Planning in assembly systems – A common modeling for products and resources

Julien Provost, Bengt Lennartson and Martin Fabian
Signals and Systems
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
{provost, bengt.lennartson, fabian}@chalmers.se

Åsa Fasth and Johan Stahre
Product and Production Development
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
{asa.fasth, johan.stahre}@chalmers.se

Abstract

This paper presents a method for modeling robot and human resources in the context of assembly systems planning. In assembly systems, several redundant resources can be used to increase system flexibility. However, the "quality" of a sequence planning strongly depends on the "quality" of the system modeling. Furthermore, occurrence of unexpected events or variations in availability of resources may have significant impact on the actual planning. Instead of using simplistic models such as available or unavailable resources, the method presented in this paper proposes a more detailed modeling of resource abilities. Products and resources are considered on the same levels and matched together on a final step. The aim of this modeling is to permit analyses and to increase system flexibility.

1 Introduction

To meet sustainability, industries face different issues and have to deal with their economic, social and environmental impacts. In the context of assembly systems, important issues are system flexibility, human utilization, and human quality of work. Indeed, system flexibility is mandatory to meet the increasing demands on mass customization and reduced time-to-market. Meanwhile, sustainable human resource policies aims at improving employment and quality of work for human operators. In practice, these issues are not independent, humans are intrinsically more flexible than machines, while automation permits to increase safety, efficiency and accuracy. A sustainable work organization is thus characterized by convergence between flexibility and working quality, both leading to an increase in productivity.

To achieve this goal all involved parts of a production system: products, resources, information systems and control functions must be well defined and coordinated such that switching between manual and automated operations is easily achieved.

Designing a product (or a group of products) and its manufacturing and assembly system requires several phases: product design, material choice, parts manufacturing, final assembly, control code generation... These phases are usually performed in parallel by different engineering teams. Thus, collaboration and coordination is needed to achieve a functional and efficient system. Several modeling languages, methods and tools have been developed in order to facilitate communication and information handling between different teams and from different organizational levels, e.g. SysML language and PLM tools [14].

Operations and sequences of operations are common elements that interconnect products, resources, process and automation information. In [9] a formal modeling language has been introduced to model and visualize Sequences of OPerations (SOP). This graphical language, called *SOP language* in the sequel of this paper, relies on self-contained operations that can be viewed from different points of view. This language is based on automata extended with variables, which permits to use it directly to apply supervisory control theory synthesis [15], verification techniques, and to perform optimization.

In this paper, the SOP language is further developed to introduce timed operations. Resource modeling is also enhanced by the introduction of the concepts of *active* and *passive operations* and the definition of *operating modes*.

2 Background on sequences of operations and sequence planning

This paper focuses on definition of resource models for planning of sequences of operations. Sequences of operations permit to express relationships between the different operations that should be executed to build a product. Indeed, sequences of operations are a common thread to many engineering tasks, e.g. in the three following main tasks:

1. Definition of the relations between operations (e.g. precedence relations)

2. Planning of sequences of operations. This planning relies on an optimization problem and also considers matching between product operations and resources performing these operations.

3. Control of the execution of these sequences of operations. Control includes logic control of the plant and rescheduling of sequences of operations due to modification of orders, time variation or occurrence of unpredicted events during system execution.

An important task when designing an automated assembly system is to specify in which order the different operations should be executed. This task, called *sequence planning*, aims at defining sequences of operations that, depending on the research field and the targeted industrial applications, guarantee different optimization criteria. For example, [20] and [11] consider construction and optimization of assembly sequences according to *product quality* and geometric deviations propagation. Several works considering resource allocation and line balancing use *total assembly time* as an optimization criterion [6]. Indeed, assembly sequences cannot be optimal according to several criteria; in that case a cost function can be defined using weighted criteria. Definition of a cost function and optimization of assembly sequences become even more complex when human resources and product variants must be considered [4, 21].

3 Motivations and method presentation

In previous works a formal language has been developed in order to define self-contained operations and describe sequences of operations (SOP) [5, 9]. A software tool called Sequence Planner has been implemented to handle the SOP language and help designers to define requirements, such as described above, and to visualize relations between operations [13].

3.1 Motivations

The modeling method presented in this paper extends existing methods in order to be able to integrate the following sources of disturbance when planning and optimizing sequences of operations:

- Preventive maintenance operations;
- Corrective maintenance operations;
- Time availability of human operators.

Preventive maintenance operations are predictable and can be planned in advance. However, since maintenance operations are often performed by human operators they should be planned to both prevent production disruption and fit with the human operators availability and workload.

On the other hand, corrective maintenance operations are due to occurrence of unpredicted events. Indeed, in large and complex industrial systems occurrence of faults is inevitable. *Diagnosis* is the process of detecting and isolating faults in a system and prevent or limit the deviations of the system behavior. During recent years, Fault Detection and Isolation (FDI) has been the subject of much research, both for timed and non-timed discrete event systems [19, 3, 16]. Once a fault is detected and isolated, and after corrective maintenance has been performed, the system must be restarted. To restart the system in its normal production mode, the controller and the physical system must be resynchronized so that their states match. A method for restart of automation system has been proposed by [1]. This method defines off-line the states from where a safe restart of the system is possible and then facilitates the resynchronization of the physical plant and its controller.

Finally, contrary to robots or machines that are often dedicated to a limited number of tasks and are very repeatable, human operator's availability is more subject to variation. For example, breaks should be taken into account, but also meetings or training activities. Even though these disruptions can be roughly predicted they are still subject to time variation, both in occurrence and duration time.

Contrary to reactive planning, which considers rescheduling only after a failure occurs, proactive planning considers planning ahead of the eventuality of failure occurrence. Planning of preventive maintenance operations, using varying levels of automation [7], and off-line computation of alternatives in case of unavailability of a specific resource are among the solutions permitting to increase proactivity of a system.

Instead of developing a new model, the results presented in this paper extend the SOP language with a definition of timed operations and detailed resource modeling. Thus, the underlying model remains the same, which permits to be easily integrated into existing tools [8].

3.2 Method presentation

The method presented in this paper extends the SOP language with the following concepts:

- Timed operations;
- Operating modes;
- Active and passive operations.

Timed operations are defined using the minimal duration of an operation. Timed operations permit to perform sequence planning using time availability of resources. The formal definition of a timed operation will be detailed in section 4.2.

In general, each resource (human operator, machine or robot) can perform a huge variety of operations. In order to organize them and ease understandability, resources can be structured using a hierarchical relation called *operating mode*.

When planning operations within an assembly system, products and resources are often considered from two different points of view. The assembly sequence of operations is first considered from the product point of view, and then resources are associated to each product operation. In order to ease definition of product operation and resource abilities, the same basis of SOP representation will be used in what follows. However in order to distinguish operations that should be performed to build a product and operations that a resource can perform, the terms *active operation* and *passive operation* are introduced. A *passive operation* is an operation that needs a resource with the corresponding ability to be performed, while an *active operation* is an operation that can be performed by a resource. For instance, both active and passive operations can be associated to a robot resource. A robot can perform operations on a product. These operations are *active* from the robot point of view but *passive* from the product point of view. However, a robot may need operations to be performed on it by another resource, e.g. maintenance tasks performed by a human operator. These operations are *passive* from the robot point of view but *active* from the human operator's point of view. In order to distinguish passive and active operations, passive operations are represented using dotted lines.

The method proposed in this paper uses a slightly different approach to model the relations between product and resources. This approach is based on three steps:

1. Modeling of the operations that should be executed to build a product;
2. Modeling of the operations each resource can perform and definition of its operating modes;
3. Matching of operations between the product description and the resource description.

First step of this approach is described in section 5, while steps 2 and 3 are presented in sections 6 and 7, respectively.

3.3 Illustrative example

The method presented in this paper will be illustrated through the product presented in Figure 1. This *product* is composed of two pieces: *Part A* and *Part B*, assembled together with seven *Rivets*. Three different resources can be used to build this product: a fixture, a robot and a human operator.

To assemble the product the following operations should be performed:

- Place parts *A* and *B*: *PlaceA* and *PlaceB*
- Fixate parts *A* and *B*: *FixA* and *FixB*
- Assemble parts *A* and *B* together: *AssAB*
- Inspect the final assembly: *InspectAB*. *InspectAB* is a parent operation of child operations *InspGeo* and *InspRiv*.

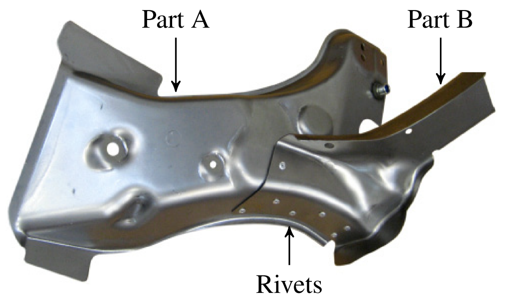


Figure 1. Product used to illustrate the proposed method

These operations can be performed by at least one resource according to the following ability list:

- A fixture: can perform *FixA* and *FixB*
- A robot: can perform *PlaceA*, *PlaceB*, *AssAB* and *InspGeo*
- A human operator: can perform *PlaceA*, *PlaceB*, *AssAB* and *InspRiv*

Detailed definition of these operations and resource abilities are given sections 5 and 6, respectively.

4 SOP language – Timed definition

The SOP language, introduced in [9], is based on self-contained and hierarchical operations. These models can be used for formal verification, controller synthesis and time optimization [13]. The operations and the relations between them are modeled by Extended Finite Automata (EFA), which are ordinary automata augmented with bounded integer variables, guard formulas and action functions [18].

In order to extend the SOP language with timed operations, operations are modeled using Timed EFA (TEFA). The following subsections present the TEFA model and give the definition of a timed operation according to the SOP language.

4.1 Timed Extended Finite Automaton

A TEFA is an EFA augmented with a finite set of digital clocks. The comprehensive syntax and semantics of TEFA are given in [10], below are given the main important concepts:

- A transition in a TEFA is executed if and only if its corresponding event is enabled and its corresponding guard formula is satisfied;
- When a transition is executed, updating actions of a set of variables and clocks may follow;
- All clocks are assumed to evolve synchronously with the same rate;

- Time only elapses at locations (transitions are executed instantaneously);
- When several TEFA are composed, a shared event is enabled if and only if it is enabled by each of the composed TEFA (all related transition guards must be true).

In this paper TEFA are assumed to be deterministic.

4.2 Definition of a timed operation

Using the SOP language, an operation O_k is defined by a TEFA with three locations O_k^i, O_k^e and O_k^f. Locations O_k^i, O_k^e and O_k^f represent that the operation O_k starts, is being executed, and is finished, respectively.

Figure 2 gives an example of an operation represented using the SOP language while Figure 3 gives its corresponding TEFA.

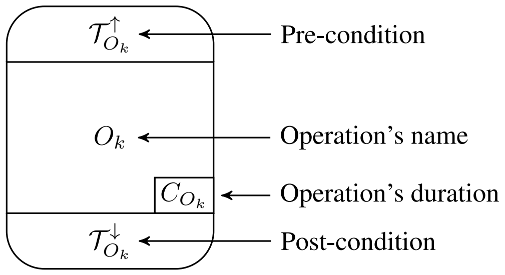


Figure 2. Operation O_k modeled using SOP language

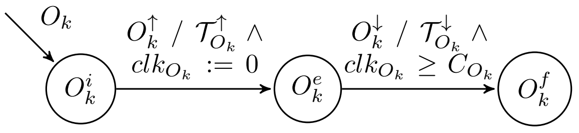

O_k
$O_k^{\uparrow}$ / $\mathcal{T}_{O_k}^{\uparrow} \wedge$ $clk_{O_k} := 0$
$O_k^{\downarrow}$ / $\mathcal{T}_{O_k}^{\downarrow} \wedge$ $clk_{O_k} \geq C_{O_k}$
O_k^i
O_k^e
O_k^f

Figure 3. TEFA of an operation O_k

In order to perform sequence planning using time availability of resources, the duration of all operations must be known. A timed operation is defined using estimation of its minimal duration. Indeed, the duration of an operation is usually related to the process and in not an intrinsic requirement from a product point of view. Thus, the definition of the operation's duration is mandatory for active operations and optional for passive operations. The value of a clock belonging to an operation can be used as a comparison variable to define pre- and post-conditions of other operations, but its value can only be reset by the operation it belongs to. Supervisory control synthesis can then be applied to guarantee a nonblocking behavior. For example, an operation with an minimal duration of X time-units should be allowed to start at the latest X time-units before the end of the time availability of the resource which will perform it.

For notational purposes and to ease definition of pre- and post-conditions, the TEFA definition is slightly reformulated. A set of transition conditions $\mathcal{T}_k$ of an operation can be composed of sets of both guards $\mathcal{G}_k$ and actions $\mathcal{A}_k$. This reformulation eases the definition of conditions such as resource booking which consists of both a guard (the resource must be available, $\mathcal{G}_k : R = 0$) and an action (book the resource, $\mathcal{A}_k : R := 1$). The resulting transition condition $\mathcal{T}_k$ is $R = 0 \wedge R := 1$. The shortcuts R^+ and R^- are introduced to simplify expression of booking conditions, and unbooking conditions, respectively.

To represent operations that should be repeated, a reset transition between O_k^f and O_k^i can be added to the initial definition. Additional transitions, or sequences of locations and transitions, can also be added to represent restart of the system after a failure. These transitions are not generic and may not be applied to all operations; they can be automatically computed and added to the initial model [2].

5 Modeling of product operations

As previously mentioned, the definition of relations between operations can be related to several design and implementation phases. Thus, instead of using explicit sequence constructions that are hard to handle for large scale systems, especially when data come from different engineering teams, the SOP language eases expression of relations between operations through pre- and post-conditions. The SOP language is based on the fact that a sequence of operations should not be considered as input data but as the result of relevant pre- and post-conditions defining relations between operations. The following examples illustrate the use of pre- and post-conditions in different phases.

- Product design phase:
  Precedence relations can be expressed through pre-conditions, e.g. a hole needs to be drilled before riveting. Results from previous quality evaluation and tolerance analysis can also be introduced in order to prioritize a specific assembly sequence.
- Resource booking:
  If operation O_1 must be performed by the specific resource R_1, then R_1 must be booked before O_1 is executed and unbooked after.
- Implementation control:
  To generate the control code for the implementation, information from the plant need to be taken into account. For instance, sensor values can be expressed through variables and permit to express conditions that must either be satisfied to start an operation (pre-condition) or to stop this operation (post-condition).
- Other purposes:
  Additional pre- and post-conditions can be generated automatically in order to solve issues related to safety, deadlock avoidance, collision avoidance, etc. [17].

The first step of the approach presented in this paper only considers the definition of pre- and post-conditions related to requirements on the product.

For the example presented section 3.3, the requirements are:

- *Part A* should be placed before being fixated;
- *Part B* should be placed before being fixated;
- *Part A* and *Part B* should be fixated before being assembled;
- The *Product* should be inspected after *Part A* and *Part B* being assembled;
- The *Product* is considered finished once inspection has been done.

As previously mentioned, product operations usually do not contain estimation of their duration. For this example, there is no minimal operation's duration related to product requirements.

Figure 4 gives the definition of these requirements for each individual operation while Figure 5 gives the resulting SOP from the product point of view.

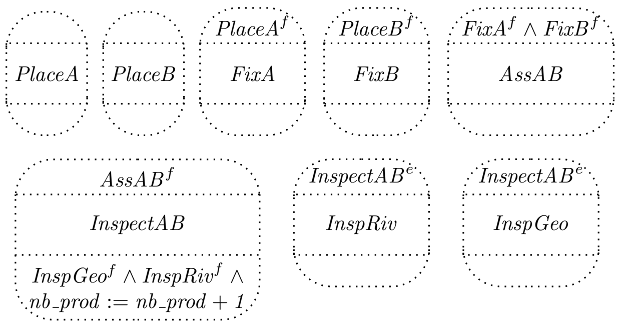


Figure 4. Product operations modeled using the SOP language: Requirements

According to the requirements on the product, operations $PlaceA$ and $FixA$ can be executed in parallel with operations $PlaceB$ and $FixB$. Then, operations $AssAB$ and $InspectAB$ must be executed sequentially. Finally, operations $InspGeo$ and $InspRiv$ can be executed in parallel. All these operations are defined as *passive operations*.

The number of products that have been assembled and inspected is defined by the variable nb_prod. Variable nb_prod is incremented by 1 in the post-condition of operation $InspectAB$. If 250 products should be produced, the marked value for variable nb_prod is 250.

6 Modeling of resource operating modes

The concept of *operating mode* presented in this section aims at facilitating organization and understandability of resource abilities. This *hierarchical relation* also permits to handle occurrence of unexpected events such as faults.

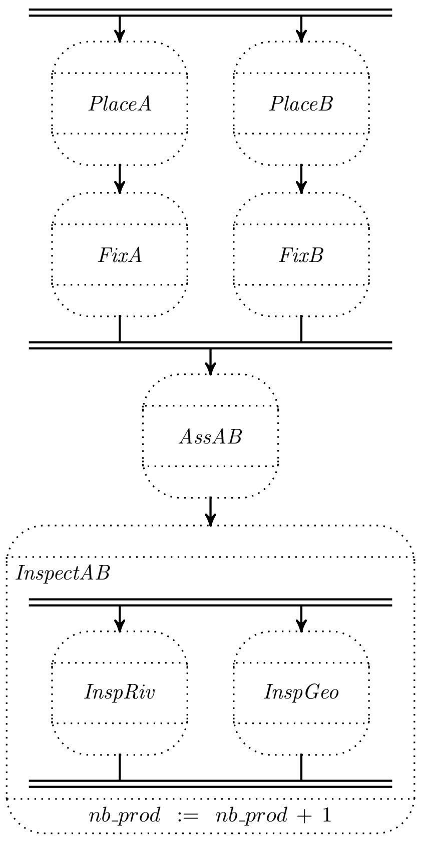


Figure 5. Product operations modeled using the SOP language: Sequence

The *hierarchy* relation, defined in [9], is denoted $O_p \sqsubset O_c$, meaning that operation O_p is a parent of the child operation O_c. This hierarchy relation means that O_p^e is included in the pre-condition of O_c, and O_c^f is included in the post-condition of O_p. The hierarchy relation can also be applied to a child sequence of operations. An example of the hierarchy relation is illustrated in Figure 5, where $InspectAB$ is a parent operation of child operations $InspGeo$ and $InspRiv$.

6.1 Definition

In comparison to the *hierarchy* relation, the *operating mode* relation, introduced in this paper, corresponds to a different hierarchical relation. The *operating mode* relation, denoted $O_m \ltimes O_k$, means that operating mode O_m includes and enables operation O_k. This *operating mode* relation means that O_m^e is included in both the pre-condition and the post-condition of O_k but O_k^f is not included in the post-condition of O_m. This means that if operating mode O_m finishes (or is deactivated) while operation O_k executes, O_k cannot finish. An operation which defines an operating mode is prioritized over its included operations. This priority permits to model interruption due to occurrence of a fault, for instance.

An operation which defines an operating mode can have a minimal duration. This minimal duration is not mandatory, though, but if a duration is defined, its meaning is the same as for ordinary operations. However, since the post-condition of an operation and its minimal duration are related (see Figure 3), this duration is also prioritized over durations of its included operations.

Figures 6 and 7 give representations of the *hierarchy* relation and *operating mode* relation, respectively.

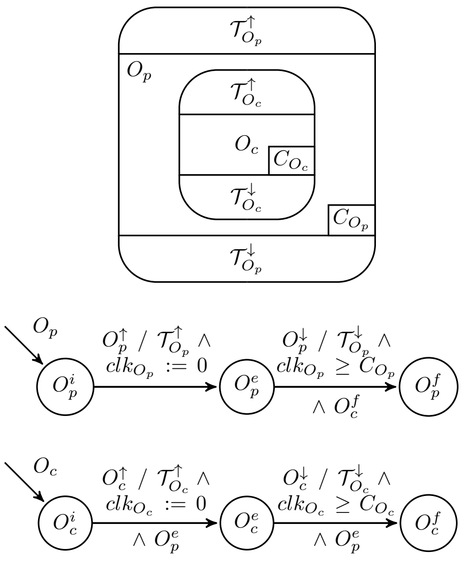

$\mathcal{T}^{\uparrow}_{O_p}$
O_p
$\mathcal{T}^{\uparrow}_{O_c}$
O_c
C_{O_c}
$\mathcal{T}^{\downarrow}_{O_c}$
C_{O_p}
$\mathcal{T}^{\downarrow}_{O_p}$
$O^{\uparrow}_p$ / $\mathcal{T}^{\uparrow}_{O_p} \wedge clk_{O_p} := 0$
$O^{\downarrow}_p$ / $\mathcal{T}^{\downarrow}_{O_p} \wedge clk_{O_p} \geq C_{O_p} \wedge O^f_c$
$O^{\uparrow}_c$ / $\mathcal{T}^{\uparrow}_{O_c} \wedge clk_{O_c} := 0 \wedge O^e_p$
$O^{\downarrow}_c$ / $\mathcal{T}^{\downarrow}_{O_c} \wedge clk_{O_c} \geq C_{O_c} \wedge O^e_p$

Figure 6. *Hierarchy* relation model using SOP language and TEFA

6.2 Illustration

For the example presented in section 3.3, several operating modes could be defined for each resource. Here, only a reduced number of modes will be considered. Figures 8 and 9 give the details about the operating modes corresponding to the robot and the human operator. The fixture is composed of only one operating mode (*Production* mode); this mode includes operations *FixA* and *FixB*.

In the *Production* operating mode, the human operator and the robot have several redundant operations (*PlaceA*, *PlaceB* and *AssAB*). However, *InspGeo* can only be performed by the robot while *InspRiv* can only be performed by the human operator.

In the *Maintenance* operating mode, operation *Maintain* is defined for both the human operator and the robot. However, *Maintain* is a passive operation for the robot while it is an active one for the human operator. For this *Maintain* operation, the human operator can perform *Maintain* on the robot.

The three sources of disturbance previously presented (preventive maintenance operations, corrective maintenance operations, and time availability of human operators) are implemented in this example.

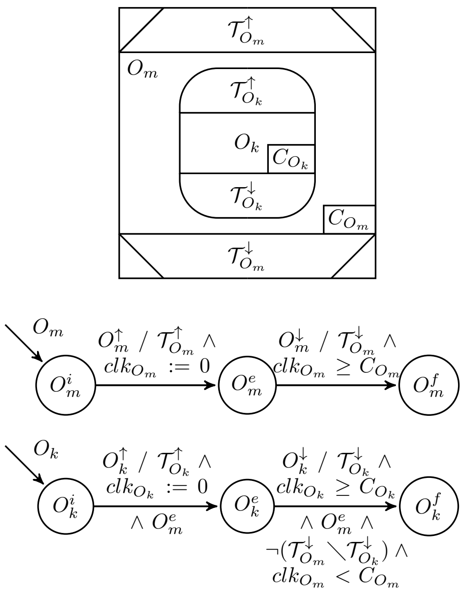

$\mathcal{T}^{\uparrow}_{O_m}$
O_m
$\mathcal{T}^{\uparrow}_{O_k}$
O_k
C_{O_k}
$\mathcal{T}^{\downarrow}_{O_k}$
C_{O_m}
$\mathcal{T}^{\downarrow}_{O_m}$
$O^{\uparrow}_m$ / $\mathcal{T}^{\uparrow}_{O_m} \wedge clk_{O_m} := 0$
$O^{\downarrow}_m$ / $\mathcal{T}^{\downarrow}_{O_m} \wedge clk_{O_m} \geq C_{O_m}$
$O^{\uparrow}_k$ / $\mathcal{T}^{\uparrow}_{O_k} \wedge clk_{O_k} := 0 \wedge O^e_m$
$O^{\downarrow}_k$ / $\mathcal{T}^{\downarrow}_{O_k} \wedge clk_{O_k} \geq C_{O_k} \wedge O^e_m \wedge \neg(\mathcal{T}^{\downarrow}_{O_m} \setminus \mathcal{T}^{\downarrow}_{O_k}) \wedge clk_{O_m} < C_{O_m}$

Figure 7. *Operating mode* relation model using SOP language and TEFA

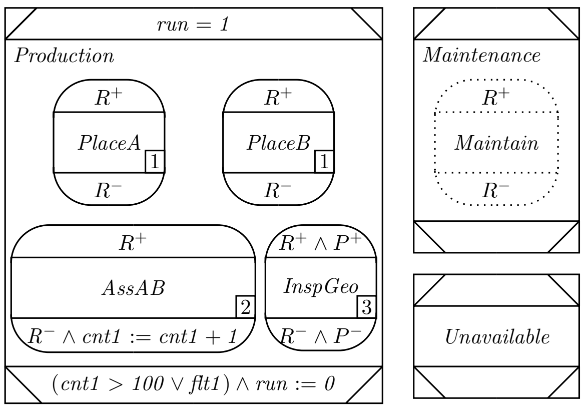


Figure 8. Operating modes of the robot

An example of preventive maintenance operations is implemented using the variable *cnt1*. The variable models a counter which is incremented by one when operation *AssAB* is executed by the robot and finishes (see the post-condition of operation *AssAB* in Figure 8). The variable *cnt1* is also used to define a post-condition of operating mode *Production* of the robot. This latter post-condition means that when *cnt1* is equal to 100 the operating mode *Production* for the robot is deactivated and none of the operations included in this mode can be executed. Operating mode *Production* can only be reactivated if variable *run* is true; this variable is set to 1 in the post-condition of the operation *Maintain* performed by the human operator.

An example of corrective maintenance operations is implemented using the variable *flt1*. In order to take into

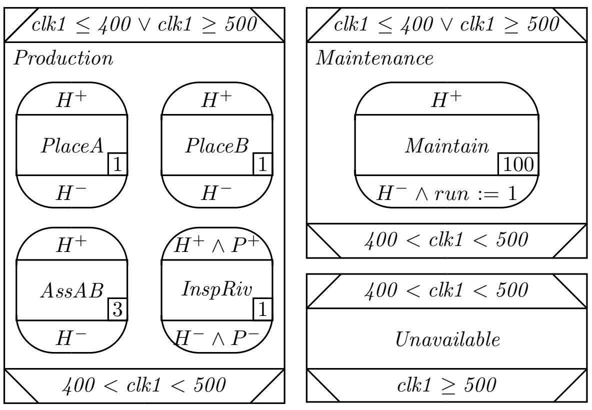


Figure 9. Operating modes of the human operator

account fault occurrence in the model, the fault needs to be detected. This is the role of a diagnoser, when the fault event *fault1* occurs the diagnoser is assumed to set the variable *flt1* to 1. Contrary to preventive maintenance, the occurrence of the fault event is not predictable. Thus, deactivation of the *Production* operating mode is not synchronized with any other operations. In that case, a restart of the system is necessary to synchronize the physical plant and its controller.

In this example, time availability of the human operator is modeled using the clock *clk1*. The variable *clk1* is used both in the pre- and post-condition of *Production* and *Maintenance* operating mode for the human operator. These modes can be activated if the value of clock *clk1* is less than 400 or more than 500; once they are active, they remain active until the value of clock *clk1* satisfies $400 < clk1 < 500$.

Since a resource can only perform a limited number of operations simultaneously, each resource needs to be booked. This resource booking is modeled using the variables R and H. If both the robot and the human operator can perform only one operation at a time, the domain of R and H is $\{0, 1\}$. Remember that the shortcuts R^+, R^-, H^+ and H^- represent booking and unbooking conditions.

The definition of operating modes also permits reusability of existing components. For example, a *Ramp-up* mode can be designed using a copy of the *Production* mode and by applying global parameters on the mode instead of modifying individually each operation contained in this mode (e.g. if a human operator is required to supervise the system during ramp-up)

7 Matching passive and active operations

When the product operations, the resource operating modes and the resource operations are defined, passive and active operations can be matched. Matching of passive and active operations is performed according to the following rules:

- Operations are matched by name;
- Each passive operation should be matched by at least one active operation;
- If a passive operation matches several active operations, alternative sequences are generated. Each alternative corresponds to the matching of the passive operation by one active operation;
- The pre- and post-conditions of the resulting operation are defined by the conjunction of both pre-conditions, and both post-conditions, respectively;
- The minimal duration of the resulting operation is equal to the maximum of both minimal durations;

Two examples are presented below, the *Maintain* and *AssAB* operations.

The *Maintain* operation is a passive operation from the robot point of view but an active operation from the human operator point of view. Figure 10 represents the operation *Maintain* after matching.

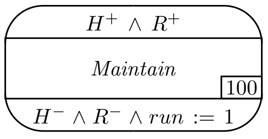
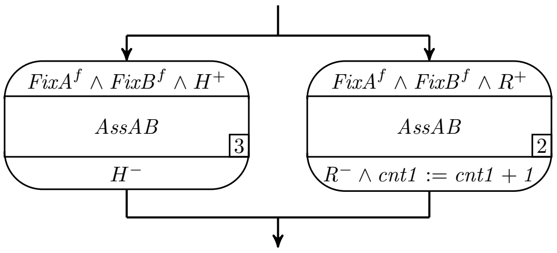


Figure 10. Operation *Maintain* after matching

The *AssAB* operation is a passive operation from the product point of view but an active operation from both the human operator and the robot points of view. Figure 11 represents the operation *AssAB* after matching. An alternative is generated, either the robot or the human operator can perform this operation.



$FixA^f \wedge FixB^f \wedge H^+$
AssAB
3
H^-
$FixA^f \wedge FixB^f \wedge R^+$
AssAB
2
$R^- \wedge cnt1 := cnt1 + 1$

Figure 11. Operation *AssAB* after matching

Formally, passive and active operations can be matched by applying an algorithm which for every operation in the set of passive operations generates an alternative SOP involving all corresponding operations in the set of active operations.

8 Conclusion and prospects

The method presented in this paper extends the SOP language with timed operations and proposes a new approach to model product operations and resource abilities using a common modeling language. Matching of passive and active operations permits to generate alternative sequences when resource abilities are redundant. Composition of TEFA permits to express all the alternatives available to reach a goal (defined by marked locations and marked values of variables). Since products and resources are modeled on a same level, this permits to reduce the gap between operation management and resource management, which is a huge issue when human resource are considered [12].

On-going work considers planning optimization with regard to sustainability. Two aspects are considered: system flexibility and quality of work for human operators. More precisely, current work focuses on routing flexibility (quantification and optimization) and workload smoothness. Quantification of routing flexibility would permits to analyze system flexibility independently of time or cost function. This quantification would aim to identify potential bottle-necks and help to select "best" sets of alternative sequences within assembly sequences. Quantification and optimization of human workload smoothness aims at increasing work quality, which is essential to meet sustainability.

References

[1] K. Andersson, B. Lennartson, and M. Fabian. Restarting manufacturing systems; restart states and restartability. *Automation Science and Engineering, IEEE Transactions on*, 7(3):486–499, 2010.

[2] K. Andersson, B. Lennartson, P. Falkman, and M. Fabian. Generation of restart states for manufacturing cell controllers. *Control EngineeringPractice*, 19(9):1014–1022, 2011.

[3] F. Basile, P. Chiacchio, and G. D. Tommasi. An efficient approach for online diagnosis of discrete event systems. *Automatic Control, IEEE Transactions on*, 54(4):748–759, 2009.

[4] C. Becker and A. Scholl. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168(3):694–715, 2006.

[5] K. Bengtsson. *Operation Specification for Sequence Planning and Automation Design*. Licenciate thesis, Chalmers University of Technology, 2010. ISSN 1403-266x.

[6] C. Del Valle, E. F. Camacho, and M. Toro. A model for assembly sequence planning in a multirobot environment. In *Proceeding of the 15th IFAC World Congress*, pages 187–192, 2002.

[7] J. Frohm, V. Lindström, M. Winroth, and J. Stahre. Levels of automation in manufacturing. *Ergonomia IJE&HF*, 30(3), 2008.

[8] K. Åkesson, M. Fabian, H. Flordal, and R. Malik. Supremica - an integrated environment for verification, synthesis and simulation of discrete event systems. In *8th International Workshop on Discrete Event Systems (WODES 2006)*, pages 384–385, 2006. Software available online: http://www.supremica.org.

[9] B. Lennartson, K. Bengtsson, C. Yuan, K. Andersson, M. Fabian, P. Falkman, and K. Åkesson. Sequence planning for integrated product, process and automation design. *IEEE Transactions on Automation Science and Engineering*, 7(4):791–802, 2010.

[10] S. Miremadi, Z. Fei, K. Åkesson, and B. Lennartson. Symbolic nonblocking computation of timed discrete event systems. 2012. Technical Report, Chalmers University of Technology. Submitted to 51st IEEE Conference on Decision and Control (CDC 2012).

[11] M. Mounaud, F. Thiebaut, P. Bourdet, H. Falgarone, and N. Chevassus. Assembly sequence influence on geometric deviations propagation of compliant parts. In *11th CIRP Conference on Computer Aided Tolerancing*, 2009.

[12] W. Neumann and J. Dul. Human factors: spanning the gap between OM and HRM. *International Journal of Operations & Production Management*, 30(9):923–950, 2010.

[13] E. Ohlson and C. Torstensson. Development, implementation and testing of sequence planner – a concept for modeling of automation systems. Technical Report EX/2009, Dept. Signals and Systems, Chalmers University of Technology, Göteborg, Sweden, 2009.

[14] S. Rachuri, E. Subrahmanian, A. Bouras, S. Fenves, S. Foufou, and R. Sriram. Information sharing and exchange in the context of product lifecycle management: Role of standards. *Computer-Aided Design*, 40(7):789–800, 2008.

[15] P. Ramadge and W. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, 1989.

[16] M. Roth, S. Schneider, J. Lesage, and L. Litz. Fault detection and isolation in manufacturing systems with an identified discrete event model. *International Journal of Systems Science*, 2012.

[17] M. R. Shoaei, B. Lennartson, and S. Miremadi. Automatic generation of controllers for collision-free flexible manufacturing systems. In *6th IEEE Conference on Automation Science and Engineering (CASE 2010)*, pages 368–373, 2010.

[18] M. Sköldstam, K. Åkesson, and M. Fabian. Modeling of discrete event systems using finite automata with variables. In *Proceeding of the 46th IEEE Conf. Decision and Control*, pages 3387–3392, 2007.

[19] S. Tripakis. Fault diagnosis for timed automata. In *Proceedings of the International Conference on Formal Techniques in Real Time and Fault Tolerant Systems (FTRTFT'02)*, volume 2469 of *Lecture Notes in Computer Science*, pages 205–224. Springer, 2002.

[20] H. Wang and D. Ceglarek. Quality-driven sequence planning and line configuration selection for compliant structure assemblies. *CIRP Annals - Manufacturing Technology*, 54(1):31–35, 2005.

[21] X. Zhu, S. Hu, Y. Koren, and N. Huang. A complexity model for sequence planning in mixed-model assembly lines. *Journal of Manufacturing Systems*, 31:121–130, 2012.