applied sciences

MDPI

Article

Conceptual Planning of Micro-Assembly for a Better Utilization of Reconfigurable Manufacturing Systems

Christoph Gielisch [1,*], Karl-Peter Fritz [1], Benedikt Wigger [1] and André Zimmermann [1,2]

[1] Hahn-Schickard, Allmandring 9 b, 70569 Stuttgart, Germany; Karl-Peter.Fritz@Hahn-Schickard.de (K.-P.F.); Benedikt.Wigger@Hahn-Schickard.de (B.W.); zimmermann@ifm.uni-stuttgart.de (A.Z.)

[2] Institute for Micro Integration (IFM), University of Stuttgart, Allmandring 9 b, 70569 Stuttgart, Germany

* Correspondence: Christoph.Gielisch@Hahn-Schickard.de; Tel.: +49-711-685-83826

Received: 28 February 2020; Accepted: 10 April 2020; Published: 18 April 2020

Abstract: Reconfigurable manufacturing systems (RMS) can be used to produce micro-assembled products that are too complex for assembly on flat substrates like printed circuit boards. The greatest advantage of RMS is their capability to reuse machine parts for different products, which enhances the economical efficiency of quickly changing or highly individualized products. However, often, process engineers struggle to achieve the full potential of RMS due to product designs not being suited for their given system. Guaranteeing a better fit cannot be done by static guidelines because the higher degree of freedom would make them too complex. Therefore, a new method for generating dynamic guidelines is proposed. The method consists of a model, with which designers can create a simplified assembly sequence of their product idea, and another model, with which process engineers can describe the RMS and the procedures and operations that it can offer. By combining both, a list of possible machine configurations for an RMS can be generated as an automated response for a modeled assembly sequence. With the planning tool for micro-assembly, an implementation of this method as a modern web application is shown, which uses a real existent RMS for micro-assembly.

Keywords: micro-assembly; electronic packaging; product development; conceptual design; reconfigurable manufacturing systems; assembly sequence modeling

1. Introduction

1.1. Motivation

Micro-assembly is used in different fields of activity like electronics packaging or the production of optical devices. The main task is to pick, place, and join small components on substrates under the condition of tolerances in the micrometer range. The primary application in the industry is the mounting of components on flat substrates, most often printed circuit boards (PCB). This can be done using fast and precise pick-and-place machines. The whole PCB manufacturing process is standardized with special guidelines as a part of the so-called design for manufacturability (DFM) [1,2]. Due to this high standardization, new products can be developed and manufactured without the need for integral changes of machinery and overly frequent feedback loops. There are powerful software solutions that incorporate those DFM guidelines in PCB manufacturing [3]. They help engineers to design their products, so that they are automatically suited for the manufacturing systems used. However, this process comes with several limitations while designing a product. Most importantly, product engineers have to work with flat substrates, so components cannot be mounted on spatial substrates. Furthermore, those substrates may be flipped in the process, but not freely moved in space. Many micro-assembled products need to overcome those limitations to be produced, e.g., specialized sensors and actuators, optical assemblies, or parts with a high sensitivity to the installation space.

If the quantity needed is small and if it is feasible regarding tolerances and handling, such products can be built manually. Another option is special-purpose machines, constructed and built just for a single product or product family. However, those machines are costly and only pay off if they are highly utilized and in use long enough. This can be a problem in several situations:

- There is a gap in the upscaling process between the low volume manual and the high volume special-purpose production. Companies cannot commission foundries for such medium-sized volumes as would be possible with the highly standardized PCB process. Therefore, they are not able to realize efficiency benefits because of a better capacity utilization at foundries.
- Many markets have become more volatile in the last few decades, and product lifecycles are getting shorter [4,5]. This makes the forecast of a future production volume more difficult. Hence, investments in special-purpose machinery is getting riskier [6].
- Individualization is a growing trend for many products. Customers demand products explicitly tailored for their needs. While this is possible with manual manufacturing, it is not doable with special-purpose machines [7].

All these reasons lead to companies rejecting potential product ideas due to a missing feasible business model. Plant engineers try to improve the situation using reconfigurable manufacturing systems (RMS) that allow the manufactures to reuse large amounts of a machine system for different products [8–10]. RMS in the field of micro-assembly have a core machine that provides a power supply, a control system, and different slots to install processing modules. Often, those processing modules are separated into processing heads attached to an axis system and base plate systems attached to a base plate. Figure 1 exemplarily shows such a system.

Figure 1. The Desktop D1 from the OurPlant platform of Häcker Automation GmbH (Schwarzhausen, Germany) is a medium-sized reconfigurable manufacturing system (RMS) for micro-assembly purposes. The rendering shows the axis system with the attached processing heads, as well as the base plate with different installed base plate systems.

With RMS, it is possible to build up a manufacturing system by only adding some special-purpose modules to already existing standard modules in a core machine. This reduces the investment risk for the production of new products. Only the special-purpose modules may suffer from low utilization; the standard ones can be reused over multiple projects. If such modules can be equipped and removed reasonably fast, manufacturers are able to adjust their production rates rapidly. This is a necessary requirement for both the upscaling process, as well as working in a volatile economic environment. By using open interfaces for hardware and software development, RMS providers can additionally enhance their platform attractiveness.

While RMS provide suitable hardware for the previously mentioned problems, they do not solely solve them. RMS suffer from "over-the-wall" design [11]. Traditionally, product developers create new products based on an idea, which they transform into a design. Following this design, process engineers try to set up a suitable way to manufacture the product. This requires experts, product developers, as well as process engineers, to work closely together to achieve good results and prevent harsh redesigns. For PCB based products, this collaboration is accomplished by the DFM guidelines. However, for non-PCB products, developers often just throw their finished designs "over the wall", leaving the process engineers alone. This can still work, assuming these products are manufactured using manual labor or special-purpose machinery. In both cases, the process engineer has a high degree of freedom designing a matching production process. However, this does not work well with RMS, which have to use standard modules to reach their utilization goal.

The goal of this paper is to find a method with which product developers are able to design their micro-assembled, non-PCB products in a way that ensures the manufacturability on RMS. Because of the higher assembly complexity compared to PCB products, this cannot be done with just a simple set of guidelines. Instead, a sufficiently dynamic bundle of data, rules, and algorithms must be coupled with an easily accessible and operable user interface, forming a powerful software tool with which to work.

The development of a new product can be structured based on the systematic approach of Pahl and Beitz (SAPB) [12]. Figure 2 shows the four different phases of product development in the SAPB. The software tool must be embedded into the product development process. Changes on the design are less costly the earlier they are done [13]. Therefore, it is most promising to build the tool around the conceptual design phase [14].

Figure 2. The four phases of the systematic approach of Pahl and Beitz (SAPB).

1.2. Related Work

Finding improvements for the conceptual design phase in manufacturing and especially assembly is an active research field. A broad range of different techniques and methods exist to handle this phase in product development. Originally, the conceptual design phase stemmed from the SAPB and was based on a functional description of the new product on a system level. This is followed by a functional decomposition, in which the defined functions are broken down into smaller subsystems and ultimately building blocks. If suitable building blocks are not available, possible solutions have to be found. Subsequently, those building blocks are reintegrated into a complete product, whereby intense verification and validation ensure a satisfying achievement of the original objectives formulated in the functional description [12]. This can be seen as a top-down design (TDD) method. The main disadvantage of the conceptual design phase method of the SAPB is its pure focus on the product itself. Neither production processes, nor their resources are considered.

Other authors tried to improve the TDD by combining or refining it with other techniques. Komoto and Tomiyama presented a framework that included a consistency management of design information across different engineering disciplines to foster concurrent engineering [15]. Chu et al. used a multi-skeleton modeling approach to link a TDD with a modular product design (MPD) [16]. In this approach, different parts in the development of a product are partitioned into so-called skeletons. Those skeletons are connected by an overlying product description. The actual development can then be done by a TDD approach in each skeleton separately.

A similar method is model based system engineering (MBSE). It is based on connecting product developers of different domains by creating linked structures between models. Rudtsch et al. showed how this method can be used to combine an assembly sequence with a simulation model [17]. Furthermore, they looked at a non-PCB, micro-assembled product as a use case, but did not work with an RMS. For modeling, the Automation Modeling Language (AutomationML) was used.

Dori introduced the Object-Process Methodology (OPM) [18]. In OPM, a product concept can be modeled by objects and processes. Processes can alter the objects. This method is not singular product-centric, but instead implements elements of processing. OPM uses the Systems Modeling Language (SysML) to model and visualize a product. While this is a working approach for designing products, it is not easy to analyze those models automatically. Kapos proposed a model-to-model (MtM) technique to improve this situation [19].

Meng presented a method to model RMS and their reconfiguration with colored Petri nets (CPN) [20], but neither considered a connection to conceptual product development nor applied the method to micro-assembly. Wang and Dagli combined CPN with the OPM technique to create a holistic model. With this model, they were able to achieve a multi-objective optimization of an RMS configuration using a conceptual model as a base [21].

The U.S. National Institute of Standards and Technologies (NIST) defines an open assembly model (OAM) as an extension of their core product model. It incorporates assembly as a concept, as well as a data structure and can be implemented in various applications [22]. Another possibility in assembly modeling and analyzing is a datum flow chain. Mantripragada and Whitney proposed a method where an assembly was described by a liaison diagram and flow chains and analyzed by using a state transition model [23]. With this method, designers should be able to get feedback regarding their assembly quality in terms of tolerances when editing the assembly sequence.

1.3. Structure

Section 1 gives the main hypotheses of this paper and a short overview of the related work in this field. Section 2 describes the new method proposed. In Section 3, the implementation of the presented method as a software tool is presented, whereby a special focus is given to micro-assembly and suitable RMS. Finally, Section 4 discusses the findings of this work followed by concluding remarks in Section 5.

2. Method

The proposed method uses an assembly-centric instead of a part-centric design. Its idea is to start in the conceptual design phase before the first concrete construction drawings have been done. Product developers using the method could outline a simplified assembly sequence of their planned product. In response, they would obtain a collection of RMS configurations that should be able to fulfill the planned assembly sequence. Every time they would change the sequence, the response would change as well. Due to this automatism, designers could modify their sequence until the response fit an RMS configuration that would be okay for the whole development team including process engineers and financial management. The combination of the sequence found, as well as the configuration could then serve as a guideline for the whole product design.

Outlining a non-existing product seems to be contradictory, because the developer is requested to plan the assembly sequence, even if some of the assembly parts are not yet existing or are still subject to change. Furthermore, product developers are neither process engineers nor do they possess extensive

process knowledge. However, this is intended. The target of the outlining is not to create a detailed basis for the later process. Instead, a decision basis for product developers and process engineers should be derived that product designers can understand with which they can work. This decision basis can then work as a kind of dynamic guideline for a concurrent development of both disciplines. Still, the software tool must not overwhelm the product designers. To ensure this, some simple rules are set:

- The designer should not worry about which assembly sequence is the best sequence. He/she should just model one possible sequence. In particular, he/she should not be requested to model a liaison graph.
- The amount of inputs the designer has to give to get a result should be as small as possible. Giving more information to get a more precise result should be possible as well, but not necessary.
- The input parameters should be as easy as possible to understand. Parameters that are easy to define should be preferred over parameters that are hard to define. The earlier a parameter can be defined, the more likely it should be selected.

2.1. Assembly Sequence Model

Based on these thoughts, this work proposes an assembly sequence model consisting of interconnected assembly steps. Each step must have at least one input component and one output component. The connection between the steps can then be realized over the components. An output component could be the input component of another step and vice versa. The assembly sequence must result in one coherent network of steps without circular connections. Therefore, among other things, each step must be directly connected to at least one other step. Input components that can or should not be connected to an output component are generated out of sources. In a real process, they must be supplied. Output components that can or should not be connected to an input component are products of the assembly and must be discharged.

Each step can have a procedure type that characterized which kind of assembly must be done. Procedure types could be clustered into groups. Such groups could help designers describe a task that was not completely decided yet. The VDI 2860 norm defines assembly as the task of mating parts and all processes that are needed to achieve this, including handling, verifying, and adjusting [24]. As a basis for mating procedure types and their classification, DIN 8580 (manufacturing techniques) can be used [25]. It consists of six main groups of manufacturing, whereby mating is one of them. Each main group is separated into subgroups. The actual procedures are assigned to these subgroups.

The classification as done by DIN 8580 may not be necessarily perfect for any use case. Nevertheless, it is a good starting point to gain an overview of a possible grouping and, more importantly, the entire range of procedures. For the field of assembly, more than 100 possible procedure types arise from this structure. This method proposes that all possible procedure types should be checked based on the chosen RMS. If a procedure type can be executed with a module or with a combination of modules, the procedure type should be selectable in the modeling of the assembly sequence. For this, a new grouping of the found types can be introduced.

Picking a specific procedure type in the conceptual design phase may not be easy or sometimes even impossible. Furthermore, a designer voluntarily might not want to set it, because he/she simply does not care as long as an RMS can do the assembly afterwards. Therefore, the procedure type or its group need not be set. Still, more information will improve the quality of the resulting machine configuration by describing the task more precisely. Hence, users of the method can be supported by adding two types of properties: process properties and component properties. Process properties are tied to a specific assembly step, while component properties are applied to all steps that use the corresponding component. Both will influence the assembly and thereby the possible procedure types. Even if a designer is not able or willing to define a specific procedure type, he/she can set properties based on already chosen parts or the functional description of his/her product. For example, he/she can define if a joint should be reversible or if a component can endure high temperatures. On the other

hand, when directly selecting a procedure or group, he/she should be able to see which properties his/her process and the involved components have to fulfill. Similar to the procedure type, component properties and process properties are optional parameters, so they do not need to be given for a result. Figure 3 shows a scheme of an assembly sequence model for a product design.

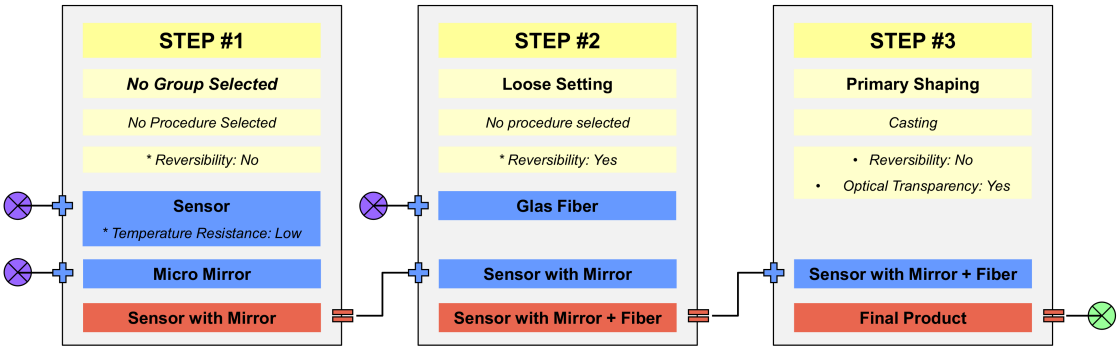

Figure 3. Schematic example of an assembly sequence model with three assembly steps. Procedure group, procedure type, and process properties are presented in the light yellow boxes. Blue boxes are used for input components and their properties, red ones for output components and their properties. Purple circles are sources, green circles wells. In the first step, a micro mirror is assembled to a sensor with low temperature resistance. This happens without further restrictions regarding the procedure group and type. The assembly should not be reversible. The created sensor with mirror is used as an input component in the second step. There, it is joined with a glass fiber to a sensor with mirror + fiber. This assembly step must use the group "loose setting" and be reversible. The last step must not be reversible, but optically transparent. The sensor with mirror + fiber is encapsulated, so the procedure group is "primary shaping", while the procedure type is "casting".

Reasonable properties may differ strongly based on the field of application, for example if the method is applied to a micro-assembly or the manufacturing of a car part. Therefore, component and process properties must be found taking the actual use case into account. When deciding which properties should be used, the number of properties should be as small as possible. It is neither necessary nor wanted to describe the procedure in detail. The only objective is to exclude unsuitable procedures fast using easy to determine parameters. Furthermore, the properties may influence each other. This means that the selection of some property combinations must be forbidden for the designer. All dependencies between properties and procedures, as well as among each other can be noted in constraint matrices.

2.2. Machine Configurations

Machine configurations are the feedback a user gets from modeling an assembly sequence. Typically, the machine configuration for an RMS consists of the following items:

1. Core machines: The core machines offer a working area for the modules and accessories. Normally, this happens through some kind of standardized space like a perforated plate or slots. In terms of software, this can be a suitable computer system and available disk space, as well. Furthermore, the core machine offers interfaces to connect the modules and accessories to its control system and the power supply. Often, but not always, core machines do not offer their own manufacturing capability. Every machine configuration must contain at least one core machine.
2. Modules: Modules are exchangeable hardware or software solutions that offer manufacturing capabilities, but require a working space and suitable interfaces. If reasonable, modules can be separated into different segments, for example based on the kind of interface or working area they need.

3. Accessories: Accessories do not offer their own manufacturing capability, but can support modules in doing so. This can mean that a module cannot work without the accessory or else that it works better with the accessory.

In the method, a designer gets a set of possible machine configurations as a result of his/her assembly sequence model. Each machine configuration in such a set is represented by a list of core machines, modules, and accessories. Additional information like the feasible performance or throughput, available capacities and an estimated price can be given as well, but is not necessary.

Machine configurations can be described by features and requirements. Requirements always need to be satisfied by a corresponding feature. There are different kinds of features and requirements that are of interest. One type is internal dependency related features and requirements. Those define which modules can be built into which core machines by determining how much and which kind of interfaces and working space they offer and need. Further dependencies can be necessary accessories for modules or incompatibilities between modules.

Another kind is manufacturing capability related features. To model capability features, this method introduces operations. Operations are actions a module can perform to fulfill a manufacturing task. For assembly, these are all operations needed to join components, as well as the handling, verifying, adjusting, and other auxiliary processes needed. Those operations may have properties like a range or force interval. The requirements in manufacturing capability originate from the manufacturing task the designer models in the assembly sequence. All actions can be modeled by using the following nine operations:

1. Positioning describes the action of moving or rotating a module. During positioning, a module can carry a component.
2. Inspecting describes the action of checking a component with a module. Often, this happens to find the exact position of a component by using a visual control with a camera system. However, other inspection methods like temperature control are possible as well.
3. Gripping describes the action of gripping, holding, and releasing a component with a module.
4. Dosing describes the action of applying a liquid or pasty substance on a component with a module.
5. Tooling describes the action of changing the tool. For example, this can be the gripper of a module that needs to be changed for every component.
6. Treating describes the action of manipulating a component with a module. This can be UV-curing of glue, thermal treatment, or mechanical deformation.
7. Conveying describes the action of moving components in, through, and out of the working area. Typically, this is realized with a conveyor belt.
8. Feeding describes the action of offering components to be picked, without the possibility to place them back.
9. Supplying describes the action of offering components a specific space to be picked and placed.

Finally, the performance is represented by the third kind of features. Performance features represent how fast or how economic a machine configuration can complete a manufacturing task. Typically, this includes features such as the movement and action speed or the price of a core machine, a module, or an accessory. Decision makers can define core performance indicators as requirements that must be met by a machine configuration.

2.3. Automated Response

One main challenge is the connection of the assembly sequence model with the machine configurations. Several problems must be solved to guarantee such a functioning automated response. The assembly sequence model can be seen as a complex representation of the requirements a designer has towards the RMS. Based on the assembly sequence model, an assembly process must be created. The assembly process differentiates itself from the assembly sequence model by setting a strict

procedure type for each assembly step. Table 1 gives an impression of how such an assembly process can look.

Table 1. An example for an assembly process based on the assembly sequence model in Figure 3.

No.	Predecessor	Procedure Type	Input Components	Output Components
1	-	UV bonding	Sensor Micro mirror	Sensor with mirror
2	1	Slide into each other	Glass fiber Sensor with mirror	Sensor with mirror + fiber
3	2	Casting	Sensor with mirror + fiber	Final product

In the assembly sequence model, the designer has the freedom to not precisely define the procedure type. This vagueness must be handled. This can be done by a purely stochastic approach. Instead of one assembly process, a list of all possible assembly processes is generated by combinatorial determination of all procedure configurations that are not excluded by the chosen parameters. Table 2 exemplarily shows some possible options for such a transformation of the assembly sequence model of Figure 3. The list of possible assembly processes can be seen as the solution space of the assembly sequence model and can get very long and unwieldy.

Table 2. Three possible options for how to resolve exemplarily the procedure type based on the user-given assembly sequence model of Figure 3.

Step No.	Procedure Type Option A	Option B	Option C
1	UV bonding	Bonding	Spring-spreading
2	Slide into each other	Inserting	Slide into each other
3	Casting	Casting	Casting

The list of possible assembly processes must be linked to a list of possible RMS configurations. This can be done by using the feature system of the machine configurations. With the operations of the capability features, a procedure type can be modeled. Therefore, operation sequences need to be introduced. Operation sequences consist of an ordered list of operations. Each operation in an operation sequence has an actor, an object, and properties. The actor is the module that is executing the operation. Objects are components, consumables, or accessories that are affected by the operation. With the properties, an operation can be modeled with more detail. Operation sequences can be created by already working assembly procedures or modeled by experienced process engineers. Table 3 shows a possible operation sequence for the procedure type "loading".

For each procedure type that is used in the method, at least one operation sequence must be existent. Otherwise, the correct functionality cannot be guaranteed, because this procedure type could not be resolved into an operation sequence. On the other hand, more than one operation sequence can exist for one procedure type. In that case, similar to the assembly processes, all possible combinations should be considered by a stochastic recombination.

An assembly process that is generated out of an assembly sequence model consists of several assembly steps, which are characterized by their procedure type. By substituting every assembly step of the assembly process with the operation sequence of the corresponding procedure type, an operation sequence for the whole assembly can be generated. This is called an assembly operation sequence. Such a sequence contains all modules that are needed to carry out the assembly. In particular, this sequence also precisely defines which modules may potentially be able to fulfill the given manufacturing task, by setting clear requirements towards the capability feature of the needed modules. The internal dependency requirements and features complete the machine configuration by adding additional items like core machines, modules, and accessories that are not directly listed in the assembly operation

sequence, but still necessary to build up a functioning system. With this algorithm, an assembly sequence model can automatically be transformed into a list of machine configurations and therefore provide an automated manufacturing feasibility response for the designer. Figure 4 gives an overview of all stages of the method when generating the feedback.

Table 3. Possible operation sequence for the procedure type "loading". Component B is placed on Component A. Both components are supplied in the working area by two modules (A and B). A third module (C) inspects both components to gather information about their specific position in the working area. Therefore, Module C must be positioned at a prior defined rough position. Afterwards, the fourth module (D) can be positioned at B's position. Module D can grip Component B, move it to the position of Component A, and release it there.

#	Operation	Actor	Objects
1	Supplying	Module A	Component A
2	Supplying	Module B	Component B
3	Positioning	Module C	-
4	Inspecting	Module C	Component A
5	Positioning	Module C	-
6	Inspecting	Module C	Component B
7	Positioning	Module D	Component B
8	Gripping	Module D	Component B, gripper
9	Positioning	Module D	-
10	Gripping	Module D	Component B, Component A, gripper

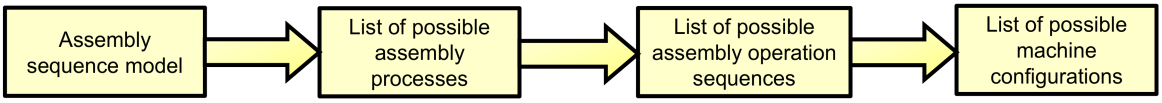

Figure 4. The four stages in generating an automated feedback for the designer: The assembly sequence model is being transformed into a list of possible assembly processes by stochastic combination. Next, a list of possible assembly operation sequences is created by using known operation sequences of procedure types. Finally, the feature based description of the RMS is used to generate the list of possible machine configurations.

3. Use Case: Planning Tool for Micro Assembly

RMS exist for many different fields of application, and the presented method may be applicable to a broad range of assembly processes. Nevertheless, this work is focused on micro-assembly. It will establish a planning tool for micro-assembly (PlaTooMA) based on the presented method in Section 2. To achieve this, a specific RMS for micro-assembly will be selected. Based on this RMS, a possible way to implement the method into a software solution is shown.

3.1. RMS

PlaTooMA was built upon the OurPlant manufacturing system of Häcker Automation GmbH (Germany). The OurPlant system was developed specifically for micro-assembly. It consists of four core machines that target different use cases from educational purposes to module and process development towards serial and large-scale production. Over standardized interfaces, more than 80 modules can be built inside the core machines. Miscellaneous accessories support the functional scope of the platform. All core machines, modules, and accessories are described in a web store called OurStore [26]. Additional information is given via the knowledge platform OurBase [27]. Furthermore, interviews with machine developers and process engineers were conducted to gather further information.

Following the proposed method in Section 2, the various procedure types of the DIN 8580 main group mating were checked. For each procedure type, a module or a module combination for the OurPlant system was searched that was able to perform this procedure type. If no such module or

module combination was found, this procedure type was excluded from further investigation. All in all, twenty-six procedure types were found that could be done with the selected RMS. To better aid the product developers, a slightly different grouping compared to DIN 8580 was introduced. Figure 5 shows the grouping, as well as eight selected examples for procedure types. In Table 4, all 26 procedure types can be found.

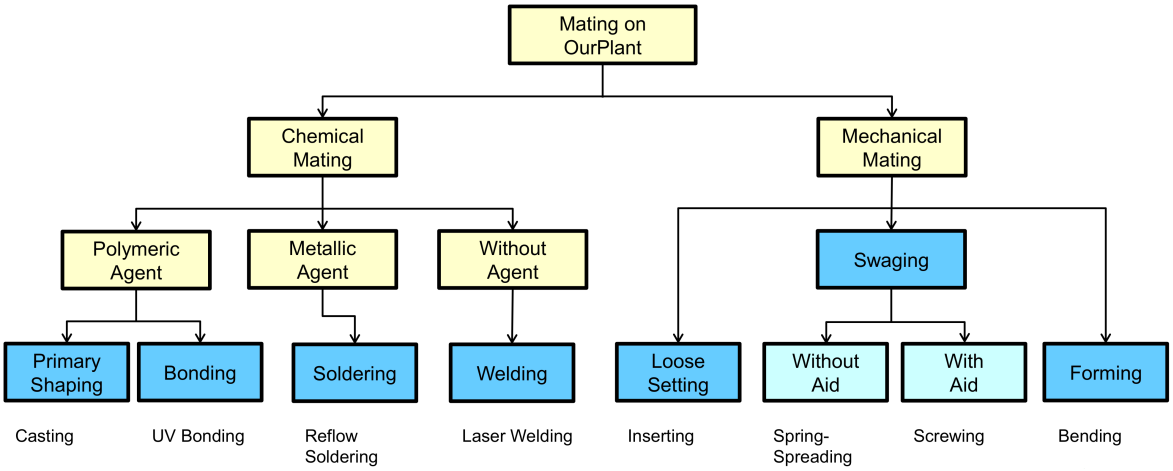

Figure 5. Overview of the new found procedure type grouping for micro-assembly. For each group, one example is given.

Table 4. The 26 procedure types and their corresponding groups.

Group	#	Procedure
Primary shaping	1	Casting
	2	Grouting
	3	Embedding
	4	Molding
	5	Caulking
Bonding	6	Bonding
	7	UV bonding
Soldering	8	Reflow soldering
	9	Laser soldering
	10	Resistance soldering
Welding	11	Laser welding
	12	Ultrasonic welding
Loose Setting	13	Loading
	14	Inserting
	15	Sliding into each other
	16	Hanging
Swaging (without aid)	17	Setting
	18	Spring-spreading
Swaging (with aid)	19	Screwing
	20	Clamping
	21	Cramping
	22	Nailing
	23	Wedging
	24	Bracing
Forming	25	Bending
	26	Crimping

For the development of PlaTooMA, one procedure type of each group was selected. Those eight procedure types were casting, UV bonding, reflow soldering, laser welding, inserting, spring-spreading, screwing, and bending. Based on the selected eight procedure types, process and component properties were searched following the outlined rules in Section 2. Three process and five component properties were found that proved to build up a sufficient system to exclude procedure types, while still following the established rules to be easy to understand and determined. Table 5 lists all eight properties, as well as a short description.

Table 5. Overview of the process and component properties to exclude unsuitable procedure types.

Type	Property	Description
Process properties	Reversibility	Can the procedure be reverted?
	Temperature resistance	Can the created connection endure high temperatures (>200 °C)?
	Optical transparency	Can the created connection be optically transparent?
Component properties	Cavity	Does the procedure need a cavity in a component?
	Metallic surface	Does the procedure need a metallic surface at a component?
	Temperature resistance	Does the procedure need a component with a high (>200 °C), a medium (200 °C–80 °C), or low (<80 °C) temperature resistance?
	Elastic deformability	Does the procedure need an elastically deformable component?
	Plastic deformability	Does the procedure need a plastically deformable component?

Each procedure type must be evaluated based on the selected properties. If a procedure type needs a property to be feasible, this must be noted. A fitting form to record those dependencies are constraint matrices. Tables 6 and 7 show the corresponding constraint matrices for the process and component properties and the eight selected procedure types.

Based on the nine possible operations, an operation sequence for each selected procedure type was found. Subsequently, a set of two core machines, eleven modules, and two accessories was selected to test the described method. This selection was done based on available information, as well as available hardware for future real-life tests and the eight selected procedure types, respectively their operation sequences. The eleven modules were split into five processing heads that can be installed at the head mounting space (HMS) at the y-axis of an xy-axis system and the base plate systems that can be installed at the base plate mounting space (BPMS). HMS and BPMS form the working space of the OurPlant system. For every machine part, the manufacturing capability features, the internal dependency features, and the internal dependency requirements were determined. Requirements, features, and also the corresponding operations had properties that defined them in more detail. Tables 8 and 9 list all machine parts, their features, and requirements.

Table 6. Constraint matrix for the process properties.

	Reversibility	Temperature Resistance	Optical Transparency
Casting	No	Yes	Yes
UV bonding	No	No	Yes
Reflow soldering	No	No	No
Laser welding	No	Yes	No
Inserting	Yes	Yes	Yes
Spring-spreading	Yes	Yes	Yes
Screwing	Yes	Yes	No
Bending	No	Yes	No

Table 7. Constraint matrix for the component properties.

	Cavity	**Metallic Surface**	**Temperature Resistance**	**Elastic Deformability**	**Plastic Deformability**
Casting	Yes	No	Yes (high)	No	No
UV bonding	No	No	No	No	No
Reflow soldering	No	Yes	Yes (high)	No	No
Laser welding	No	No	Yes (medium or high)	No	No
Inserting	No	No	No	No	No
Spring-spreading	Yes	No	No	Yes	No
Screwing	No	No	No	No	No
Bending	No	No	No	No	Yes

Table 8. Core machines and processing heads used for the planning tool for micro-assembly (PlaTooMA). HMS, head mounting space; BPMS, base plate mounting space.

Type	**Name**	**Manufacturing Capability Features**	**Internal Dependency Features**	**Internal Dependency Requirements**
Core Machines	D1	-	150 mm HMS 530 × 350 mm^2 BPMS Power supply 5× CAN ports 5× Ethernet ports	-
	X-Tec	-	150 mm HMS 500 × 500 mm BPMS Power supply 5× CAN ports 5× Ethernet ports Laser protection	-
Processing Heads	Camera 3D	Inspecting Positioning	-	49 mm HMS Power supply Ethernet port
	Universal Head 49-5	Gripping Positioning	-	49 mm HMS Power supply CAN port Tool adapter
	Dispenser D-X30	Dosing Positioning	-	49 mm HMS Power supply Ethernet port Drip tray
	UV curing head	Treating Positioning	-	49 mm HMS Power supply CAN port
	Laser Head Mergenthaler	Treating Positioning	-	79 mm HMS Power supply Ethernet port Laser protection

Table 9. Base plate systems and accessories used for PlaTooMA.

Type	Name	Manufacturing Capability Features	Internal Dependency Features	Internal Dependency Requirements
Base Plate Systems	Tool changing unit	Tooling		73×73 mm^2 BPMS Power supply CAN port
	Needle Inspecting Unit	Inspecting		180×290 mm^2 BPMS Power supply Ethernet port
	3D Alignment Support	Supplying Positioning		365×80 mm^2 BPMS Power supply Ethernet port
	Waffle Pack Support	Supplying		54×110 mm^2 BPMS
	Conveyer	Conveying		73×73 mm^2 BPMS Power supply Ethernet port
	Die Eject Unit	Feeding		500×300 mm^2 BPMS Power supply CAN port
Accessories	Tool adapter	-	-	-
	Drip tray (300 mL)	-	Drip tray	88×120 mm^2 BPMS

3.2. Implementation

For the implementation of the presented method, many variants were possible. To foster the accessibility and the ease of use, a web based single-page application was proposed for the assembly sequence modeling. This application is called Configurator. Users can work with such an application without the need for installing a dedicated software. It runs on every platform that supports a modern web browser. A web application can be developed by using existent frameworks like Angular, Ember, React, or Vue.js. For Configurator, Angular was used. Angular is an open-source web application framework. It was developed by Google and built upon Node.js. Node.js is an open-source JavaScript library, enabling server-side executed JavaScript web applications. Node.js also has the node package manager (npm) that allows the easy installation of Angular and other packages, as well as their dependencies.

An Angular application development is mainly based on components and services. Services are reusable TypeScript modules that can be imported and executed in different components. Components determine what happens on a patch of screen called a view [28]. They are written in TypeScript and offer classes, methods, loops, and more to enable a developer to build a dynamic application. The actual website, which is displayed in the browser, is realized with an accompanying template of the view. It defines a structure via a Hypertext Markup Language (HTML) file. This HTML file can be styled using different styling sheets. The most common and used with Configurator is the Cascade Style Sheet (CSS). The connection between this structure's HTML file and the TypeScript component was implemented via data bindings of variables. If the user changes something on the screen and therefore changes a defined variable in the HTML implementation, the Angular framework as a controller will automatically rerun the accompanying component logic and recalculate all involved variables. Through the data binding, this may re-affect the HTML structure and lead to a reloading of the website, displaying new or changed content.

With Angular, Material, an adaption of Google's Material Design philosophy, is available and used in Configurator. It can be installed using the npm. Angular Material offers versatile possibilities for dynamic user interface elements like buttons, forms, dialogs, or drop-down menus. Nevertheless, the modeling of assembly sequences requires an additional, more powerful, and flexible solution to

draw and modify objects on the screen. This can be achieved by a JavaScript 2D- or 3D-graphics library that is usable alongside Angular like fabric.js, PixiJS, or Konva. For Configurator, fabric.js was used, which is also installable through the npm. Figure 6 shows a screenshot of the implemented Configurator application.

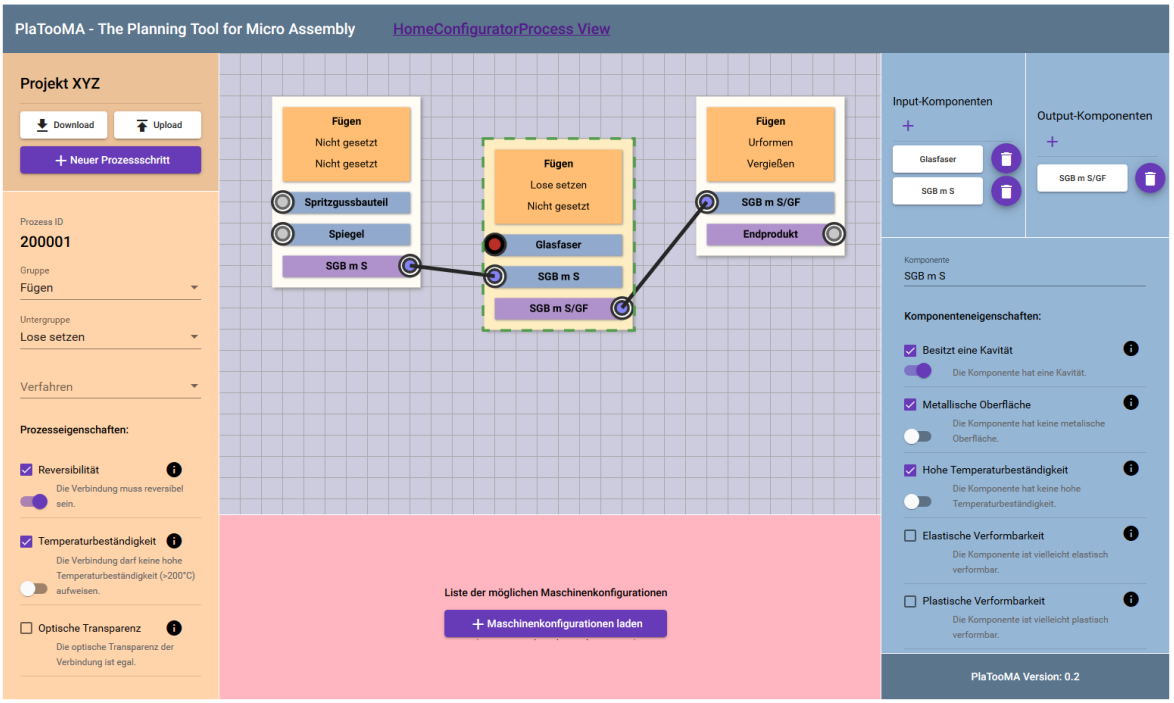

Figure 6. Screenshot of Configurator for PlaTooMA. At the grey canvas in the middle assembly steps can be added, arranged, and connected. The selected step is highlighted by a dashed green border. In the yellow area on the left, the procedure type and the process properties of the selected step can be set. In the blue area on the right, input and output components of the selected step can be added and deleted. Their properties can be modified as well. By clicking on the purple button in the red bottom area, fitting machine configurations can be shown.

Two databases were established for the implementation of the method. One was for the feature based description of the selected RMS. This was the Machine Database. The other collected the operational sequences for the procedure types. This was the Assembly Database. For both databases, a SQL system was used. SQL databases work with relational database management to store data. They are a broadly used technique in web development, offering fast and stable access to data. Many different implementations of SQL servers exist. Well known open-source SQL servers would be MySQL, PostgreSQL, or MariaDB. For the two databases in this work, PostgreSQL was used.

To finalize the implementation of the method in PlaTooMA, the modeled assembly sequence in Configurator must be analyzed and transformed into a machine configuration. Therefore, an application must exist that can receive the modeled assembly sequence, has access to both databases, and is able to perform the necessary algorithms to transform the assembly sequence to a list of possible operation sequences and subsequently generate the list of possible machine configurations.

To achieve these objectives, an application based on a web service was proposed. Representational State Transfer (REST) is a technique to build web services. Over the Hypertext Transfer Protocol (HTTP), an application can transmit to and receive data from a REST web server. This is called a RESTful web service [29]. Configurator must encapsulate the modeled assembly sequence in a standardized file format to communicate with the RESTful web service. Therefore, the JavaScript Object Notation (JSON) was used. Figure 7 shows the structure for assembly sequences in JSON files. Similar to the assembly

sequence, a JSON structure for the list of possible machine configurations must be established, as such a list is the expected answer of the RESTful web service.

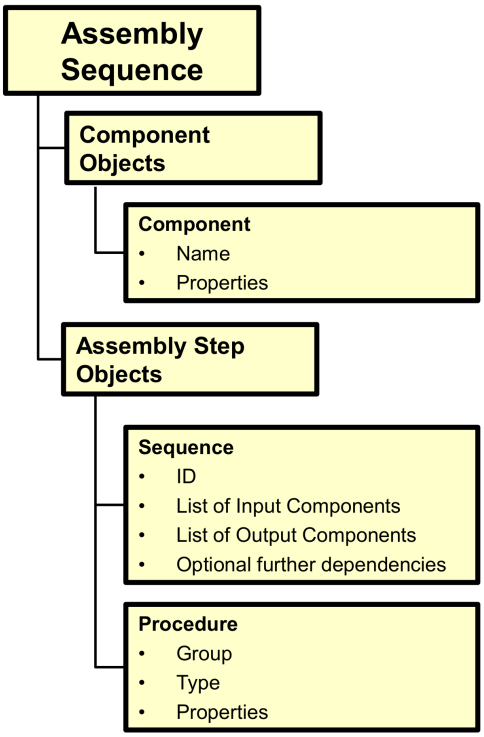

Figure 7. Overview of the hierarchical object structure of the assembly sequence JSON files.

The Django REST framework enables developers to build a RESTful web service with the Django web framework. Django is an open-source developed project written in Python. It offers a way to write applications in Python that can generate and control HTTP requests. Examples are web pages or web services. For this, Django implements its own web server. In addition, it has an object-relational mapper (ORM) to easily access data in SQL databases. With the ORM, a software developer can work with the data in a SQL database as they would be stored in Python objects and variables. Therefore, data models must be defined using a special type of Python class. The ORM migrates these data models to a relational SQL structure in a SQL database. Over different routines, various SQL databases are supported, among others PostgreSQL. With a serializer, rules can be implemented and how new instances of a data model are created and stored. The Django REST framework is able to do the following actions:

1. Receive an assembly sequence through a JSON file in an HTTP request.
2. Transfer the JSON file into a corresponding Python object.
3. Trigger the relevant transforming algorithms to generate a list of possible machine configurations.
4. Convert the list of possible machine configurations to a JSON file.
5. Offer the JSON file with the list of possible machine configurations as an HTTP response.

Configurator must be able to receive the HTTP response of the RESTful web service. The generation of this response may take a while depending on the complexity of the assembly sequence and its processing. During this time, the user of Configurator should not be blocked and instead be able to work on the assembly sequence. Therefore, the communication between Configurator and the RESTful web service must be asynchronous. This may require a mechanism to skip machine configuration generation or to clock the HTTP requests, to prevent flooding of requests if the assembly sequence is modified to fast. Figure 8 shows an overview of the presented software architecture.

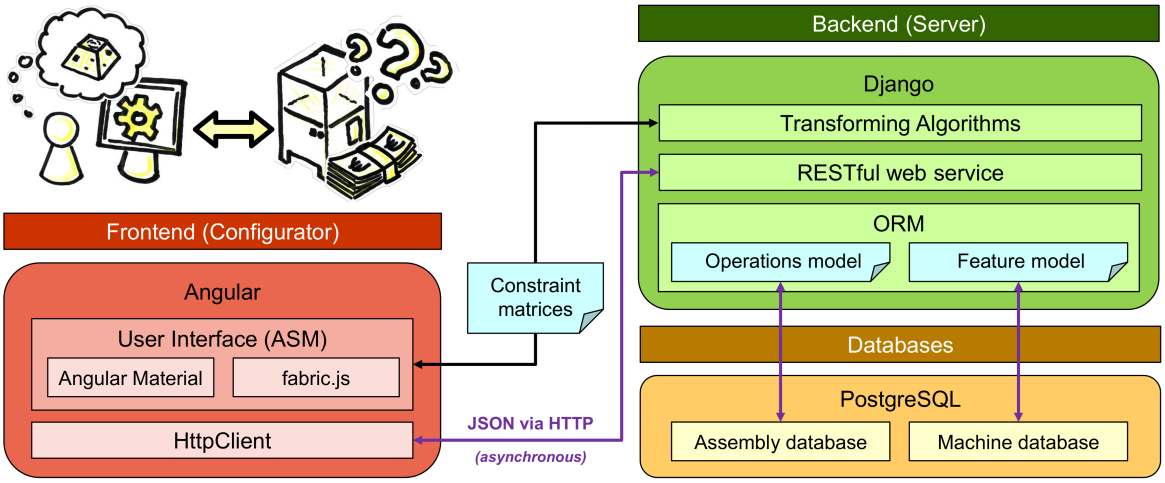

Figure 8. Overview of the whole software architecture of PlaTooMA.

4. Discussion

The primary goal of this work was to find a way to help designers improve the manufacturability of their designs on RMS with an emphasis on micro-assembled, non-PCB products. Methods found in literature lacked several important aspects. The conceptual design as introduced by the SAPB is only part-centric and does not include processing issues. Therefore, it does not solve the "over the wall" problem of conceptual design. Other TDDs try to overcome those issues by adding more improved methods, but do not link them to the special needs of RMS and also do not show the ways that an implementation can happen. Methods based on the datum flow chain show little interest in an easy-to-use designer integration, only optimize the assembly sequence, and do not find possible machine configurations.

OPM is useful to describe an assembly task, but lacks proper ways to be automated. MtM techniques can work around this issue, and with CPN, a method for RMS description exists. Still, modeling a product with objects and processes is not as accessible as modeling an assembly sequence. Beyond that, the OPM techniques do not include methods to work with non-determined product descriptions by the designer and do not give instructions on how to reach RMS solutions that fit the needs of process engineers and management.

With the presented method and its implementation, an easy-to-use software tool could be introduced, with which designers could model simplified assembly sequences of their designs before the construction has been done. Designers obtain live feedback with which machine configurations of the selected RMS their assembly sequence model can be realized. Together with process engineers and the management, they can decide which machine configurations should be targeted. By this selection, a collection of properties for processes and components is set that can act as a combined working frame for a concurrent development of a product and process. This helps to avoid costly redesigns in the later product development phases and can be seen as a kind of dynamic guideline system. By setting targeted machine configurations, the utilization of RMS can be fostered, which decreases the investment risk of RMS. The basic requirement for a successful implementation is powerful and well defined databases. With the machine and the assembly database, the presented method showed the ways such databases could be generated.

To prove the feasibility of the introduced method and the proposed implementation, PlaTooMA was developed as a use case. With a web based application and a state-of-the-art user interface that followed modern design principles, it had a low entry barrier and great accessibility. Following the method, relevant procedure types could be detected. With a selection of those procedures, fitting properties for components and processes were found. Constraint matrices for both property types were introduced that helped to exclude unsuitable procedure types and thereby helped designers

model their assembly sequence. With the OurPlant manufacturing system, the integration of a specific RMS in PlaTooMA was shown.

Several future improvements of the method, implementation, and use case are possible. No standardized modeling languages were used to describe the assembly sequence, the machine configurations, or the feature based description of machine parts and operations for the databases. This need not to be a disadvantage in every case. Especially the assembly sequence modeling benefited from being tailored to designer needs, had no strong ties to other systems, and may not be able to support the same degree of freedom, while still providing an automated analysis. However, describing machine parts and their features is often necessary for various situations and applications. With a standardized description or modeling language, the machine configurations would be reusable. For example, a machine configuration could be used as a digital twin in the later production or as a virtual test bed for missing module development. This would greatly enhance software applications like PlaTooMA.

At the moment, PlaTooMA generates too many possible machine configurations even if the modeled assembly sequence is not overly complicated. This may overwhelm the decision makers. To improve this situation, the machine configurations should be filtered, evaluated, and ranked. Filtering machine configurations can work through excluding duplicates and dominating machine configurations. For the ranking, criteria have to be found to rate machine configurations. This can happen with an evaluation function. A simple input parameter is the price a machine configuration costs. Another input would be the performance of the system. To test the performance of a machine configuration, simulations would be a possible way. However, they are too time consuming to be done for every possible machine configuration, even if some kind of filtering has been done before. Artificial intelligence algorithms could be a way to hasten this process by evaluating the performance indirectly with simulation knowledge instead of direct simulations.

5. Conclusions

With a new method and its implementation, this work showed how a higher utilization of RMS in the field of non-PCB, micro-assembled products could be achieved. This was accomplished with an enhanced conceptual design phase in the product development process. Product designers can model an assembly sequence by defining assembly steps with input and output components, as well as already known properties for both the components and their processing. Unknown properties at this time in development need not to be specified. As a result, machine configurations for a specific RMS can be generated that should satisfy the needs of the modeled assembly. By selecting some of these machine configurations, a collection of properties can be set that could act as a dynamic guideline for the product and the process development. By using this method, both disciplines, product, as well as process development, could simultaneously work on a new product without losing too many degrees of freedom regarding their design. The finished product and process could later be executed on the chosen RMS without the need for excessive redesign loops or expensive special-built machinery.

This would only work with well defined databases that inherit processing knowledge. Each assembly could be done with a chain of operations called an assembly operation sequence. The corresponding database allowed the automated transformation of assembly sequences into assembly operation sequences. A second database was able to map those operations to suitable machine parts of an RMS. The combination of both databases made the automated generation of machine configurations possible. The method pointed out ways such databases could be systematically and successfully generated. With PlaTooMA, a software tool was developed that was able to prove the feasibility of both the method and implementation. It was built as a modern web application using state-of-the-art technologies. As a use case, OurPlant RMS was successfully applied.

Author Contributions: Conceptualization, C.G.; methodology, C.G.; software, C.G.; validation, C.G., K.-P.F., B.W., and A.Z.; formal analysis, C.G.; investigation, C.G.; resources, C.G.; data curation, C.G.; writing, original draft preparation, C.G.; writing, review and editing, C.G., K.-P.F., B.W., and A.Z.; visualization, C.G.; supervision, B.W.; project administration, C.G.; funding acquisition, K.-P.F. All authors read and agreed to the published version of the manuscript.

Funding: This research was funded by the Federal Ministry of Education and Research of Germany in the framework of "Methodische Werkzeuge zur Erhöhung der Wandlungsfähigkeit von Mikromontage-Anlagen bei Entwicklung, Konfiguration und Monitoring (MIKROKOMO)", Grant Number 02P15A120.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

BPMS	Base plate mounting space
CPN	Colored Petri nets
DFM	Design for manufacturability
HMS	Head mounting space
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
MBSE	Model based system engineering
MPD	Modular product design
MtM	Model-to-model
NIST	National Institute of Standards and Technologies
npm	Node package manager
OAM	Open assembly model
OPM	Object-process methodology
ORM	Object-relational mapper
PCB	Printed circuit board
PlaTooMA	Planning tool for micro assembly
REST	Representational State Transfer
RMS	Reconfigurable manufacturing system
SAPB	Systematic approach of Pahl and Beitz
SysML	Systems Modeling Language
TDD	Top-down design

References

1. Krishnan, S.; Srihari, K. A knowledge-based object oriented DFM advisor for surface mount PCB assembly. *Int. J. Adv. Manuf. Technol.* **1995**, *10*, 317–329. [CrossRef]
2. Chow, J.; McCormick, H.; Hamilton, C.; Berry, M.; Cortero, R.; Facchini, G. DFM Rules for Smartphones: An Analysis of Yield on Extremely Dense Assemblies. In Proceedings of the IPC APEX EXPO Technical Conference 2011, Las Vegas, NV, USA, 12–14 April 2011; p. 11.
3. Smith, G.L. Utilization of STEP AP 210 at the Boeing Company. *Comput.-Aided Des.* **2002**, *34*, 1055–1062. [CrossRef]
4. Abele, E.; Elzenheimer, J.; Liebeck, T.; Meyer, T. Globalization and Decentralization of Manufacturing. In *Reconfigurable Manufacturing Systems and Transformable Factories*; Dashchenko, A.I., Ed.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 3–13._1. [CrossRef]
5. Doheny, M.; Nagali, V.; Weig, F. Agile operations for volatile times. *McKinsey Q.* **2012**, *3*, 126–131.
6. Abele, E.; Liebeck, T.; Wörn, A. Measuring Flexibility in Investment Decisions for Manufacturing Systems. *CIRP Ann.* **2006**, *55*, 433–436. [CrossRef]
7. Kölmel, B.; Pfefferle, T.; Bulander, R. Mega-Trend Individualisierung: Personalisierte Produkte und Dienstleistungen am Beispiel der Verpackungsbranche. In *Dialogmarketing Perspektiven 2018/2019: Tagungsband 13. Wissenschaftlicher Interdisziplinärer Kongress für Dialogmarketing*; Verband eV, D.D., Ed.; Springer Fachmedien: Wiesbaden, Germany, 2019; pp. 243–260._11. [CrossRef]

8. Koren, Y.; Shpitalni, M. Design of reconfigurable manufacturing systems. *J. Manuf. Syst.* **2010**, *29*, 130–141. [CrossRef]
9. Hees, A.; Reinhart, G. Approach for Production Planning in Reconfigurable Manufacturing Systems. *Procedia CIRP* **2015**, *33*, 70–75. [CrossRef]
10. Koren, Y.; Wang, W.; Gu, X. Value creation through design for scalability of reconfigurable manufacturing systems. *Int. J. Prod. Res.* **2017**, *55*, 1227–1242. [CrossRef]
11. Boothroyd, G.; Alting, L. Design for Assembly and Disassembly. *CIRP Ann.* **1992**, *41*, 625–636. [CrossRef]
12. Pahl, G.; Beitz, W. *Engineering Design: A Systematic Approach*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013. Google-Books-ID: 4uvSBwAAQBAJ.
13. Reich, Y. What is a reference? *Res. Eng. Des.* **2017**, *28*, 411–419. [CrossRef]
14. Gielisch, C.; Fritz, K.P.; Noack, A.; Zimmermann, A. A Product Development Approach in The Field of Micro-Assembly with Emphasis on Conceptual Design. *Appl. Sci.* **2019**, *9*, 1920. [CrossRef]
15. Komoto, H.; Tomiyama, T. A framework for computer-aided conceptual design and its application to system architecting of mechatronics products. *Comput.-Aided Des.* **2012**, *44*, 931–946.
16. Chu, D.; Chu, X.; Li, Y.; Lyu, G.; Xue, D. A multi-skeleton modelling approach based on top-down design and modular product design for development of complex product layouts. *J. Eng. Des.* **2016**, *27*, 725–750. [CrossRef]
17. Rudtsch, V.; Bauer, F.; Gausemeier, J. Approach for the Conceptual Design Validation of Production Systems using Automated Simulation-Model Generation. *Procedia Comput. Sci.* **2013**, *16*, 69–78. [CrossRef]
18. Dori, D. *Model-Based Systems Engineering with OPM and SysML*; Springer: New York, NY, USA, 2016. [CrossRef]
19. Kapos, G.D. Enabling system models automated evaluation through cross-concept information utilization. In Proceedings of the 2015 IEEE 9th International Conference on Research Challenges in Information Science (RCIS), Athens, Greece, 13–15 May 2015 . [CrossRef]
20. Meng, X. Modeling of reconfigurable manufacturing systems based on colored timed object-oriented Petri nets. *J. Manuf. Syst.* **2010**, *29*, 81–90. [CrossRef]
21. Wang, R.; Dagli, C.H. Developing a Holistic Modeling Approach for Search-based System Architecting. *Procedia Comput. Sci.* **2013**, *16*, 206–215. [CrossRef]
22. Rachuri, S.; Han, Y.H.; Foufou, S.; Feng, S.C.; Roy, U.; Wang, F.; Sriram, R.D.; Lyons, K.W. A Model for Capturing Product Assembly Information. *J. Comput. Inf. Sci. Eng.* **2006**, *6*, 11–21. [CrossRef]
23. Mantripragada, R.; Whitney, D. Modeling and controlling variation propagation in mechanical assemblies using state transition models. *IEEE Trans. Robot. Autom.* **1999**, *15*, 124–140. [CrossRef]
24. *VDI 2860—Montage- und Handhabungstechnik; Handhabungsfunktionen, Handhabungseinrichtungen; Begriffe, Definitionen, Symbole*; Verein Deutscher Ingenieure (VDI): Düsseldorf, Germany, 1990.
25. *DIN 8580:2003-09, Fertigungsverfahren_- Begriffe, Einteilung*; Technical Report; Beuth Verlag GmbH: Berlin, Germany, 2003. [CrossRef]
26. OurStore | Market Place for the Micro Assembly Platform OurPlant. Available online: https://store.ourplant.net/ (accessed on 25 February 2020).
27. OurBase | The Digital Knowledge Database for Microassembly. Available online: https://base.ourplant.net/ (accessed on 25 February 2020).
28. Angular—Introduction to Components. Available online: https://angular.io/guide/architecture-components (accessed on 25 February 2020).
29. Richardson, L.; Ruby, S. *RESTful Web Services*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2008.