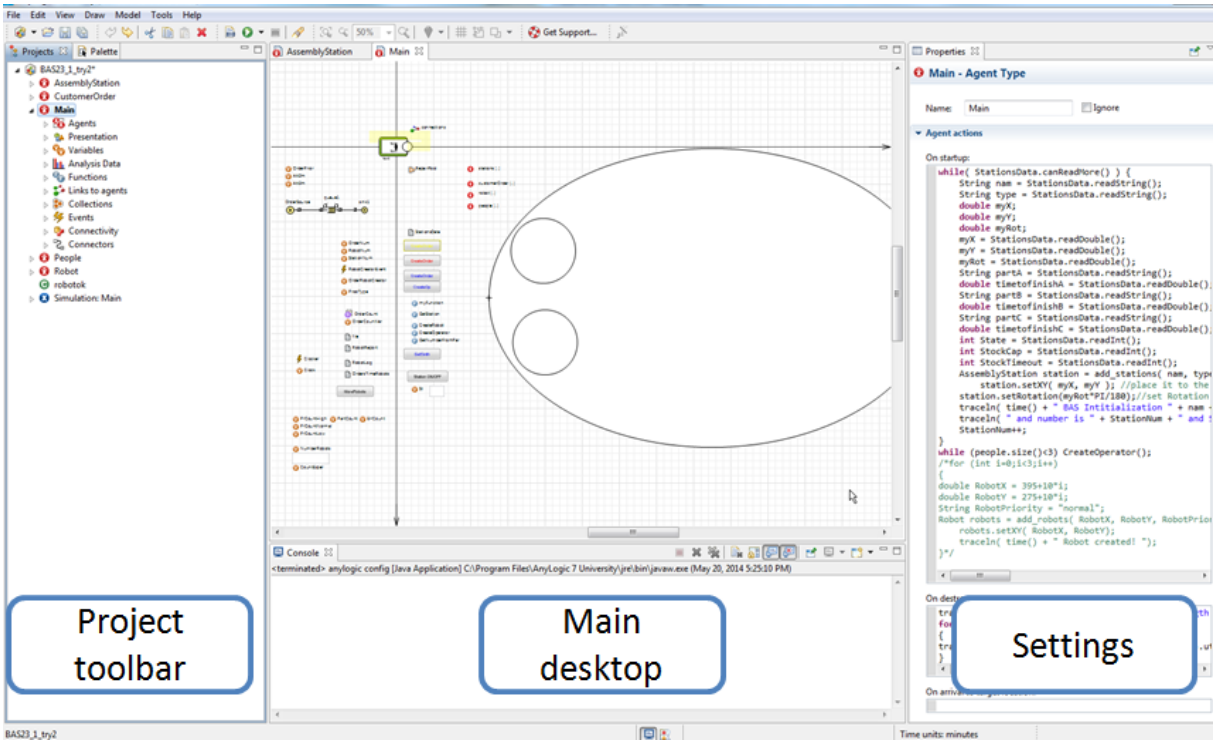


Fig. 7.2. Anylogic interface

- **Project toolbar.** This toolbar contains information about the project, its classes and elements.
- **Main Desktop.** Each class has its own empty desktop for physical presentation of its elements and functions.
- **Settings.** Properties of current element are displayed on the right side, at section settings.

Anylogic supports number of different libraries and functions. Functions relevant for modeling the assembly systems behavior are described below.

Functions. Anylogic uses own libraries and functions and support user Java functions definitions. User can call the functions from any place of the model. As an answer, function will return the value of an expression. Functions help to re-use same procedure more times within one model.

Database Connectivity. There is a possibility to link Anylogic model with databases using database connectivity tools. This tool is very useful for setting up the agents based on database, organization of log data and working with preset spreadsheets and files.

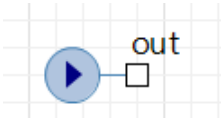
AnyLogic Enterprise Library This library allows using process-centric (discrete event) modeling paradigm. That means that the library allows to model real world system in terms of:

- entities such as pallets, products, parts, etc.
- processes including sequences of operations with queues, delays, resource utilization, etc.
- resources, such as stocks, parts etc.

Anylogic organizes processes in the form of flowcharts - a widely adopted graphical representation used in manufacturing, IT, logistics, business processes, etc. Flowcharts in AnyLogic are object oriented which enables the user to model large hierarchical, scalable and complex systems at any level of detail. Enterprise Library also allows organizing complex animations of process models. Key elements of Enterprise Library are discussed below:

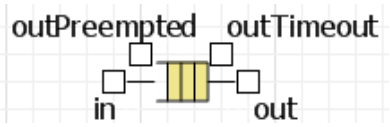
- **Entity.** It is a base class for all units that are generated, take part in the process flow and use the resources in process-centric models. In manufacturing an entity may represent a product, a part, a piece of information, a pallet, a vehicle- anything that is an object in the process. Standard entity has a number of properties. These properties could be extended by organization of additional java class, with a set of properties defined by modeler.

- **Source.**



It is an element, which generates entities. The entities may be standard, or of any Modeler can customize the generated entities by specifying the entity type in New entity field, and then specifying the action that should be performed before the entity exits the Source object in On exit action field.

- **Queue.**

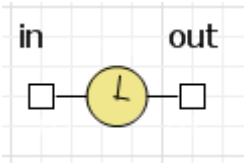


This element acts as buffer of entities waiting to be accepted by the next object in the process flow. Another function could be storage for the entities. Modeler can also remove entities programmatically from any position in the queue.

The queuing discipline is set to the First-In-First-Out (FIFO) by default and also supports priority-based sorting. There are two ways to set the priority. The first one is direct storage inside of the entity. The second one is priority calculation, which is based on the entity properties and/or external conditions. A queue which using prioritizing follows the algorithm:

1. Accept incoming entity
2. Evaluate its priority
3. Set entity place in the queue regarding its priority

- **Delay.**



This element delays entities for a preset amount of time. The delay time may be set dynamically and depend on the entity type, properties and other conditions. Delay in a simulation of assembly system may be used as assembly operation at assembly station, packing, repair or any other operations, which require time.

Enterprise library is very effective tool to model traditional linear assembly systems with a set tact time. Therefore this library is very useful to organize internal functions of assembly station. But for modeling self organizing shop floor, where elements properties and behavior are individual, it is important to include agent based modeling approach.

Agent-Based Modeling. For the practical applications agent based modeling can be defined as essentially decentralized, individual-centric approach to model design. The global system-level behavior comes as a result of interactions of many individual behaviors.

AnyLogic supports Agent Based modeling (as well as Discrete Event and System Dynamics Modeling) and allows to efficiently combine it with other approaches (Borshchev, 2014). A number of agents types in Anylogic model cannot be less than two. The first agent is a Main class. This is a class for a top-level object, which creates the environment. Another class creates an agent which act at this environment. Each agent has their own properties and functions.

There is a big number of agents applications in literature. In order to define the agents properties for practical applications, number of statements is presented. These statements are based on the continuous research at the field of agent based modeling for practical applications (Borshchev, 2014):

- ***Agents are not the same thing as cellular automata*** *and they do not have to live in discrete space (like the grid in The Game of Life). In many agent based models space is not present at all. When space is needed, in most cases it is continuous, sometimes a geographical map or a facility floor plan.*
- ***Agent based modeling does not assume clock "ticks"*** *or "steps" on which agents test conditions and make decisions (synchronous discrete time). Most well-built and efficient agent based models are asynchronous (conditions are tested and things happen only when they need to). Continuous time dynamics may also be a part of agent or environment behavior.*
- ***Agents are not necessarily people.*** *Anything can be an agent: a vehicle, a piece of equipment, a project, an idea, an organization, an investment. A model of a steel converter plant where each machine is modeled as an agent and steel is produced as a result of their interaction, is an agent based model.*
- ***An object that seems to be absolutely passive can be an agent.*** *For example, a pipe segment of a water supply network can be modeled as agent: we can associate with it maintenance and replacement schedules, cost, breakdown events.*
- ***There can be many and can be very few agents in an agent based model*** *(compare the model of American automobile market and the model of steel converter plant). Agents can be of the same type and can be all of different types.*
- ***There are agent based models where agents do not interact at all.*** *For example, in the area of health economics there are models of alcohol usage, obesity, chronic diseases where individual dynamics depends only on personal parameters and, sometimes, on the environment.*

Statecharts. Statechart are used to define more complex agents' behavior. It is the most advanced tool to describe event- and time-driven agent behavior. Statecharts also describe the state space of a given algorithm, the events that cause a state transition and the actions

leading to state change. Statecharts open a wide variety of discrete behaviors, which is not limited to just idle/busy, open/closed, or up/down status offered by most of block-based tools. The behavior of a statechart is defined in a graphical editor using the elements shown in Fig. 7.3.

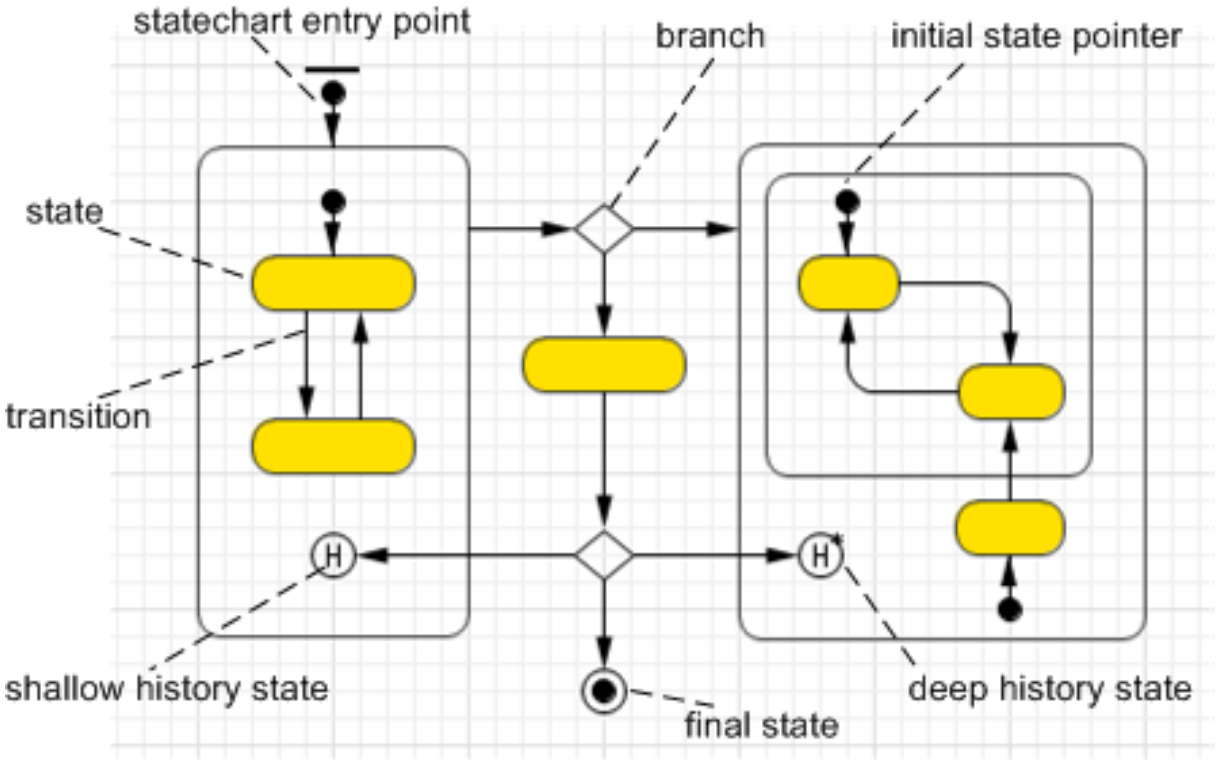


Fig. 7.3 Statechart elements description (Anylogic, 2014b)

Main elements of statechart are states and transitions. State represents a location of control with a particular set of reactions to conditions and/or events. Transitions may be triggered by user-defined conditions (timeouts or rates, messages received by the statechart, and Boolean conditions). Transition execution leads to a state change where a new set of transitions becomes active. States in the statechart may be hierarchical, i.e. contain other states and transitions. (Anylogic, 2014b)

Variables. Any class and function in AnyLogic could contain variables. There are two main functions of variables:

- Keep some data or object characteristics, which changes over time
- Keep the results of model simulation

There are two types of variables supported by AnyLogic – variables and collections. Variable is a single changeable element of a random scalar type or Java class. A collection represents a number of objects, known as its elements. Collections could have following functions:

- Group and operate with multiple elements inside one unit
- Keep and process aggregated data.

There are possibilities to store duplicate elements inside one collection, and organize them in one particular order.

7.2 BAS AnyLogic model organization

All acting elements inside the model are defined as classes as shown in Fig. 7.4. There are 6 classes in a project: Main, Customer Order, Assembly Station, Robot, Operator and robotok.

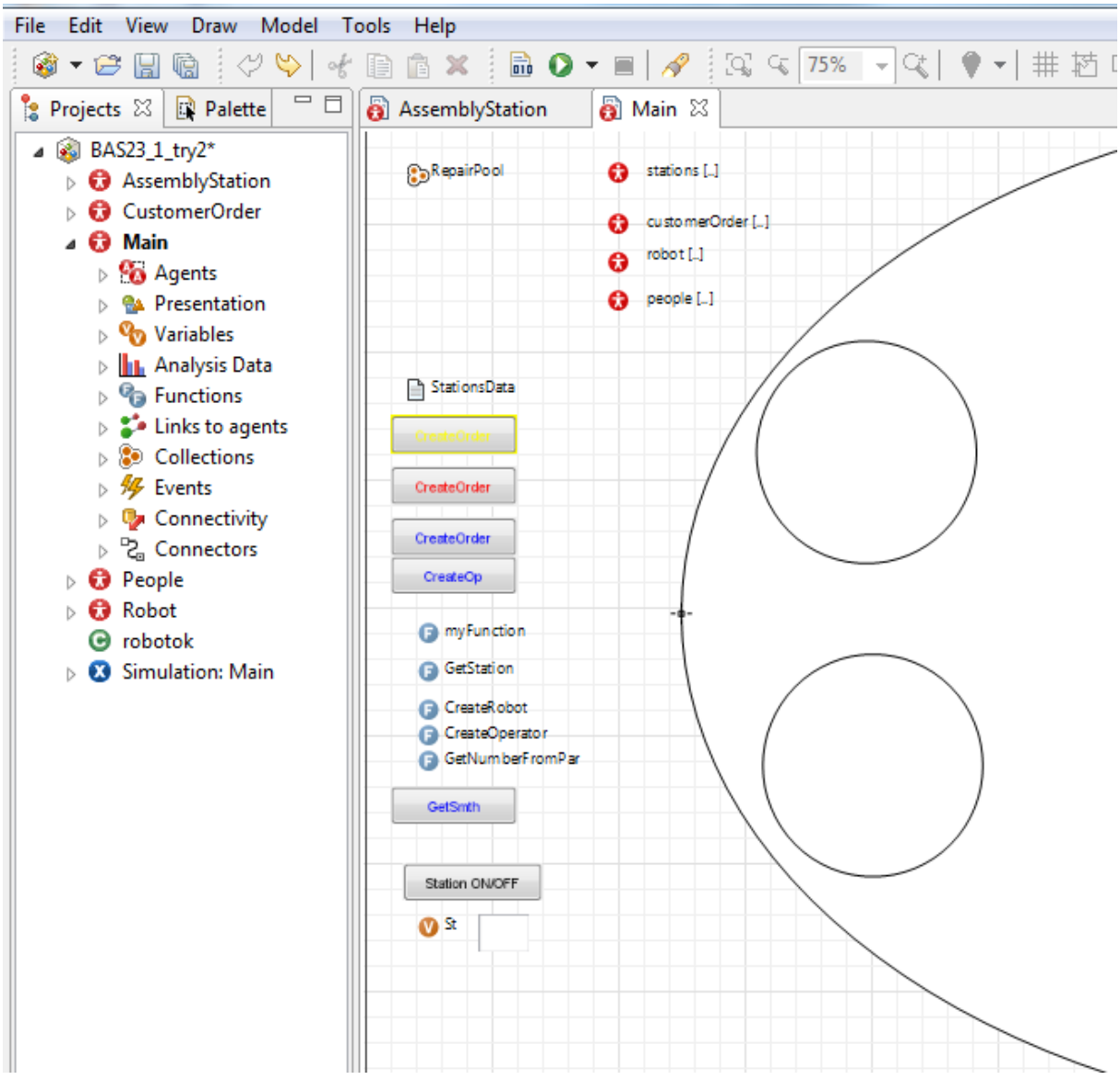


Fig. 7.4. BAS model classes in Anylogic

Main. This agent class is a parent class, which represents system shop floor and BAS Cloud. This class creates an environment, where all agents are acting and communicating. Main has four replicated agents' objects: orders, stations, robots and operators. It also holds the main functions needed for other agent classes. Main creates the number of agents of each type with defined properties. There are three ways how agents are defined at main.

1) Automatically from external database. Stations properties are taken from station technical input data file (Tab. 7.2), which is connected to main with database connectivity tool.

Tab. 7.2 Station technical input data file

Num	Name	X	Y	Rot	Ability 1	Time	Ability 2	Time, sec	State	Stock Capacity	Reset time
AS1	BD1	1075	135	90	Op1-5	21	X	21	1	5000	300
AS2	BD2	1185	140	120	Op1-6	17	X	17	1	5000	300
AS3	C1	915	680	270	Op1-12	20	X	20	1	500	600
AS4	C2	835	680	270	Op1-13	20	X	20	1	500	600
AS5	ES1	755	135	90	Op1-2	17	X	17	1	600	350
AS6	ES2/ES	1250	580	225	Op1-8	17	Op2-2	17	1	600	600
AS7	FIX1	840	470	150	Op1-10	14	Op2-8	14	1	10000	600
AS8	FIX2	790	385	150	Op1-11	22	Op2-9	22	1	1000	600
AS9	FP-AS	675	135	90	X	20	Op2-10	20	1	2500	600
AS10	GUM1	1325	385	180	X	14	Op2-4	14	1	600	600
AS11	GUM2	1325	300	180	X	14	Op2-5	14	1	5000	600
AS12	HEAT1	835	135	90	Op1-3	24	Op2-6	24	1	30000	600
AS13	HEAT2	995	135	90	Op1-3	13	Op2-6	13	1	30000	600
AS14	PAC	260	405	0	Op1-18	10	Op2-13	10	1	1000	600
AS15	QC1-1	570	170	60	Op1-15	8	Op2-11	8	1	500	600
AS16	QC1-2	675	680	270	Op1-16	19	Op2-12	19	1	500	600
AS17	QC2-1	505	205	60	Op1-15	10	Op2-11	10	1	500	600
AS18	QC2-2	570	665	300	Op1-16	10	Op2-12	10	1	500	600
AS19	QC3	505	630	300	Op1-17	8	X	8	1	500	600
AS20	REP	740	300	150	X	20	X	20	2	0	600
AS21	ROTATE	1075	680	270	Op1-9	12	X	12	1	30000	600
AS22	ROTOR	1325	470	180	Op1-7	21	Op2-3	21	1	500	600
AS23	STATOR	915	135	90	Op1-4	16	Op2-7	16	1	1000	600
AS24	V1	755	680	270	Op1-14	20	X	20	1	500	300

Main reads following station parameters:

- **Num.** Station number from one till the last according to the Station technical input Data File
- **Name.** Station technical input Data File. This name is also shown at the layout for operator orientation.
- **X.** Station coordinate X. According to this coordinate station will be aligned horizontally on the layout.
- **Y.** Station coordinate Y. According to this coordinate station will be aligned vertically on the layout.
- **Rot.** Station rotation angle. According to this angle station rotation will be aligned on the layout.
- **Ability N.** Station Ability. The operation number, which station is able to produce. There could be more than one option for each assembly station.
- **Time N, sec.** Station Time. Time, which is required to make the operation.
- **State.** Station state. 1 - online, 2 – offline.
- **Stock Capacity.** Station stock capacity. If this number gets to 0, station is not able to work anymore. Operator would need to make a setup.
- **Reset time.** Time needed to finish one station setup.

2) Manually with a button function. Customer orders and operators are created from the button, as shown at Fig. 7.5. Agent properties are set inside the function. Such agents could be created within an experiment any time on users click. It is possible to preset standard agent's properties, and they will be copied for each of agent. This allows user to take part at working scenario and model different system behaviors. The function CreateRobot() is shown in Tab. 7.3.

3) Automatically with event. Robot agents are activated with an event. This event is triggered when number of conditions is fulfilled. For example when number of robots is lower than optimal, Main class automatically creates additional robots using event.

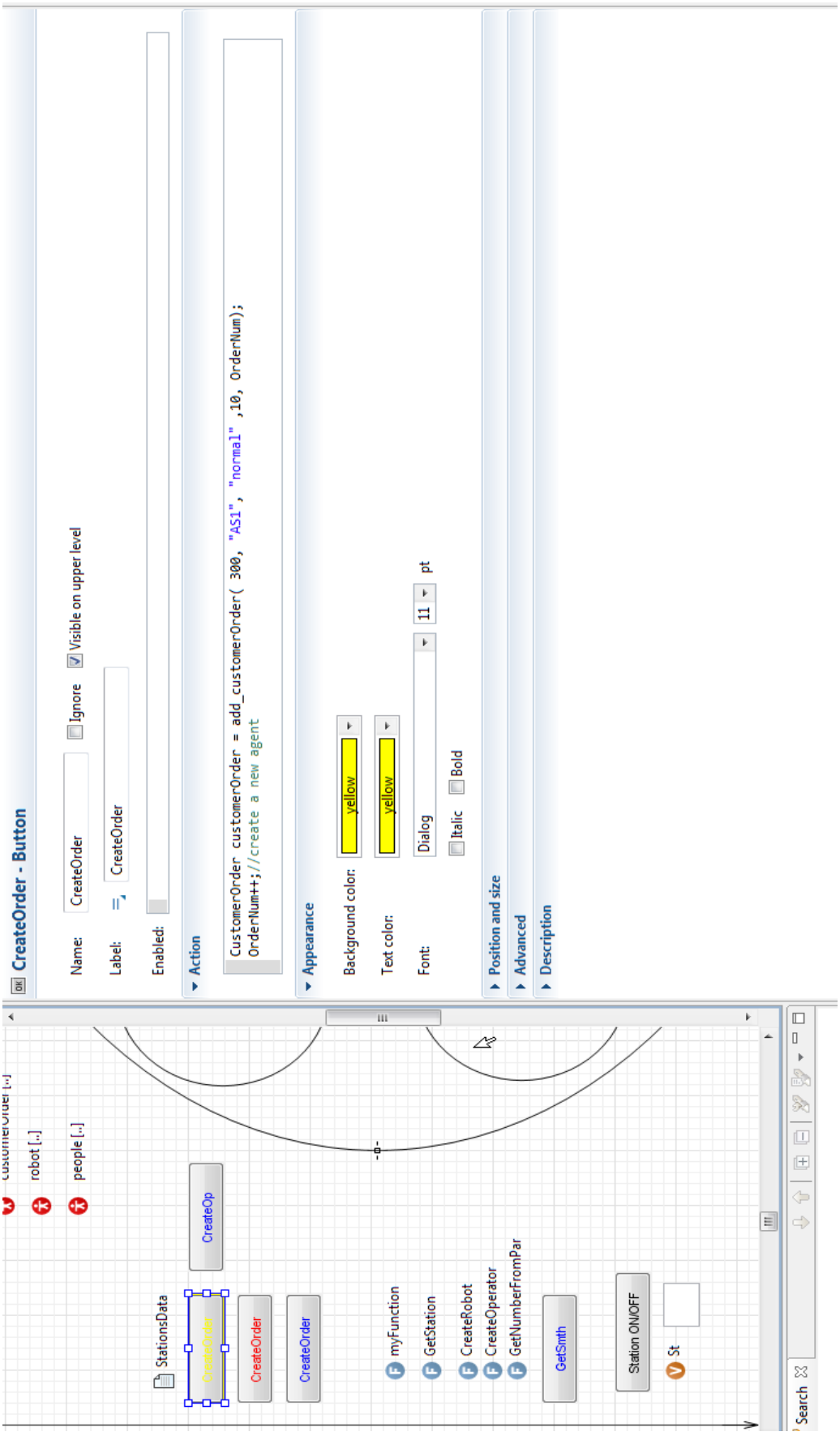


Fig. 7.5 Agent creation with a button function

Tab. 7.3. CreateRobot function of class main()

Parameters	
Name	**Value**
Static	false
Access Type	default
Use Units	false
Function body	

Body	double RobotX = 395;
	double RobotY = 275;
	//traceln(time() + " Robot created! ");
	Robot robot = add_robot(RobotX, RobotY, Priority, RobotNum, OrderNum, ProdType); //create a new agent
	robot.setXY(RobotX, RobotY);
	RobotReport.println(Math.ceil(time()*100)/100+ "; Robot "+RobotNum+"; Created!");
	RobotReport.close();
	traceln(time() + " Robot created! and RNum " + RobotNum + " OrderNum is " + OrderNum);

Customer order. This agent represents a customer, giving the set of assembly orders. Each customer order has its own priority, product type and volume. This orders come to the queue of orders. These orders are sorted according to the priority. When each order leaves the queue, a new RAO is generated.

Assembly station. This agent represents all stations inside the system. All initial station properties are created by Main according to Station technical input data file. At this case station has its number, name, abilities to make one or more operations, time for each operation, Stock Capacity and Reset time. Two last properties are needed for station setup modeling. Station setup process described by statechart, which is placed at the station desktop. Main desktop of Assembly station agent is shown in Fig. 7.6.

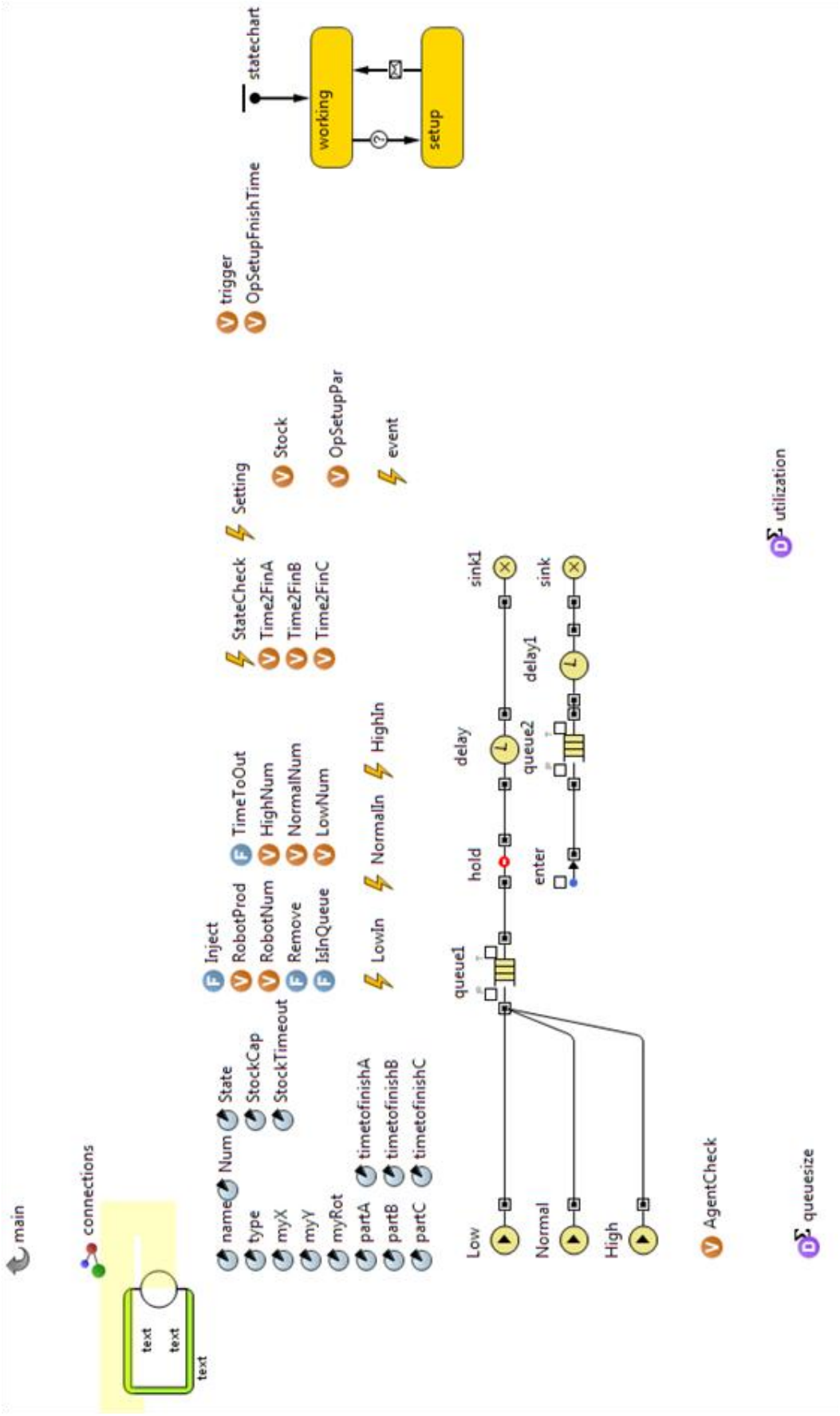


Fig. 7.6 Main desktop of Assembly Station agent

Statechart consist of two states: working and setup. These states are connected with two transitions. Transition to setup is triggered by condition: station stock equals to null. When it happens, station automatically gets to the setup state and blocks all other operations. The transition to working state is only possible if station will get a message about OSO successful finish.

Station presentation is at the left corner of desktop. If station is active, the color is green, and if not - red. Other desktop elements include station parameters, local variables, needed for the proper simulation functioning and station functions.

robotok. This is an additional entity class, which represents assembly pallet. It has additional properties and helps to connect assembly station and mobile robot agents. It has a number of properties, which hold information needed for modeling purposes. Structure of robotok entity is shown in Tab. 7.4.

Tab. 7.4. robotok entity properties

Name	**Value**

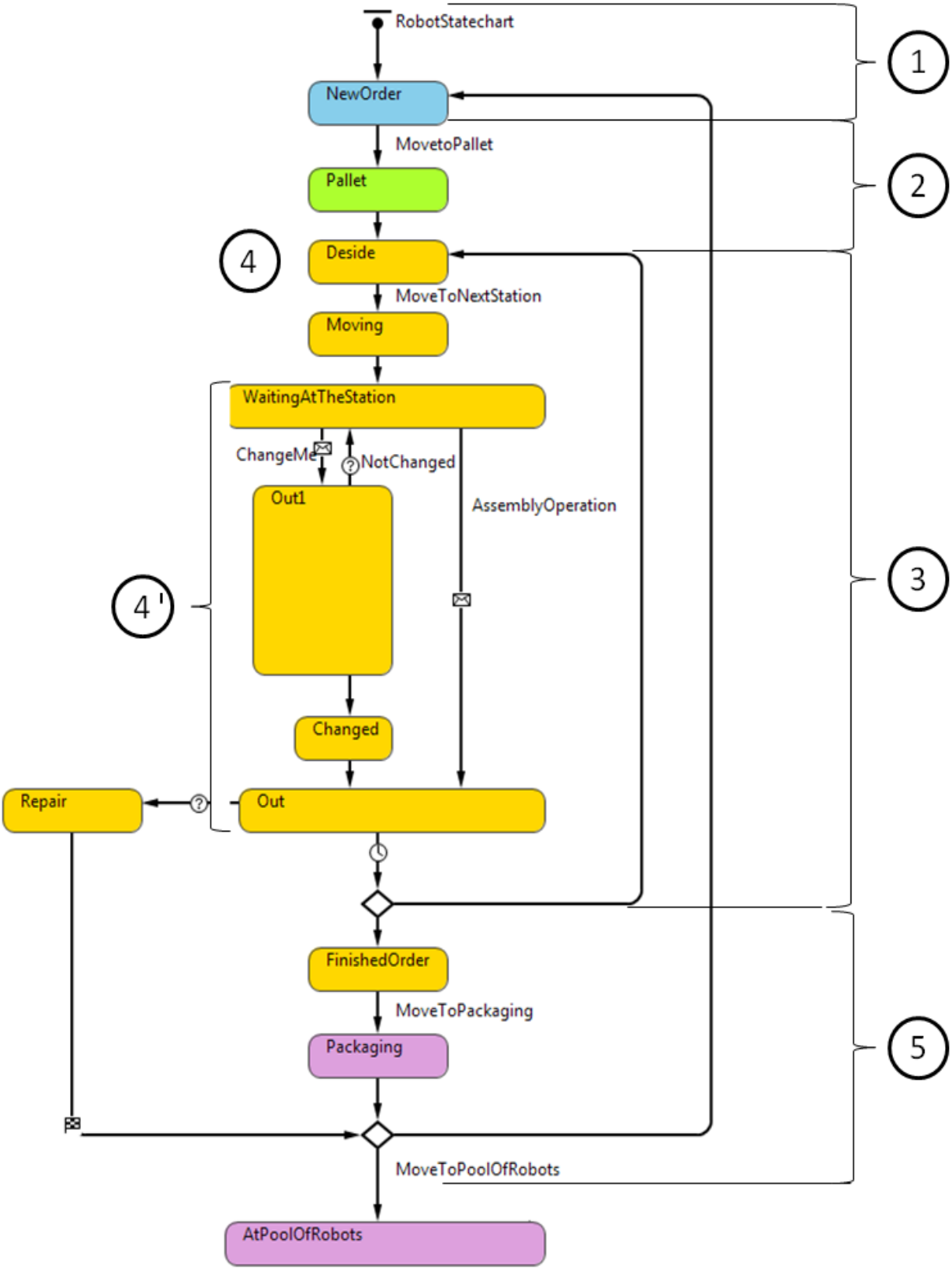
robotok	*
	*/public class robotok extends com.xj.anylogic.libraries.enterprise.Entity implements java.io.Serializable {
	
	String RobotName;
	int RobotPriority;
	double Time;
	double Att1;
	int Att2;
	String Att3;
	
	public robotok(){
	}
	
	/**
	* Constructor initializing the fields
	*/
	public robotok(String RobotName, int RobotPriority, double Time, double Att1, int Att2, String Att3){
	this.RobotName = RobotName;
	this.RobotPriority = RobotPriority;
	this.Time = Time;
	this.Att1 = Att1;
	this.Att2 = Att2;

```
                this.Att3 = Att3;
    }
        @Override
        public String toString() {
                return
                        "RobotName = " + RobotName +" " +
                        "RobotPriority = " + RobotPriority +" " +
                        "Time = " + Time +" " +
                        "Att1 = " + Att1 +" " +
                        "Att2 = " + Att2 +" " +
                        "Att3 = " + Att3 +" ";
        }
        private static final long serialVersionUID = 1L;
}
```

Robot. This agent class represents BAS mobile robot. All BAS robot behavior is organized in working algorithm. This algorithm consists of 5 sections:

1. Search of RAO
2. Reservation and preparation of RAO
3. Assembly sequence loop
4. "Best operation" search loop
5. Post-assembly actions

These sections are described in previous chapter. This algorithm could be transferred to Anylogic using statechart tools. Each section has a number of robot states and transitions between them. Transformed mobile robot algorithm is described by statechart of robot agent class shown in Fig. 7.7.



7.7. Robot agent class statechart

1. Search of RAO. When mobile robot is activated, it comes from entry statechart point to the NewOrder state. Transition to the next state will be triggered if only RAO is found.

2. Reservation and preparation of RAO. This section includes order reservation by signing its own ID into the RAO file, and physical shop floor movement to the pool of pallet. Movement is organized with use of moveTo(X, Y) function of Anylogic. The coordinates of Pool of Pallets are taken from BAS Cloud and used as X,Y for a moveTo function.

3. Assembly sequence loop. This loop includes four main states:

- Search of the new assembly station. The search loop is explained at Section 4.
- Moving to assembly station. As at the previous step, movement is organized using moveTo() function.
- Assembly operation. This step consists of two states - waiting in a queue and assembly operation. There is a possibility for a robot to search for alternative station while staying in a queue. It constantly makes the search loop, as seen in section 4' at Fig. 7.7. If the candidate station is found, robot signs out from the current queue and drives to another station.
- Repair. After of each assembly operation quality control is performed. Quality state failures are randomly generated for 2% of each product. When quality state is negative, robot moves to the repair station, where waits for an operator agent.

The number of assembly steps robot downloads from RAO File. The loop is repeated till the last assembly step is completed.

4. / 4' Best operation" search loop. For each assembly step robot has to find the best station from available ones. If all preconditions are satisfied and this operation was not completed yet, robot requires station IDs and operation times suitable for this assembly operation.

There is no predefined solution for this loop, therefore the search process is organized in two additional Java functions: ChooseOperations() and GetStation(). These functions are designed as separate functions within agent class robot, and could be used by any agent of this type. As an example ChooseOperations() function listing and properties are shown in Tab. 7.5. This function is responsible for checking operation availability by comparing preconditions set with list of completed operations. It reads the mobile robot assembly matrix, and for each operation compares list of finished operations with preconditions. When operation is suitable for the next assembly step it is copied to the Options collection.

When the operation options are chosen, robot starts GetStation() function from Main. This station searches for the best time to finish the assembly operation and returns station ID.

5. Post-assembly actions Robot gets an empty pallet from the packing station and checks if there is any available order for this pallet type on a cloud. If yes, robot reserves it by

horizontally uploading its own ID number to the cloud. Then it horizontally uploads Pallet ID and Pallet Quality State fields of the RAO data file.

If there are no available orders for this pallet, robot brings the pallet back to the pool of pallets. Movement is organized with use of moveTo(X, Y) function of Anylogic. The coordinates of Pool of Pallets are taken from BAS Cloud and used as X,Y for a moveTo function. After that it gets a new state from Main class. If it stays Standby, it goes to Section 1. If it gets Switched-off state, it drives to the pool of robots and shuts-down automatically.

Tab.7.5 ChooseOperations function of robot agent class

Parameters	
Name	**Value**
Static	false
Access Type	default
Use Units	false
Function body	

```
for (int j=1;j<=ProductCounter;j++)
{
	for(int b=2;b=get_Main().stations.size();b++)
	{
		if (ProductChoose.cellExists(1,j,b)==true)
		{
		if(preconditions.contains(ProductChoose.getCellStringValue(1,j,b))==false)
		{
			String p= ProductChoose.getCellStringValue(1,j,b);
			preconditions.add(p);
		}
		}
	}
	//traceln("j= "+j+" ; "+preconditions);

	if ((AssemblyMatrix.contains (ProductChoose.getCellStringValue(1,j,1))!=true)
	&&(AssemblyMatrix.containsAll(preconditions)))
	{
		Options.add(ProductChoose.getCellStringValue(1,j,1));
		traceln("j= "+j+" ;opts= "+Options);
	}
	preconditions.clear();
}
traceln("Robot "+Num+"; Step "+counter+"; Options "+Options);
```

People. This agent represents core subsystem operators. These operators are responsible for assembly, setup and repair orders. At first case they act as manual assembly station and get their SAOs as other stations. Operator statechart is shown in Fig. 7.8.

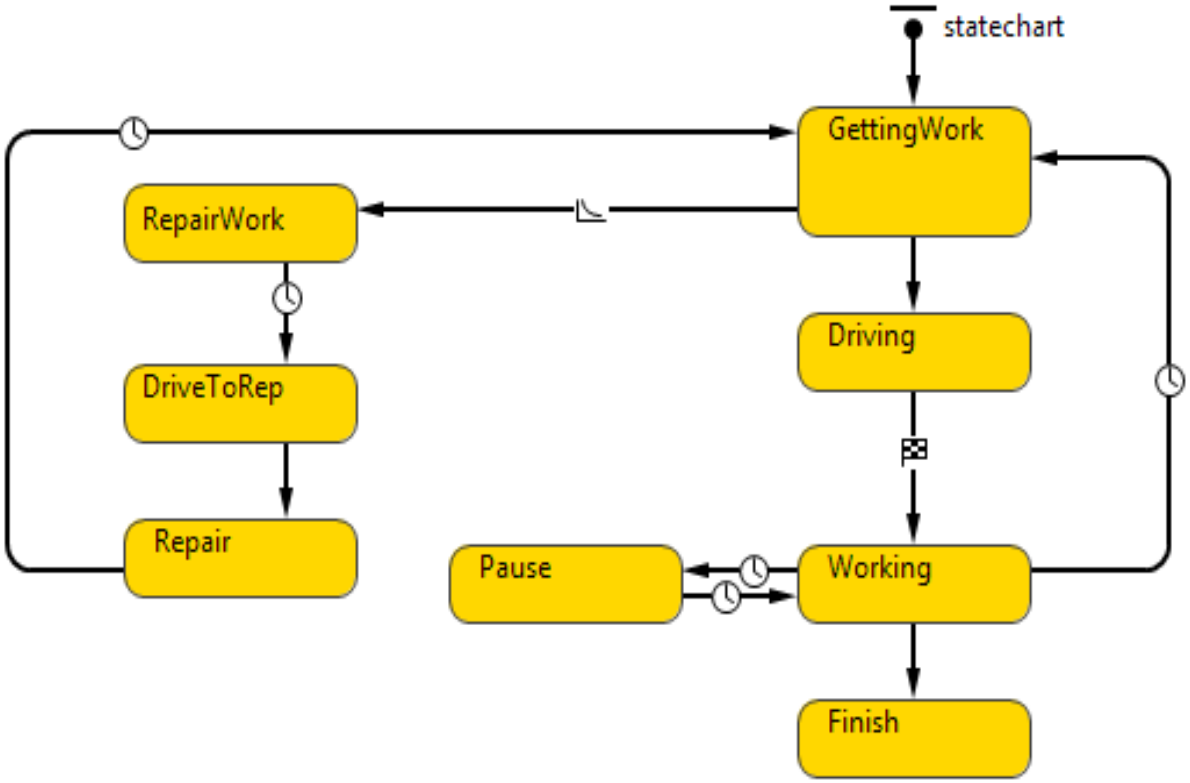


Fig. 7.8. Operator statechart

- **Getting Work.** Operator initial state is GettingWork. At this state operator gets Repair or Setup work. Setup work has a priority, and if there are setup orders, operator will choose them first.
- **Moving.** Operator moves to the chosen station for setup or repair station. The movement is organized using moveTo(X,Y) function. Operator agent uses preset path, called PolylineOperator, as shown in Tab. 7.6. At first case it gets X,Y from the chosen station, at the second case, from repair station.

Tab.7.6 Driving / DrivingToRep state description of operator agent class

Name	Value
General	
Entry Action	OpNewX=get_Main().stations.get(NextStNum).myX; OpNewY=get_Main().stations.get(NextStNum).myY; moveTo(OpNewX, OpNewY,get_Main().PolylineOperator);

- **Working.** Repairs are organized at one by one basis. That means, operator checks after each repair operation if there is an setup order. Station setups may last longer period of time, therefore operator pause time is also counted.
- **Pause.** Each operator gets his preset pause. This acts as a timer. When pause is over it gets back to his duties.
- **Finish.** Operator gets finished state when his shift is over. The shift length could be set as the preference and set to default eight hours.

7.3 Summary

This chapter focuses on design of the tool for system behavior and characteristics verification. The best candidate of this tool is simulation model. Designed tool allows observation and verification of the general system scenarios without physical implementation.

Designed tool, based on AnyLogic software allows the complete analysis of BAS behaviour and working characteristics. All BAS shop floor elements act as agents within one environment. Therefore tool allows to set the parameters of each BAS element and see the interactions in real-time. The tool is suitable for microanalysis of each element as well as general analysis of system behaviour and performances.

The main challenge is the description of BAS simulation tool and translation of BAS working scenarios and algorithms into the internal functions of AnyLogic. Developed additional JAVA functions allow to enlarge the tool possibilities and describe exact working scenario. Initial adjustments are made for the case of two product families assembly with minimal number of stations.

Chapter 8

Experiments and Analysis

This chapter presents BAS behavior analysis, based on simulation based tool. This allows to test different working modes, conditions and amount of needed robots and stations. The tool could be used to minimize the risk of bad investment in case of system increase. The experiments deal with following tasks:

1. **System behavior analysis.** General system analysis and verification. Initial layout verification and optimization. Search and optimization of system week points.
2. **Robot behavior analysis.** Robot working scenario analysis and algorithm verification. Analysis of dynamic sequence planning algorithm.
3. **System performance comparisons.** Comparison of system behavior for single product family and both product family runs.
4. **Comparison of FAS and BAS.** The comparisons of main characteristics, modes of assembly, control system, workers integration and reconfiguration.

8.1 System behavior analysis

In this experiment BAS diagnostics and its internal reserves and reaction will be analyzed and verified. The experiment is organized to verify and optimize initial system layout using internal system reserves.

The goal of experiment is to understand, if simulation tool could show system weak points and the ways to improve them.

The assembly mode in first experiment is BAS normal working mode that means that system works without breaks and interruptions at continuous mode of assembly. Three system operators are responsible for system setups and work continuously without technical breaks. Initial layout consists of 24 stations, located as shown in Fig. 8.1.

Initial layout is organized with a minimal number of stations, and other ones are stored at the supplementary subsystem. The goal is to find an optimal stations combination for a given product run.

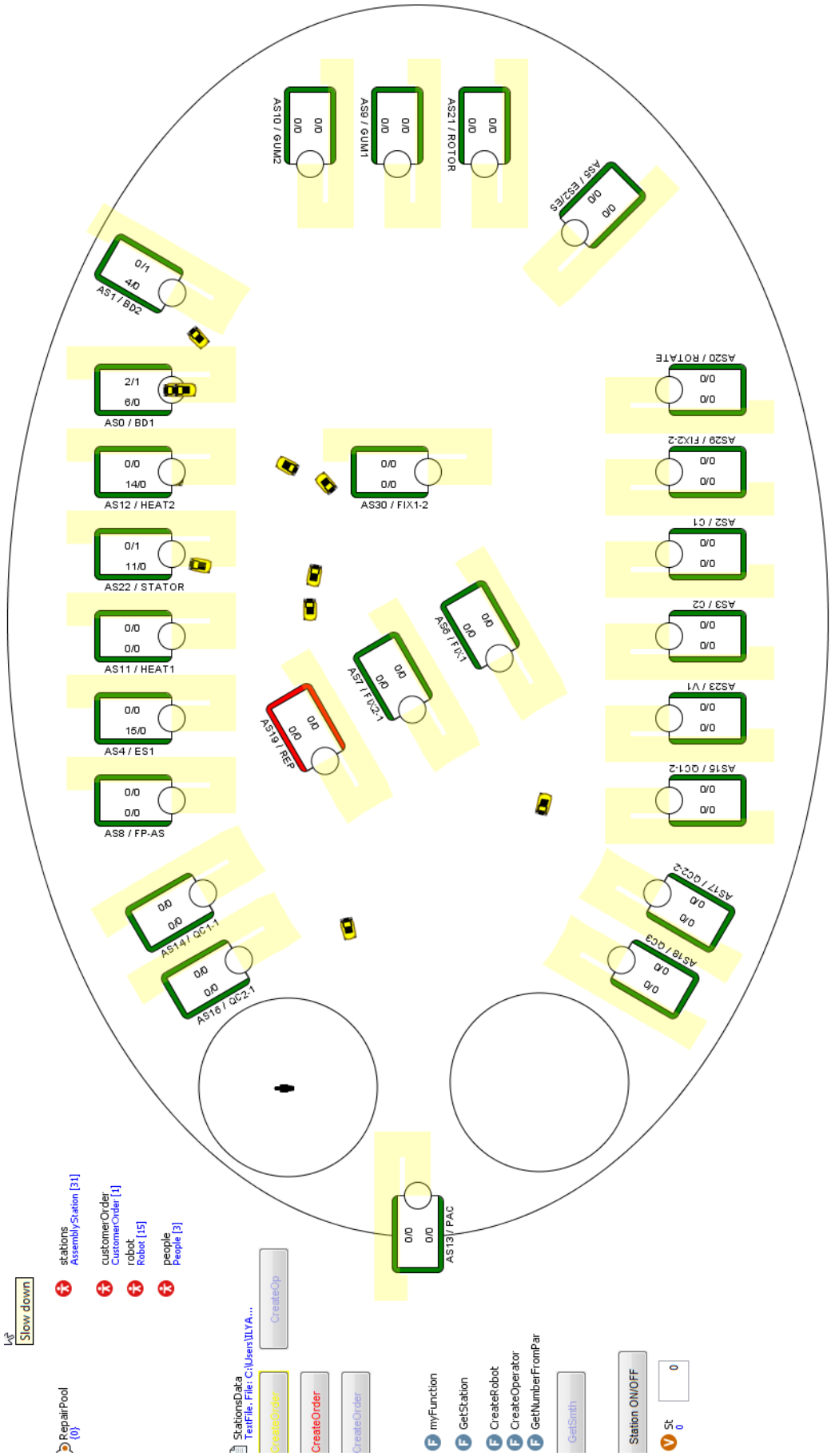


Fig. 8.1. Anylogic experiment view of BAS

This figure presents AnyLogic main experiment window. Active stations are marked with green, and inactive in red. Only inactive station is repair station. Scenario includes five independent runs. Average station utilization for number of runs is calculated according to (8.1).

$$Ust_i = \left(\sum_{n=1}^{N} \left(\frac{\mathrm{Tw}_\mathrm{n}^\mathrm{i}}{\mathrm{Trun}_\mathrm{n}}\right)\right)/N, \quad (8.1)$$

where

i - is number of station,
N - Total number of runs
n - number of run.
Trun_n - time needed to complete n^{th} run,
$\mathrm{Tw}_\mathrm{n}^\mathrm{i}$- is i^{th} station working time within n^{th} run.

Average queue length is calculated according to (8.2).

$$Qst_i = \left(\sum_{n=1}^{N} \left(\mathrm{Q}_\mathrm{n}^\mathrm{i}\right)\right)/N, \quad (8.2)$$

where

i - is number of station,
N - Total number of runs.
n - number of run.
$\mathrm{Q}_\mathrm{n}^\mathrm{i}$ - is a mean of the number of robots in front of i^{th} station within n^{th} run.

The experiment is organized in three steps.

Step 1. This step focuses on initial system analysis. This includes analysis of the 3 product runs:

- P1 - product run of 3500 double-sided motors
- P2 - product run of 3500 one-sided motors
- P1+P2 - product run of 3500 double-sided motors and 3500 one-sided motors at the same time.

Fig. 8.2 shows the diagram for average station utilization (a). Maximal utilization in ideal scenario could achieve 100%. Average station queue length are shown in Fig. 8.2 (b).

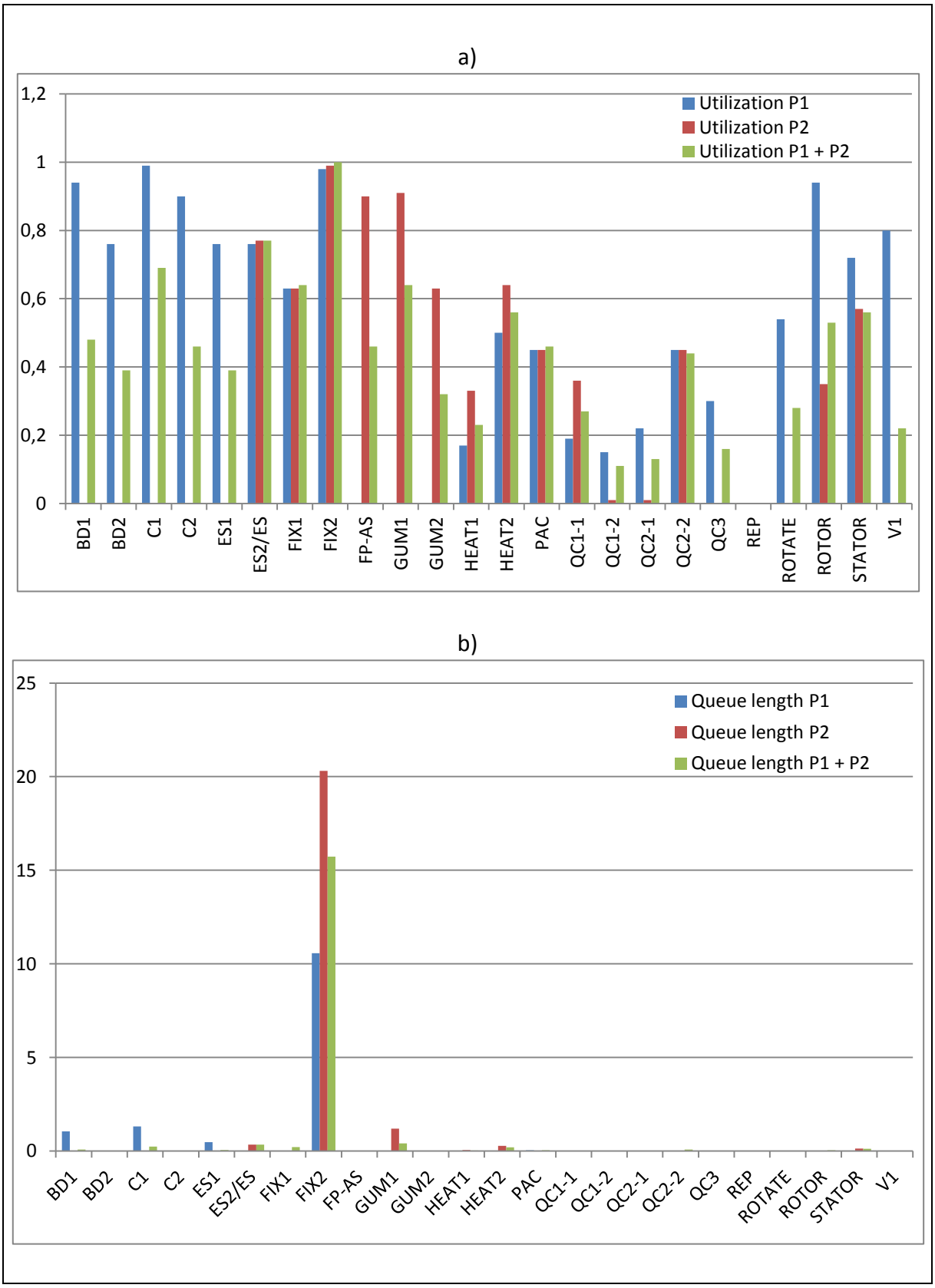


Fig. 8. 2. Step 1. Diagrams for average station utilization (a) and average station queue length (b)

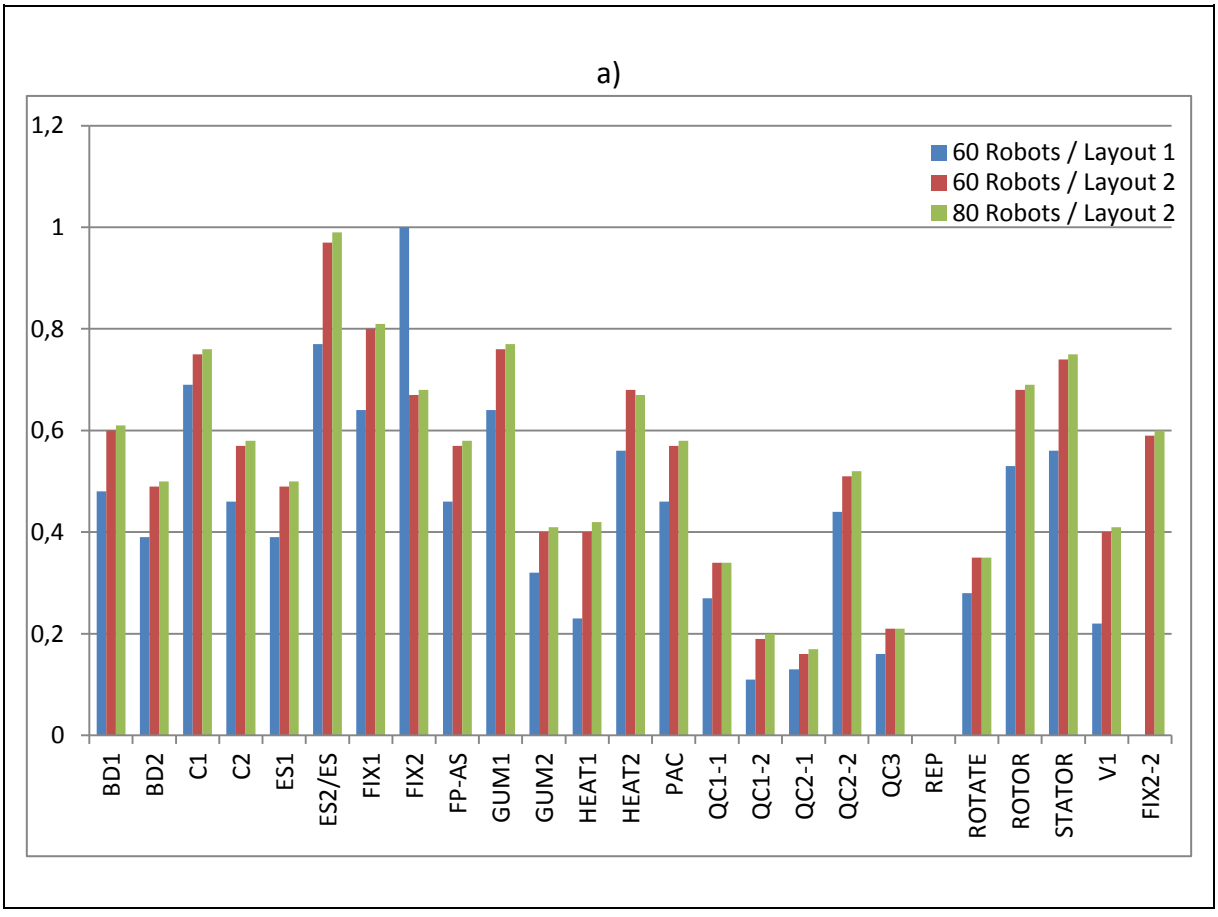
The value of the queue length is restricted only by number of active robots in the system. Initial number of robots is set to 60. It means that queue length cannot exceed 60 robots.

At the single runs average utilization of used stations is about 60%. This figure includes only stations included into assembly process. But average system utilization for the third case is lower than 50%.

Reason of this difference is bottleneck at station FIX2, which is responsible for fixture of motor and electrical components. Average queue length for this station is 15 robots. Therefore this station must be backed up.

Step 2. To improve this situation an additional station for fixture of motor and electrical components FIX2-2 is added. This station is stored as reserve station in supplementary subsystem, so it is easy to include it into the layout. The results of this optimization are shown at Fig. 8.3. Average system utilization increased from 44 to 53%, and the run production time decreased on 20%. That means that for this number of robots system became balanced, because average queue length became less than 1.

To increase the system utilization additional 20 robots were introduced into the system. Injection of additional 20 robots shows another bottleneck, station ES-2/ES This station is overloaded due to the work for both product families. Utilization increase is less than 2% per station.



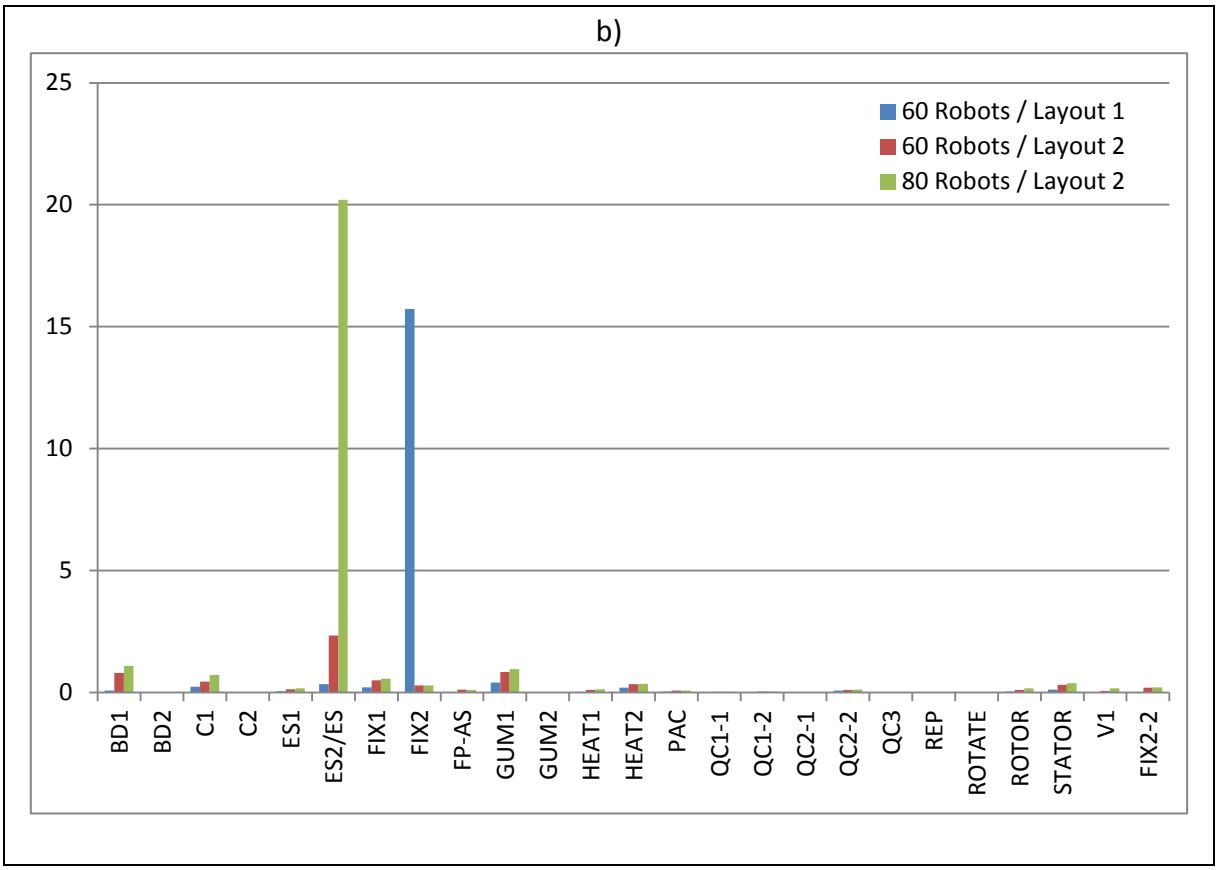


Fig. 8. 3. Step 2. Diagrams for average station utilization (a) and average station queue length (b) with additional station FIX2

Step 3. An optimization possibility is to transform the station ES1 into the ES1/ES. This would mean that it will be opened for an end-shield assembly for the one-sided electrical motor family. The results of this optimization are shown at Fig. 8.4. Trials of layout 3 are made with 60 and 80 robots. At first case results are similar to the layout 2, but the utilization of end shield stations becomes balanced. With use of 80 robots average system utilization increased from 53% to 64%, and the run production time decreased on 15%.

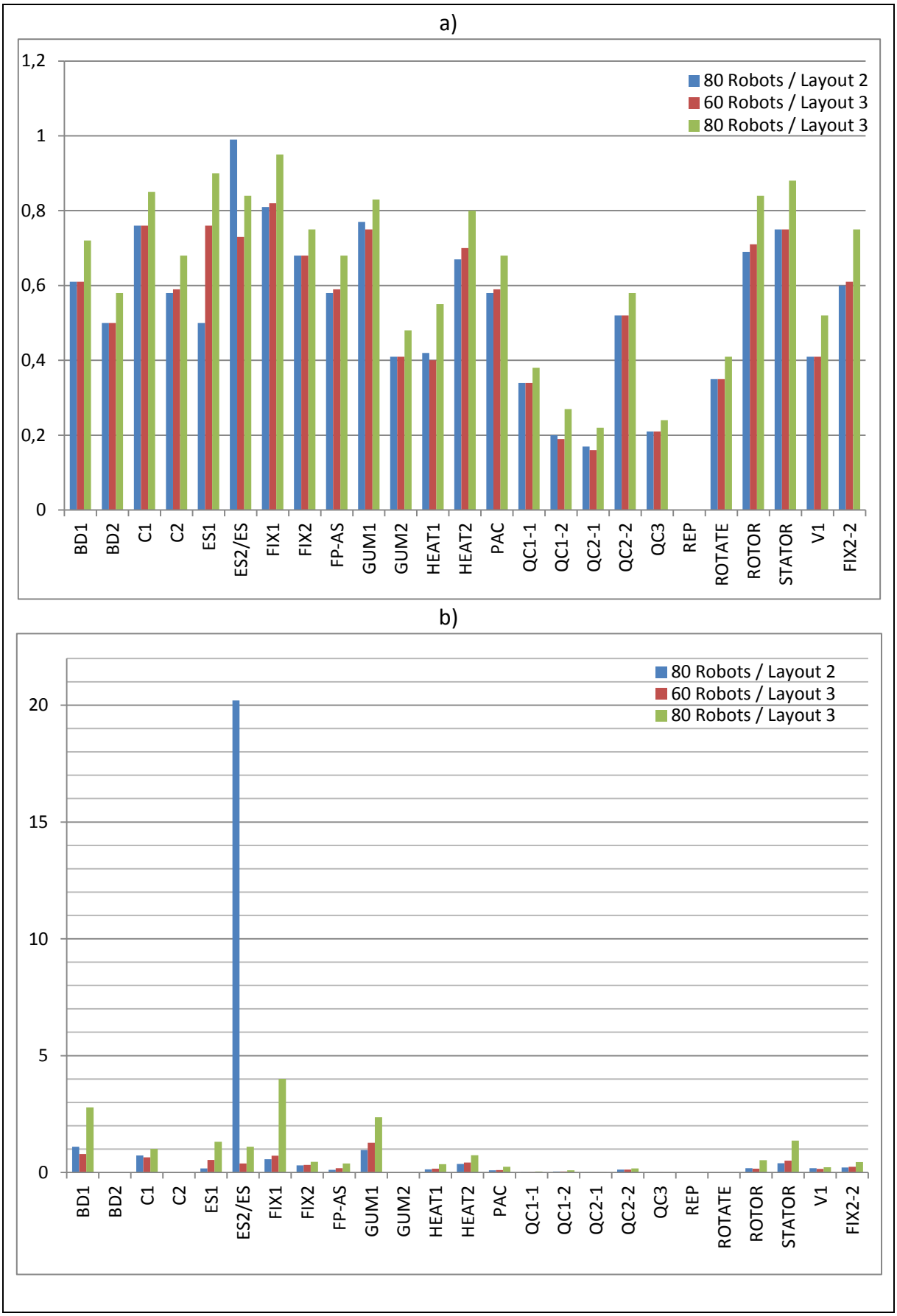


Fig. 8. 4. Step 3. Diagrams for average station utilization (a) and average station queue length (b) with additional station FIX2 and reorganized station ES-1/ES

Results. Developed system has a higher flexibility, then systems of previous generation. This means that there are a lot of factors influencing on the system behavior. Experiment shows following results:

1. **Diagnostics.** Developed simulation tool shows its suitability for deep system analysis as well as prediction of system bottlenecks. System diagnostics for particular experiment gives following output:

   - System initial layout with minimal number of stations has a number of bottlenecks.
   - An optimal number of mobile robots is dynamic and strongly depends on the system configuration
   - Stations variation. The number and type of assembly stations is variable. Optimal number of assembly stations depends on the run mix as well as products technical data.

2. **Internal reserves and reaction of the system.** System has internal reserves to increase the utilization and decrease the run time. Experiment shows that introduction of additional stations, as well as internal reconfiguration of BAS gives the possibility to increase the system utilization.

8.2 Robot behavior analysis

This experiment is focused on analysis of mobile robot behavior at normal working scenario and scenario with disturbances. For this reason number of simulations is made to analyze robot behavior. All history of mobile robot actions is stored in log data file at BAS cloud. Therefore it is easy to export and analyze the data after working scenario. There are possibilities to choose optimal assembly operation for an assembly step and optimal station for this operation.

Step.1. Analysis of alternative stations.
Mobile robot could make an assembly of one product using different trajectories. Therefore to track its behavior there is an analysis of its trajectory given. The first analysis shows how many times robot chooses one of alternative stations. Example of this an analyze of robot R15 behavior, shown at Fig.8.5.

The quality control stations QC1-1 and QC1-2 are alternative for operation QC1 for double-sided motor. Experiment shows that QC1 operation was made almost equal number of times on both alternative stations by this particular robot within one product run.

The quality control stations QC2-1 and QC2-2 are alternative for operation QC2 for double-sided motor. However operation QC2 was made in 80% of cases at station QC2-2.

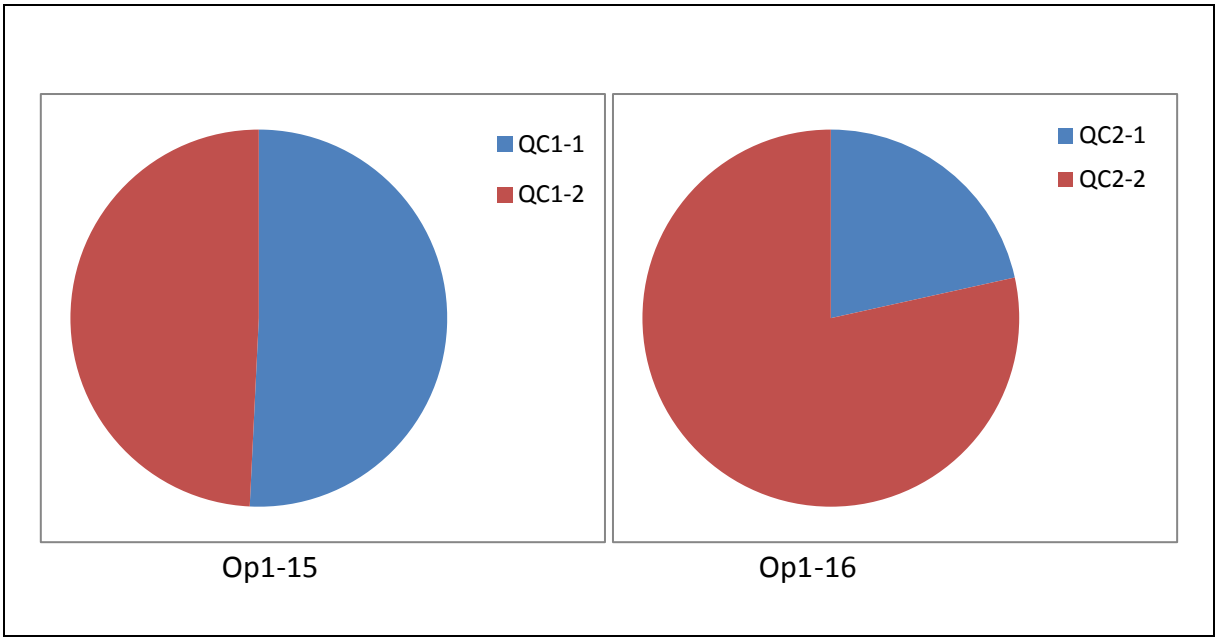


Fig. 8.5. Analysis of alternative stations usage.

Step.2. Analysis of alternative operations.

For each step robot is free to choose an assembly operation. The trajectory used by more than 70% of robots for double-sided motor assembly is shown at Tab 8.1. Most of the robots choose first the alternative operation of fan assembly than assemble a capacitor.

Tab. 8.1. Assembly sequence for double-sided electrical motor family

Step 1	Op1-1	POOL-P
Step 2	Op1-2	ES-1
Step 3	Op1-3	HEAT
Step 4	Op1-4	STATOR
Step 5	Op1-5	BD1
Step 6	Op1-6	BD2
Step 7	Op1-7	ROTOR
Step 8	Op1-8	2ES / ES
Step 9	Op1-9	ROTATE
Step 10	Op1-10	FIX-1
Step 11	Op1-11	FIX-2
Step 12	Op1-14	V-1
Step 13	Op1-12	C-1
Step 14	Op1-13	C-2
Step 15	Op1-15	QC1-1 / QC1-2
Step 16	Op1-17	QC-3
Step 17	Op1-16	QC2-1 / QC2-2
Step 18	Op1-18	PAK

The trajectory used by more than 65% of robots for double-sided motor assembly is shown at Tab 8.2. The heating operation and stator assembly are preferred to be completed earlier than assembling of rubber buffer.

Tab. 8.2. Assembly sequence for one-sided electrical motor family

Step 1	Op2-1	POOL-P
Step 2	Op2-2	2ES/ES
Step 3	Op2-3	ROTOR
Step 4	Op2-6	HEAT
Step 5	Op2-7	STATOR
Step 6	Op2-4	GUM1
Step 7	Op2-5	GUM2
Step 8	Op2-8	FIX-1
Step 9	Op2-9	FIX-2
Step 10	Op2-10	FP-AS
Step 11	Op2-11	QC1-1 / QC1-2
Step 12	Op2-12	QC2-1 / QC2-2
Step 13	Op2-13	PAK

Disturbances analysis. An overall assembly time for one product within one run with disturbances has 250% tolerance. That means that the quickest robots could finish their task 2,5 times faster, then the slowest ones. This could be explained by different path and trajectories planning, as well as station setup breaks. The 60% of robots are faster than average assembly time within a run.

Results. Results of this experiments show that model reflects robot behavior, described in previous chapters. Robots choose an optimal station and operations for each assembly steps and form their unique dynamic assembly sequences.

1. **Prognosis:** An experiment shows that the system has the possibility to develop alternative production scenarios, based on the actual state of the system as well as on the algorithm for the real-time sequence planning, for a period of time in advance.
2. **Optimization of production process:** Modeling of robot scenarios and working with its log data opens wide possibilities for layout optimizations. This data of BAS's work could be used for the human operator support for the BAS most suitable working scenario allocation.
3. **Interface between BAS and the environment:** BAS cloud structure organizes the internal communication and opens wide possibilities to work with log data. The information and material flow between BAS and its environment is organized and each transaction can be reconstructed.

8.3. System performance comparisons

This experiment shows BAS performances for assembly of different run mixes. At this experiment system is tested at three different run mixes:

- 100% Double-sided motor family
- 100% One-sided motor family
- 50/50 mix from double-sided and one-sided motor families

Each run was simulated five times. A comparison is based on average run time. The results are shown at Fig. 8.6.

The initial runs include 2000 pieces of electrical motors of each type. The given run time in single mode sums up two single modes times, and exceeds 1400 minutes. However the mixed mode time is about 850 minutes for a run. That means that system is able to cover the target volume of 4000 motors daily.

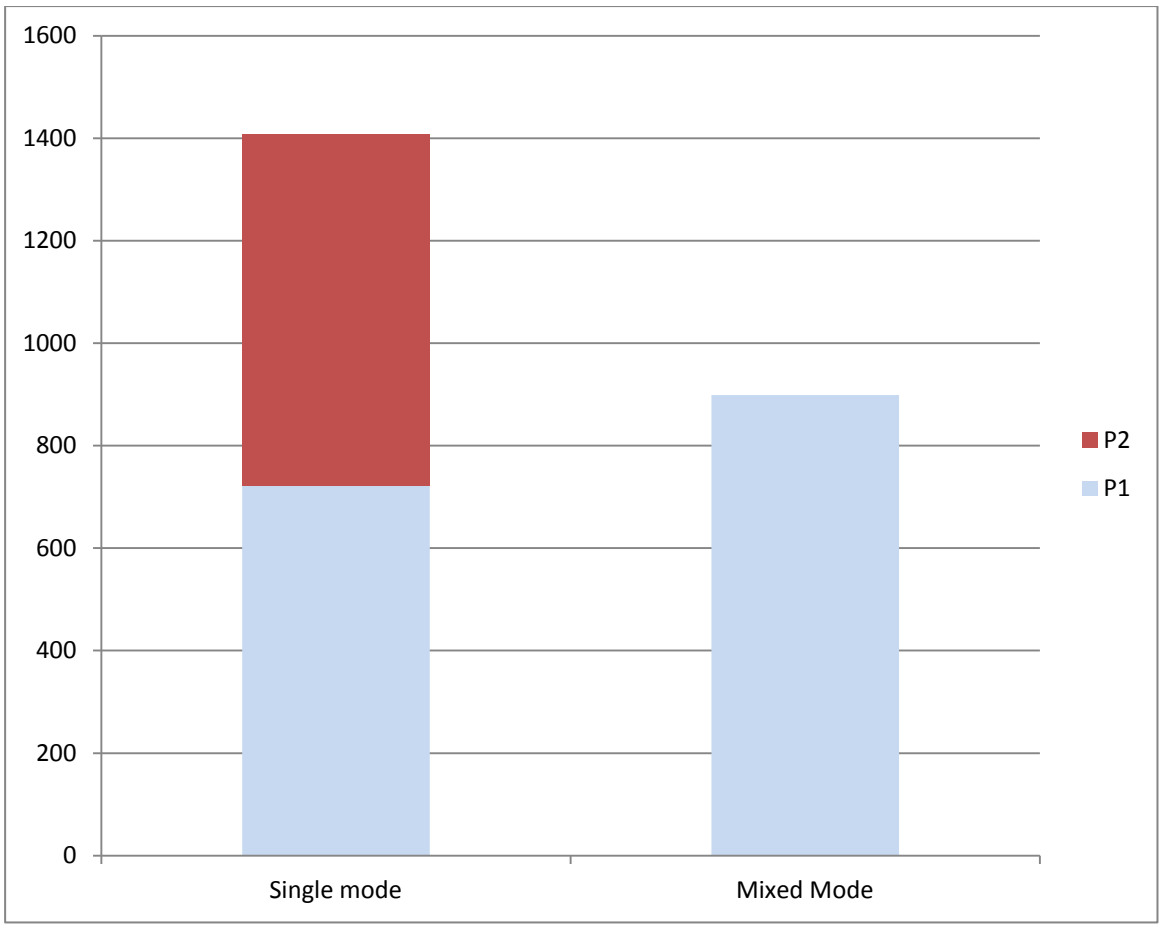


Fig. 8.6. Comparison of run assembly time for single and mixed assembly modes

For single run performances system is comparable with traditional FAS. However the additional transport and docking increases an assembly run time.

Real advantage of BAS could be observed in a mixed mode. Due to the higher machines utilization and sequence planning flexibility system decreasing its run time using its internal reserves.

Results. This case experiment shows following results.

- Designed system is not competitive in a single run mode, due to the similar results but much higher reorganization costs
- It is competitive in a mixed mode
- The run assembly time strongly depends on the product mix
- System could decrease the run time using the internal reserves
- The target system daily productivity is achieved in a mixed mode of assembly
- Simulation is suitable for research and assessment of the best run mix

8.4. Comparison of FAS and BAS

Mode of assembly

FAS has the trend of specialization on the family of product type. That means that it is effective only with limited aims, complexity, and scope. Investigation shows that FAS are more effective in assembly of one family of electrical motors.

BAS is more universal and could be used for more product families. Investigation shows that BAS is more effective in mix mode of two electrical motor families.

Control system

FAS control structure is based on sub-ordination. This approach uses complex software with the intelligence at the highest level. This structure allows the system respond quickly on the customer demand. The main disadvantage of this concept is its lack of flexibility for the disturbances and inability to solve the assembly problems on the lower system level.

BAS has a hybrid control structure. This approach allows to use more simple software, which distributes system intelligence. This structure allows to combine the top-down subsystem on upper level and self-organizing subsystem on shop floor level. Main problem of this concept is a conflict between these subsystems. Introduced BAS cloud concept solves the problem of communication between the subsystems. This leads to the increase of control system performance and general robustness.

Workers Integration

FAS has no possibility to include workers in assembly process. Inclusion of workers will lead to the system imbalance and would require additional reconfiguration, replaning and balancing of FAS.

BAS gives the possibility to include worker as alternative manual assembly station. Inclusion of additional workers increases system efficiency step-by-step without big additional investments.

Reconfiguration

FAS has a limited possibility of layout reconfiguration. Any additional element has to be integrated with the whole system replanning. Fall out of one component makes the system unable to continue system work.

BAS allows dynamic reconfigurations with possibilities to introduce additional elements without additional replanning. BAS layout variability opens a possibility of constant increase of system capabilities. In case of one component fall out, BAS has an ability to reconfigure and continue the system work.

8.4. Summary

A number of experiments were organized to verify BAS behavior under the industrial conditions. Developed system has a higher flexibility, then systems of previous generation. This means that there a lot of factors influencing on the system behavior. Experiments show a number of system and tool characteristics:

BAS characteristics

1. The experiments show that BAS answers the planned characteristics of prognosis, internal reserves, diagnostics and communication organization
2. BAS initial layout with minimal number of stations has a number of bottlenecks, which are diagnosed with simulation tool
3. An optimal number of mobile robots is dynamic and strongly depends on the system configuration
4. System has internal reserves to increase the utilization and decrease the run time
5. Robots choose an optimal station and operations for each assembly steps and form their unique dynamic assembly sequences.
6. BAS is not competitive in a single run mode, due to the similar results but much higher reorganization costs

7. BAS is competitive in a mixed mode due to its flexibility and internal reserves
8. The run assembly time strongly depends on the product mix
9. System could decrease the run time using the internal reserves
10. The target BAS daily productivity of 4000 motors is achieved in a mixed mode of assembly

Simulation tool characteristics

- Simulation is suitable for research and assessment of the best run mix
- Simulation tool is suitable for deep system analysis as well as prediction and of bottlenecks
- Model reflects robot behavior, described in previous chapters
- Modeling of robot scenarios and working with its log data opens wide possibilities for layout optimizations

Conclusion

This dissertation describes the results of research and development of working scenarios and algorithms for self-organizing assembly systems. The main goal is investigating the suitability of self-organizing systems for the assembly of technical products with a electrical motor complexity level.

1. The result of this investigation shows that studied concept of self-organizing Bionic Assembly System is suitable for assembly of the multiple number of product families with a complexity level of electrical motor. The verification of this concept is organized as simulation of working scenarios and strategies for assembly of two families of electrical motors.
2. The cloud concept has been implemented as informational interface between sub-ordinated and self-organizing subsystems of BAS control structure. The implementation of cloud concept brings following advantages:
   - direct and simple horizontal communication on BAS Shop-Floor
   - connection of sub-ordinated and self-organized subsystems
   - storage of system log data
   - storage of system technical data

   The disadvantages of BAS cloud application are:
   - Disconnection of central control system from the shop-floor elements
   - Need in additional cloud backup system
   - Need in additional data security algorithms
3. Assembly robots have to complete an assembly order in shortest possible time. Time is influenced by the robot ability to choose the alternative stations for one operation and alternative assembly operations for one assembly step. The mobile robot algorithm for the selection of the most suitable assembly operation-station combination, based on the shortest time is developed. The algorithm allows to find out sub-optimal trajectory for each robot, considering system states and conditions. Simulation shows that implementation of this algorithm allows to decrease assembly run time.

4. Simulation-based investigation of working scenarios and strategies of developed system gives the possibilities to:
    - determine the optimal range of mobile robots in the system
    - analyze system bottlenecks and develop the strategy for their elimination.
    - find the most suitable system working scenarios for given product mix
    - support the human operators in decision-making process
    - determine the product family/type mix ratio, based on the highest BAS utilization.

Future research will be focused on the further analysis and implementation of self-organizing systems to the industrial applications. This includes the further development of BAS cloud communication and its connectivity with top-down control subsystem. Other part of further research will be focused on the experiments and further implementation of BAS concept for plant optimizations.

References

* (2013) International Comparisons of Hourly Compensation Costs in Manufacturing Industries, xls file, *Available online at* http://www.bls.gov/fls/ichccin dustryreport.htm

** (2013) "Process." *Merriam-Webster.com*. Merriam-Webster, n.d. Web. 4 Aug. 2013. <http://www.merriam-webster.com/dictionary/process>

*** (2013) "Process." *Merriam-Webster.com*. Merriam-Webster, n.d. Web. 4 Aug. 2013. <http://www.merriam-webster.com/dictionary/organization>

**** (2014) Electric Motor & Machines Terminology, NIDEC Motor corporation, *Available online at* <http://www.usmotors.com/TechDocs/ProFacts/Motor-Machinery-Terminology. aspx>

AB (2014) Alen-Breadly Glossary of Motor Terms. *Available online at* <http://www.ab.com/support/abdrives/documentation/fb/1022.pdf>

Anylogic (2014a) Anylogic website, *Available online at* <http://www.anylogic.com/>

Anylogic (2014b) Using AnyLogic Help System *Available online at* <http://www.any logic.com/anylogic/help/>

Asadi, N., Fundin, A. and Jackson, M. (2013). Exploring Optimal Flexible Assembly Systems, *22nd International Conference on Production Research, In: ICPR22*. pp.1-7

Ashby, W. R. (1962). Principles of the self-organizing system. *Principles of Self-organization*, 255-278.

Becker, T.; Graf, G. (2004) Changeability in Operations: A Critical Strategic Resource for European Manufacturing? *Proceedings of the 2nd International Conference "An Enterprise Odyssee: Building Competitive Advantage"*, Galetic L (Ed.), Zagreb/Croatia: pp. 904-917

Bi, Z. M., Lang, S. Y. T., Shen, W., & Wang, L. (2008). Reconfigurable manufacturing systems: the state of the art. *International Journal of Production Research*, 46(4), 967-992

Blau, I., Avner C.(2009) "What type of collaboration helps? Psychological ownership, perceived learning and outcome quality of collaboration using Google Docs" In *Proceedings of the CHAIS conference on instructional technologies research*, pp. 48-55

Briggs, J., Peat, F. D. (2006) Die Entdeckung des Chaos - Eine Reise durch die Chaos-Theorie. Ungekürzte Ausgabe März 1993, 9. Aufl., dtv, ISBN-13: 978-3-423-33047-3, München

Borshchev, A., & Filippov, A. (2004, July). From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools. In*Proceedings of the 22nd international conference of the system dynamics society* (No. 22).

Borshchev, A. (2014). Multi-method modelling: AnyLogic. *Discrete-Event Simulation and System Dynamics for Management Decision Making*, 248-279.

Camazine, S. (2003). *Self-organization in biological systems*. Princeton University Press.

Cochran, D. S.; Arinez, J. F.; Duda, J. W.; Linck, J. (2002) A decomposition approach for manufacturing system design. Journal of Manufacturing Systems, Vol. 20, No. 6, pp. 371-389

Cochran, David S., et al. "A decomposition approach for manufacturing system design." *Journal of Manufacturing Systems* 20.6 (2002): 371-389

Davenport, T H. (2013) Process innovation: reengineering work through information technology. Harvard Business Press

De Toni, A.; Tonchia, S. (1998) Manufacturing flexibility: a literature review. *Journal of Production Research,* 36, pp. 1587-1617

Eigen, M. (1978). The hypercycle: A principle of natural self-organization. *International Journal of Quantum Chemistry*, 14(S5), 219-219

Gastermann, B., Stopper, M., Luftensteiner, F., & Katalinic, B. (2014). Implementation of a Software Prototype with ConWIP Characteristics for Production Planning and Stock Management. *Procedia Engineering,* Volume 69, pp 423-432, ISSN 1877-7058, DOI: 10.1016/j.proeng.2014.03.008

Groover, M. P. (2001) Automation, Production Systems and Computer Integrated Manufacturing, Prentice Hall, ISBN 0-13-089546-6, New Jersey

Groover, M. P. (2007). *Fundamentals of modern manufacturing: materials processes, and systems*. John Wiley & Sons

Haken, H. (2006) Information and self-organization: A macroscopic approach to complex systems. Springer

Jiang, L., Dou, W., & Pan, Y. Q. (2013). Manufacturing Industries Transfer Trend in China 1999-2010. *Advanced Materials Research,* 655, 2284-2287

Katalinic, B., (1990). Industrieroboter und flexible Fertigungsysteme für Drehteile, VDI Verlag, ISBN 3-18-40-1027-9, Duesseldorf 2.

Katalinic, B. (1997) Design Methodology of Scheduling Strategies and Scenarios of Complex Flexible Systems, *Proc. 29th CIRP International Seminar on Manufacturing Systems (Ed. K. Iwata & K. Ueda),* pp 179-185, Osaka University, May 11-13, 1997, Osaka, Japan

Katalinic, B.; Visekruna, V. & Kordic, V. (2002). Bionic Assembly Systems: Design and Scheduling of Next Generation of Self-organising Complex Flexible Assembly System in CIM environment, *Proceedings of The 35th CIRP-International Seminar on Manufacturing Systems*, 12-15 May 2002, Seoul, Korea

Katalinic, B., Kukushkin, I., & Haskovic, D. (2014a) Bionic Assembly System Cloud: Functions, Information Flow And Behavior. *Proceedings of the 9th International Conference of DAAAM Baltic, Industrial Engineering*, 24-26 April 2014, Tallinn, Estonia , pp 103 – 108, ed. T. Otto, Tallinn, Estonia

Katalinic, B., Kukushkin, I., Pryanichnikov, V., & Haskovic, D. (2014b). Cloud Communication Concept for Bionic Assembly System, *Procedia Engineering, Volume 69, 2014,* Pages 1562-1568, ISSN 1877-7058, doi:10.1016/j.proeng.2014.03.156

Katalinic, B., Pryanichnikov, V., Ueda, K., Torims, T., Kukushkin, I., Cesarec, P., Kettler, R. (2013) Control Structure and Scheduling of a Hybrid Assembly System. *Estonian Journal of Engineering,* 2013, Vol.19, Iss.1, pp.18-29. ISSN 1736-7522. doi:10.3176/eng.2013.1.03.

Katalinic, B.; Pryanichnikov, V.E.; Ueda, K.; Kukushkin, I.; Cesarec, P.; Kettler, R. (2012) "Bionic assembly system: hybrid control structure, working scenario and scheduling," *in Proceedings of 9th National Congress on Theoretical & Applied Mechanics*, Brussels:, pp. 111-118

Koren, Y.; Heisel, U.; Jovane, F.; Moriwaki, T.; Pritschow, G.; Ulsoy, G.; Van Brussel, H. (1999) Reconfigurable Manufacturing Systems. Annals of the CIRP, Vol. 48, No. 2, pp. 527-540

Kukushkin , I. K.; Katalinic , B.; Cesarec , P. and Kettler , R. (2011) Reconfiguration in self-organizing systems. *Annals of DAAAM for 2011 & Proceedings of the 22nd International DAAAM Symposium "Intelligent Manufacturing & Automation: Power of Knowledge and Creativity"*, Editor B.ranko. Katalinic, ISSN 1726-9679, ISBN 978-3-901509-83-4, pp 641–642, Vienna, Austria, Published by DAAAM International, Vienna, 2011.

Kukushkin, I.K.; Katalinic, B.; Cesarec, P.; Zdyb, D. & Kettler, R. (2012) Modeling of mobile robot behavior for line-less Bionic Assembly System, *Annals of DAAAM for 2012 & Proceedings of the 23rd International DAAAM Symposium,* ISBN 978-3-901509-91-9, ISSN 2304-1382 pp 0865-0870, Editor B[ranko] Katalinic, Published by DAAAM International, Vienna, Austria, 2012

Lazinica, A., & Katalinic, B. (2007). Bionic assembly system: new concept of self-organising multirobot system. *International Journal of Automation and Control*, (1), 16-27

Lee, T.; Jeziorek, P. N. (2006) Understanding the value of eliminating an off-diagonal term in a design matrix. *Proceedings of 4th International Conference on Axiomatic Design, ICAD06,* Florence/Italy, June 2006

Lotter, B., & Wiendahl, H. P. (2006). Montage in der industriellen Produktion. Springer-Verlag Berlin Heidelberg.

Lotter, B.; Hartel, M.; Menges, R. (1998) Manuelle Montage wirtschaftlich gestalten, Expert Verlag, Renningen-Malmsheim/Germany

Matt, D. T. (2006). Fast Configuration of Lean and Customer Driven Production Systems with Axiomatic Designed Production Module Templates. *Proceedings of CIRP-ICME06 – International Conference on Intelligent Computation in Manufacturing Engineering*, pp. 373-378, Ischia/Italy, July 2006

Matt, D. T. (2008). Template based Design of Lean Production Systems. *Journal of Manufacturing Technology Management,* Vol. 19, No. 7, pp. 783-797, doi:10.1108/17410380810898741

Mehrabi, M. G.; Ulsoy, A. G.; Koren, Y. (2000) Reconfigurable manufacturing systems: Key to future manufacturing. *Journal of Intelligent Manufacturing.* Kluwer Academic Publishers, 11, pp. 403-419

Monostori, L., Váncza, J., & Kumara, S. R. (2006). Agent-based systems for manufacturing. *CIRP Annals-Manufacturing Technology*, *55*(2), 697-720

Nanasi, J. (1996) Konzept eines intelligenten Moduls für die Optimierung der Abläufe innerhalb des FFS, *Dissertation,* Technische Universität Wien, Wien

Nyhuis, P.; Kolakowski, M.; Heger, C. L. (2005) Evaluation of Factory Transformability *Proceedings of CIRP 3rd International Conference on Reconfigurable Manufacturing,* Ann Arbor/MI/USA

O´Connor, J.; McDermott, I. (1998) Die Lösung lauert überall - Systemisches Denken verstehen und nutzen. VAK Verlags GmbH, ISBN 3-932098-29-3, Kirchzarten bei Freiburg

Ostgathe, M., & Zaeh, M. F. (2013). System for product-based control of production processes. *In Computational Intelligence In Production And Logistics Systems (CIPLS), IEEE Workshop*, pp. 138-144, IEEE

Panfilov,P.; Salibekyan, S. (2014) Dataflow Computing and its Impact on Automation Applications, Procedia Engineering, Volume 69, 2014, Pages 1286-1295, ISSN 1877-7058, http://dx.doi.org/10.1016/j.proeng.2014.03.121

Pryanichnikov, V.; Katalinic, B.; Platonov, A. (2011) Application of the autonomous mobile robots «AMUR» for the modelling of the self-organizing systems, (in Russian), UDK 681.518.3*, Intellectual and Adaptive Robots*, Vol. 6, No 1-2, 2011, pp 8-18

Rosati, G., Faccio, M., Carli, A., & Rossi, A. (2011). Convenience analysis and validation of a fully flexible assembly system. In *Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on*(pp. 1-8). IEEE

Shanthikumar, J. George, and Kathryn E. Stecke. (1986) "Reducing work-in-process inventory in certain classes of flexible manufacturing systems." European Journal of Operational Research 26.2: 266-271

Spath, D.; Scholz, O. (2007) Mutability for a Profitable Assembly in Germany (in German). Industrie Management, Vol. 23, No. 2, pp. 61-64

Suarez, F.; Cusumano, M.; Fine, C. (1991) Flexibility and Performance: A Literature Critique and Strategic Framework. Sloan School WP 3298-91-BPS, MIT/Cambridge/MA

Suh, N. P. (2006) Application of Axiomatic Design to Engineering Collaboration and Negotiation. Proceedings of 4th International Conference on Axiomatic Design, ICAD06, Florence/Italy, June 2006

Sun, Y., Lambert, D., Uchida, M., & Remy, N. (2014). Collaboration in the cloud at Google. In *Proceedings of the 2014 ACM conference on Web science*(pp. 239-240). ACM.

Ueda, K. (2014). Emergent Synthesis. CIRP Encyclopedia of Production Engineering, 458-461.

Ulrich, H.; Probst, G. (1995) Anleitung zum ganzheitlichen Denken und Handeln – Ein Brevier für Führungskräfte, 4. Aufl., Paul Haupt Verlag, ISBN : 978-3-258-05182-6, Bern

Vester, F. (1999) Die Kunst, vernetzt zu denken. Ideen und Werkzeuge für einen neuen Umgang mit Komplexität. Deutsche Verlags-Anstalt (DVA), ISBN-10 3421053081, Stuttgart

Whitney, D. E. (2004). Mechanical Assemblies: Their Design, Manufacture, and Role in Product Development. New York, NY: Oxford University Press, 2004. ISBN: 0195157826

Wiendahl, H.-P., Heger, C. L. (2003) Justifying Changeability - A Methodical Approach to Achieving Cost Effectiveness. *Proceedings of CIRP 2nd International Conference on Reconfigurable Manufacturing*, Ann Arbor/MI/USA

Zaeh, M. F.; Moeller, N.; Vogl, W. (2005) Symbiosis of Changeable and Virtual Production – The Emperor's New Clothes or Key Factor for Future Success? *Proceedings of CARV 05 International Conference on Changeable, Agile, Reconfigurable and Virtual Production. Garching/Germany,* September 2005

Appendix

A. Anylogic listing

This listing presents elements and functions of agent class Robot. This includes robot parameters, functions, events, collection, statechart elements (transitions, states, branches), statistics elements and excel files connectivity elements

Full program listing on 128 pages is available via permanent link http://goo.gl/H4rlKg

Agent Type: Robot

Name	Value
General	
Exclude From Build	false
Advanced	
Limit the number of array elements	false
Auto-create Datasets	true
Entity actions	
Flowcharts Usage	ENTITY
Movement parameters	
Velocity	15
Rotate Animation Towards Movement	true
Advanced	
Make Default View Area	true
Movement parameters	
Rotate Animation Vertically	false
Advanced Java	
Generic	false
Advanced	
Recurrence	1

Parameter: RobotX

Name	Value
General	
Show name	true
Array	false

Name	Value
Exclude From Build	false

Type	double
Show At Runtime	true
Advanced	
Modificator	STATIC
Use Units	false
Save In Snapshot	true
Editor	
Editor Control	TEXT_BOX

Parameter: RobotY

Name	Value
General	
Show name	true
Array	false
Exclude From Build	false
Type	double
Show At Runtime	true
Advanced	
Modificator	STATIC
Use Units	false
Save In Snapshot	true
Editor	
Editor Control	TEXT_BOX

Parameter: RobotPriority

Name	Value
General	
Show name	true
Array	false
Exclude From Build	false
Type	String
Show At Runtime	true
Advanced	
Modificator	STATIC
Use Units	false
Save In Snapshot	true
Editor	
Editor Control	TEXT_BOX

Parameter: Num

Name	Value
General	
Show name	true
Array	false
Exclude From Build	false

Name	Value
Type	int
Show At Runtime	true
Advanced	
Modificator	STATIC
Use Units	false
Save In Snapshot	true
Editor	
Editor Control	TEXT_BOX

Parameter: OrderNum

Name	Value
General	
Show name	true
Array	false
Exclude From Build	false
Type	int
Show At Runtime	true
Advanced	
Modificator	STATIC
Use Units	false
Save In Snapshot	true
Editor	
Editor Control	TEXT_BOX

Parameter: RobotProduct

Name	Value
General	
Show name	true
Array	false
Exclude From Build	false
Type	String
Show At Runtime	true
Advanced	
Modificator	STATIC
Use Units	false
Save In Snapshot	true
Editor	
Editor Control	TEXT_BOX

Function: ChooseStations

Name	Value
General	
Show name	true
Exclude From Build	false
ReturnModificator	VOID
Show At Runtime	true

Name	Value
Arguments	
Parameters	()
Advanced	
Access Type	default
Function body	

Body	for (int j=1;j<=ProductCounter;j++)
	{
	for(int b=2;b<28;b++)//50 number of 0s in excel sheet string. could be more or less.
	{
	if (ProductChoose.cellExists(1,j,b)==true)
	{
	
	if(preconditions.contains(ProductChoose.getCellStringValue(1,j,b))==false)
	{
	String p= ProductChoose.getCellStringValue(1,j,b);
	preconditions.add(p);
	}
	}
	}
	//traceln("j= "+j+" ; "+preconditions);
	
	if ((AssemblyMatrix.contains(ProductChoose.getCellStringValue(1,j,1))!=true)&&(AssemblyMatrix.containsAll(preconditions)))
	{

Advanced	
Use Units	false
Static	false

Event: event

Name	Value
General	
Occurence Time	30
Recurrence	0.5
Show name	true
Mode	cyclic
Event Recurrence Time Unit	MODEL_TIME_UNIT
Exclude From Build	false
Occurs At Time	true
Trigger Type	timeout
Show At Runtime	true
Action	
Action	if (get_Main().Clock==Num) RobotStatechart.receiveMessage(Num);

Variable: varX

Name	Value
General	

Show name	true

Name	Value
Exclude From Build	false
Initial Value	400
Type	double
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: varY

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	520
Type	double
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: counter

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	1
Type	int
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: ProductNum

Description: Number of the stations in the Product Matrix

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	1
Type	int
Show At Runtime	true
Advanced	
Access Type	public
Description	
Description	Number of the stations in the Product Matrix
Advanced	
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: ProductCounter

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	1
Type	int
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: ProductChoose

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	Product1
Type	ExcelFile
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: time

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	-10000
Type	double
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: StNum

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	0
Type	int
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: StTimeToFinish

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	0
Type	double
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: CurrentStTimeToFinish

Name	Value
General	

Name	Value
Show name	true
Exclude From Build	false
Initial Value	0
Type	double
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: ChMess

Name	Value
General	
Show name	true
Exclude From Build	false
Initial Value	false
Type	boolean
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: OrderTime

Name	Value
General	
Show name	true
Exclude From Build	false
Type	double
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: PartNum

Name	Value
General	

Show name	true
Exclude From Build	false

Name	Value
Type	String
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Variable: ChooseStationsPartNum

Name	Value
General	
Show name	true
Exclude From Build	false
Type	String
Show At Runtime	true
Advanced	
Access Type	public
Use Units	false
Static	false
Save In Snapshot	true
Constant	false

Collection: preconditions

Name	Value
General	
Collection Class	ArrayList
Show name	true
Exclude From Build	false
Element Class	String
Show At Runtime	true
Advanced	
Access Type	public
Colleciton Initializer Code	{}
Static	false
Save In Snapshot	true

Collection: AssemblyMatrix

Name	Value
General	
Collection Class	ArrayList
Show name	true

Exclude From Build	false
Element Class	String
Show At Runtime	true
Advanced	

Name	Value
Access Type	public
Colleciton Initializer Code	{}
Static	false
Save In Snapshot	true

Collection: Options

Name	Value
General	
Collection Class	ArrayList
Show name	true
Exclude From Build	false
Element Class	String
Show At Runtime	true
Advanced	
Access Type	public
Colleciton Initializer Code	{}
Static	false
Save In Snapshot	true

Statechart Entry Point: RobotStatechart

Name	Value
General	
Show name	true
Exclude From Build	false
Show At Runtime	true

Transition: MovetoPallet

Name	Value
General	
Show name	true

Action	RobotX=getX(); RobotY=getY();
	traceln (RobotX + " ; " + RobotY);
	varX=400;
	varY=520;

Exclude From Build	false
Guard	(get_Main().customerOrder.get(get_Main().GetNumberFromPar(OrderNum.get_Main().customerOr
Trigger Type	timeout
Timeout	1

Transition: transition4

Name	Value
General	
Show name	false
Exclude From Build	false
Trigger Type	arrival

Transition: MoveToNextStation

Name	Value
General	
Show name	true
Exclude From Build	false
Trigger Type	condition
Condition	AssemblyMatrix.contains(Options)==false

Transition: transition6

Name	Value
General	
Show name	false

Action	RobotX=getX(); RobotY=getY();
	traceln("I am giving to the station "+StNum+ " my number " +Num);
	//get_Main().stations.get(StNum).TimeToOut(RobotPriority, Num);
	get_Main().stations.get(StNum).RobotNum=Num;
	get_Main().stations.get(StNum).RobotProd=RobotProduct;
	get_Main().stations.get(StNum).Inject(RobotPriority);

Exclude From Build	false
Trigger Type	arrival

Transition: ChangeMe

Name	Value
General	
Show name	true
Filter Type	equalsTo

Name	Value
Equals	Num
Exclude From Build	false
Trigger Type	message
Message Type	int

Transition: NotChanged

Name	Value
General	
Show name	true
Exclude From Build	false
Trigger Type	condition
Condition	(CurrentStTimeToFinish<=StTimeToFinish==true) || (!get_Main().stations.get(StNum).IsInQueue(RobotPriority,Num))

Transition: transition9

Name	Value
General	
Show name	false
Exclude From Build	false
Trigger Type	condition
Condition	(CurrentStTimeToFinish>StTimeToFinish==true)&&(get_Main().stations.get(StNum).IsInQueue(RobotPriority,Num)==true)

Transition: transition8

Name	Value
General	
Show name	false
Action	ChMess=false;
Exclude From Build	false
Trigger Type	condition
Condition	ChMess

Transition: transition2

Name	Value
General	
Show name	false
Exclude From Build	false
Trigger Type	timeout
Timeout	2

Transition: transition

Name	Value
General	
Show name	false
Action	/*ChooseStations(); //StNum=get_Main().GetStation(ProductChoose.getCellStringValue

Name	Value

	(1,counter,1),RobotPriority,Num);
	CurrentStTimeToFinish=StTimeToFinish;
	varX = get_Main().stations.get(StNum).myX;
	varY = get_Main().stations.get(StNum).myY;
	traceln (RobotX + " ; " + RobotY+ " ; " + counter + " ");
	moveTo(varX, varY, get_Main().polyline);*/

Exclude From Build	false
Default Transition	false
Condition	counter<=ProductCounter == true

Transition: transition3

Name	Value

General	
Show name	false
Exclude From Build	false
Default Transition	false
Condition	counter<ProductCounter == false

Transition: MoveToPackaging

Name	Value
General	
Show name	true
Exclude From Build	false
Trigger Type	arrival

Transition: transition5

Name	Value
General	
Show name	false
Exclude From Build	false
Trigger Type	timeout
Timeout	1

Transition: MoveToPoolOfRobots

Name	Value
General	
Show name	true

Action	int y=get_Main().GetNumberFromPar(OrderNum,get_Main().customerOrder);
	get_Main().customerOrder.get(y).StCustomerOrder.receiveMessage("ThatsAll");
	traceln("Robot: "+Num+" All Done. Moving home");
	
	varX = 395;

Exclude From Build	false
Default Transition	false
Condition	(get_Main().customerOrder.get(get_Main().GetNumberFromPar(OrderNum,get_Main().customerOr

Transition: KillRobot

Name	Value
General	
Show name	false

Action	traceln("I am Robot "+Num+" is killed NOW! " +time());
	get_Main().RobotReport.println(Math.ceil(time()*100)/100+"; Robot "+Num+"; Killed");
	get_Main().RobotReport.close();

Exclude From Build	false
Trigger Type	arrival

Transition: transition1

Name	Value
General	
Show name	false
Action	traceln("Robot: "+Num+" Taking New Order"); counter = 1;
Exclude From Build	false
Default Transition	true

Transition: transition7

Name	Value
General	
Show name	false
Exclude From Build	false
Guard	1<0==true
Trigger Type	condition
Condition	get_Main().PartCount%2500==0==true

Transition: transition10

Name	Value
General	
Show name	false
Action	traceln("Robot: "+Num+" Taking New Order"); //counter = ProductCounter:
Exclude From Build	false
Trigger Type	arrival

Transition: AssemblyOperation

Name	Value
General	
Show name	true
Filter Type	equalsTo

Action	counter++;
	AssemblyMatrix.add(ChooseStationsPartNum);
	get_Main().RobotLog.println(Math.ceil(time()*100)/100+"; Robot "+Num+"; StNum "+StNum+"; "+ChooseStationsPartNum);
	get_Main().RobotLog.close();

Name	Value
Equals	"OpenMe"
Exclude From Build	false
Trigger Type	message
Message Type	String

State: NewOrder

Name	Value

General	
Exit Action	//choose the assembly matrix
Fill Color	skyBlue
Show name	true
Exclude From Build	false
Entry Action	AssemblyMatrix.clear(); int y = get_Main().GetNumberFromPar(OrderNum,get_Main().customerOrder);

State: Pallet

Name	Value
General	
Fill Color	greenYellow
Show name	true
Exclude From Build	false

Entry Action	RobotX=getX(); RobotY=getY();
	if (RobotProduct == "AS1") {ProductChoose=Product1;
	AssemblyMatrix.add("Op1-1");}
	if (RobotProduct == "AS2") {ProductChoose=Product2;
	AssemblyMatrix.add("Op2-1");}
	if (RobotProduct == "AS3") ProductChoose=Product3;
	ProductChoose.readFile();
	ProductCounter=ProductChoose.getLastRowNum(1);
	traceln("Productcounter for "+ProductChoose+" is "+
	ProductCounter);
	/*if (RobotPriority == "normal") ProductCounter=6;
	if (RobotPriority == "high") ProductCounter=5;
	if (RobotPriority == "low") ProductCounter=6;

State: Deside

Name	Value
General	
Exit Action	if (Options.isEmpty()==false) StNum=get_Main().GetStation(ChooseStationsPartNum,RobotPriority Num);
Show name	true
Exclude From Build	false
Entry Action	ChooseStations(); //StNum=get_Main().GetStation(ProductChoose.getCellStringValue(1,counter,1),RobotPriority,Num);

State: Moving

Name	Value
General	
Show name	true
Exclude From Build	false

Entry Action	traceln ("St Num = " + StNum);
	varX = get_Main().stations.get(StNum).myX;
	varY = get_Main().stations.get(StNum).myY;
	traceln (RobotX + " ; " + RobotY+ " ; " + counter + " ");

State: WaitingAtTheStation

Name	Value
General	
Show name	true
Exclude From Build	false

State: Out1

Name	Value
General	
Show name	true
Exclude From Build	false
Entry Action	ChooseStations(); int

State: Changed

Name	Value
General	
Show name	true
Exclude From Build	false
Entry Action	get_Main().stations.get(StNum).Remove(RobotPriority, Num);

State: Out

Name	Value
General	
Show name	true
Exclude From Build	false

State: FinishedOrder

Name	Value
General	
Show name	true
Exclude From Build	false

Entry Action	//I'm at the station.
	//wait in the queue
	//wait for the TimetoFinish

Name	Value

	finised!");
	traceln("Robot: "+Num+" moving to packaging");
	varX = 300;
	varY = 405;
	moveTo(varX, varY);

State: Packaging

Name	Value
General	

Exit Action	int y =
	get_Main().GetNumberFromPar(OrderNum,get_Main().customerOrder);
	get_Main().customerOrder.get(y).NumberMinus();
	traceln ("Robot: Task with " + RobotPriority + " Priority is finished!");
	OrderTime=time()-OrderTime;
	OrderTime=Math.ceil(OrderTime*100)/100;
	get_Main().RobotReport.println(Math.ceil(time()*100)/100+"; Robot "+Num+"; Product "+RobotProduct+"; Priority "+RobotPriority+"; Finished in "+ OrderTime);
	get_Main().RobotReport.close();
	
	//message that piece bla bla in order bla bla is completed

Fill Color	plum
Show name	true
Exclude From Build	false

Entry Action	//I'm at the station.
	//wait in the queue
	//wait for the TimetoFinish

State: AtPoolOfRobots

Name	Value
General	
Fill Color	plum
Show name	true
Exclude From Build	false
Entry Action	//I'm at the station. //void remove_people(Person personToRemove)

State: Repair

Name	Value
General	

Exit Action	get_Main().stations.get(StNum).RobotNum=Num;
	get_Main().stations.get(StNum).RobotProd=RobotProduct;
	get_Main().stations.get(StNum).Inject(RobotPriority);
	time=time();

Show name	true
Exclude From Build	false

Entry Action	StNum=25;
	traceln ("St Num = " + StNum);
	varX = get_Main().stations.get(StNum).myX;
	varY = get_Main().stations.get(StNum).myY;
	traceln (RobotX + " : " + RobotY+ " : " + counter + " ");

Name	Value
	moveTo(varX, varY, get_Main().polyline);

Branch: branch1

Name	Value
General	
Show name	false
Exclude From Build	false

Branch: branch2

Name	Value
General	
Show name	false

Action	/*for (int i=0;i<=get_Main().customerOrder.size();i++)
	{
	if (get_Main().customerOrder.get(i).TakenNumber>=1)
	{
	RobotPriority = get_Main().customerOrder.get(i).OrderPriority;
	OrderNum = get_Main().customerOrder.get(i).Num;
	traceln ("I am robot " + Num + " changed to the order " + OrderNum);
	break;
	}
	}*/
	/*for (int i=0;i<=get_CustomerOrder().get_Main().customerOrder.size();i++)

Exclude From Build	false

Statistics: statistics

Name	Value
General	
Exclude From Build	false
Discrete	true
Show At Runtime	true
Data update	
Analysis Auto Update	true
Recurrence	1

Excel File: Product1

Name	Value
General	
File name	C:/Users/ILYA/Google Drive/DISS_2014_Final/Anylogic/Anylogic Models/BAS22_1_trv/AssemblvMatrixP1_Preconditions.xlsx
Show name	true
Exclude From Build	false
Show At Runtime	true
Advanced	
Save on model termination	true

Load on model startup	true
Save In Snapshot	false

Excel File: Product2

Name	Value
General	
File name	C:/Users/ILYA/Google Drive/DISS_2014_Final/Anylogic/Anylogic Models/BAS22_1_trv/AssemblvMatrixP2_Preconditions.xlsx
Show name	true
Exclude From Build	false
Show At Runtime	true
Advanced	
Save on model termination	true
Load on model startup	true
Save In Snapshot	false

Excel File: Product3

Name	Value
General	
File name	C:/Users/ILYA/Google Drive/DISS_2014_Final/Anylogic/Anylogic Models/BAS22_1_trv/AssemblvMatrixP3.xlsx
Show name	true
Exclude From Build	false
Show At Runtime	true
Advanced	
Save on model termination	true
Load on model startup	true
Save In Snapshot	false

Appendix B.

Kukushkin Ilya, Dipl.-Ing.

Gumpendorferstraße, 39/232
1060 Wien, Österreich
ilia.kukushkin@gmail.com
+43 650 990 6296

Persönliche Daten

Staatsangehörigkeit	Russland
Aufenthaltstitel	Österreich

Ausbildung

09/2011 - 06/2014(voraussichtlich)	**Doktoratsstudium Maschinenbau / Wirtschaftsingenieurwesen** **TU Wien und Boston University/MIT (USA)** - Dissertation über "Anwendung von selbstorganisierenden Systemen für Produktionsunternehmen" betreut von Prof. Dr.sc. Dr.mult. h.c. Prof.h.c. Katalinic, Projekt in Zusammenarbeit mit MAGNA Exteriors & Interiors GmbH - Notendurchschnitt - 1.0 - 1 Semester an der Boston University/MIT, Forschungsgruppe von Prof. Belta - Teilnehmer der 2nd DAAAM International Doctoral School (4 ECTS, Note: 1.0) - Zusätzliche Kurse (TU Wien): Moderne Methoden im Produktionsmanagement / Projektmanagement / Creativity Engineering
09/2006 - 07/2011	**Diplomstudium Ingenieurwesen** **Omsk State Transport University, Russland** - Diplomarbeit "Development of working algorithms for self-organizing production systems" unter der Leitung von Prof. Dr.sc. Dr.mult. h.c. Prof.h.c. Katalinic in Zusammenarbeit mit der TU Wien - 2 Auslandssemester, Frühjahr 2009: TU Wien - Abschluss mit Auszeichnung
06/2006	**Lyzeum "Business und Informationstechnologien", Omsk, Russland** - Matura mit Auszeichnung

Berufserfahrung

09/2011 - laufend	**TU Wien/Österreich** **Wissenschaftlicher Mitarbeiter, Dissertant** - Mitarbeit in der Forschungsgruppe von Prof. Dr.sc. Dr.mult. h.c. Prof.h.c. Katalinic im Rahmen von "Next Generation Production Systems" Initiative in direkter Zusammenarbeit mit MAGNA Exteriors & Interiors GmbH - Simulation und Ablaufoptimierung von flexibel und selbstorganisierenden Systemen - Lehre im Rahmen von "Intelligent Manufacturing Systems" Kurs von Prof. Dr.sc. Dr.mult. h.c. Prof.h.c. Katalinic im WS2012/WS2013 an der TU Wien
03/2013 - 07/2013	**Boston University/MIT (USA)** **Wissenschaftlicher Mitarbeiter** - Mitarbeit in der Forschungsgruppe von Prof. Belta
09/2011 - laufend	**DAAAM International Vienna - Verband der Ingenieure und Forscher in einem Gebiet der Automatisierung und Fertigung / Österreich** **Senior Assistent** - Bearbeitung der wissenschaftlicher Artikeln für "DAAAM International Scientific Book 2011/2012/2013" - Organisation der DAAAM International Kongresse 2011/2012/2013
03/2010 – 06/2011	**AJAX-Agro GmbH, Omsk, Russland** **Vertreter in Westafrika** - Eröffnungsvorbereitung und Geschäftsentwicklung der Filiale in Ghana, Westafrika - Marktforschung und Kundenkontakte in Ghana, Westafrika
09/2008-01/2009	**Samsung Russland GmbH** **Trainer in Samsung Innovation School** - Durchführung des Workshops "Student Mobile Technologies"

Sprachkenntnisse

Russisch:	Muttersprache
Englisch:	Verhandlungssicher
Deutsch:	Fließend
Slawisch:	Sehr gute Kenntnisse
(Bosnisch, Kroatisch, Serbisch)	

EDV-Kenntnisse

MS-Office-Anwenderkenntnisse	Sehr gut
C / C ++	Sehr gut
Java	Gut
Simulationssoftware Kenntnisse	Sehr gut
(Anylogic, Arena, Plant Simulation)	
AutoCAD, MS Visio	Gut
CorelDraw, Adobe Photoshop	Gut

Stipendien und Preise

2013 FESTO Stipendium für 2nd DAAAM International Doctoral School
2013 Marshallplan Stipendium für den Forschungsaufenthalt in der USA
2012 Best Oral Presentation Award at DAAAM International World Symposium on Intelligent Manufacturing and Automation
2011 Erasmus Mundus Multic Programme Stipendium für das Doktoratsstudium in Österreich
2009 Erasmus Mundus Action 2 ECW Programme Stipendium für das Studienaufenthalt in Österreich
2009 Best Samsung Innovation School Trainer

Publikationen

2013 - Kukushkin, I.; Katalinic, B.; Haskovic D.; Pryanichnikov V., Communication in Bionic Assembly System, Procedia Engineering, Elsevier Publishing, Accepted: 2013-10-30
2012 - B. Katalinic, V. Pryanichnikov, K. Ueda, P. Cesarec, I. Kukushkin, R. Kettler: "Bionic Assembly System: working modes, control and scheduling"; International Journal of Industrial Engineering and Management, Vol 3 (2012), 3; 121 - 131.
2012 - Kukushkin, I. K.; Katalinic, B.; Cesarec, P.; Zdyb, D. & Kettler, R.: Modeling of mobile robot behavior for line-less Bionic Assembly System, Annals of DAAAM for 2012 & Proceedings of the 23rd International DAAAM Symposium, ISBN 978-3-901509-91-9, ISSN 2304-1382, pp 0865 - 0870, Editor B[ranko] Katalinic, Published by DAAAM International, Vienna, Austria, 2012
2012 - Katalinic, B., Kukushkin, I. K., Cesarec, P & Kettler, R., Hybrid Control Structure and Scheduling of Bionic Assembly System, Proceedings of the 8th International Conference of DAAAM Baltic, Industrial Engineering, 19-21 April 2012, pp 483 – 489, ed. T. Otto, Tallinn, Estonia, 2012
2011 - Kukushkin, I. K.; Katalinic, B.; Cesarec, P. & Kettler, R.: Reconfiguration in Self-Organizing Systems, Annals of DAAAM for 2011 & Proceedings of the 22nd International DAAAM Symposium, ISBN 978-3-901509-83-4, ISSN 1726-9679, pp 0641-0642, Editor B[ranko] Katalinic, Published by DAAAM International, Vienna, Austria 2011

Zusätzliche Informationen

Alumnus der internationalen Studentenorganisation AIESEC (www.aiesec.org)
Aktuelles Mitglied der ÖIAV-1848 - Österreichischer Ingenieur- und Architekten-Verein
Aktuelles Mitglied der DAAAM International - Verband der Ingenieure und Forscher in einem Gebiet der Automatisierung und Fertigung (www.daaam.org)
Mitglied des IDDS Team von Technische Universität Tallinn und HyNESS Lab von Boston University
Mitglied des internationalen Goldenflower Taj Chi Vereins (www.goldenflower.us)
Führerschein Kategorie B

Referenzen

Dissertationsbetreuer Prof. Dr.sc. Dr.mult. h.c. Prof.h.c. Katalinic, TU Wien, IFT, katalinic@mail.ift.tuwien.ac.at