Human operator and robot resource modeling for planning purposes in assembly systems

J. Provost[1], Å. Fasth[2], J. Stahre[2], B. Lennartsson[1] and M. Fabian[1]

1: Signals and Systems - Chalmers University of Technology - SE-412 96 Gothenburg - Sweden
2: Product and Production Development - Chalmers University of Technology - SE-412 96 Gothenburg - Sweden

This paper presents how robot and human resources can be modeled for planning purposes. Instead of using simplistic models such as available or unavailable resources, the method for modeling resources presented in this paper integrates parameters that are realistic and relevant for the considered assembly system. For example, a robot resource model can take into account maintenance tasks and ramp-up parameters. The framework of this modeling is based on the definition of Sequences of Operations (SOPs) and includes a formal relation between product operations and resources abilities. The main idea is to avoid the representation of long and static sequences of operations, since this typically reduces flexibility and is even intractable for large systems. To tackle this issue, relations between operations and resources are defined using only strictly necessary pre-conditions and post-conditions for each individual operation. The Sequences of Operations that permit to express the minimally restrictive behavior of an assembly system are automatically generated. Finally, the SOPs can be viewed from different angles, e.g. from a product or a resource perspective. These multiple views increase the interoperability between different engineering disciplines. Experiments have shown that, even for simple examples, obtaining the optimized assembly sequence is not an easy task [1]. That is why a sequence planning software associated to realistic resource models, including both humans and robots, as presented in this paper, is a crucial help to increase flexibility in assembly systems that require different Levels of Automation [2].

Modelling, Planning, Automation, Assembly, Flexibility, Task allocation

1. Introduction

To achieve high productivity and flexibility, assembly systems commonly need to use different Levels of Automation, i.e. both robots or machines and human operators are considered as resources. Tasks allocation between these resources is often planned in the design phase of the system due to machine investments, etc. However, if task allocation is done in early design phases, occurrence of unpredicted events leads to an inflexible system that cannot be dynamic and proactive during execution. For instance, if a robot breaks down, this robot cannot continue to assemble products and it must be repaired; this will have two consequences. The first consequence is that the tasks, or operations, allocated to the robot must be reallocated. The second consequence is that a repair task must be performed on this robot by another resource.

For consistency reasons, in the remainder of this paper, the term *operation* will refer to what the resources of the assembly system can perform or execute, and the term *task* will refer to high-level phases, e.g. design tasks.

For complex systems, the dynamic allocation of operations cannot be performed manually. Thus, in order to perform optimization on this allocation of operations, both operations and resources need to be precisely modeled. This paper is aiming to define these operation and resource models.

The proposed method for modeling resources will be illustrated in the paper through a simple example. The sequence of operations and the allocation of operations for this example are generated using the Sequence Planner software [3].

2. Sequences of Operations and sequence planning

An important task when designing an automated assembly system is to specify in what order the different operations can or must be executed. This task is called sequence planning and permits to define Sequences of Operations (SOP). In the following subsections a formal SOP language and a software tool that handles this language will be presented. Interested readers are referred to [4], [5] for further information.

2.1. Product example

The method presented in this paper will be illustrated through the example presented in Figure 1. This example is a *Product* composed of two pieces: *Part A* and *Part B*, assembled together with seven *Rivets*. This product is produced in a cell composed of three resources: a robot, a fixture and a human operator. These resources can perform different operations; their abilities can be redundant or not as will be detailed in the next sections. The following description briefly explains how the product is to be produced:

- First, *Part A* and *Part B* must be placed on the fixture. They can be placed either by the robot or the human operator.
- Then, *Part A* and *Part B* are fixated on the fixture by clamps controlled by the fixture itself.
- Then, *Part A* and *Part B* are assembled together with seven *Rivets*. This operation can also be performed either by the robot or the human operator.

- Finally, the product is inspected. Depending on which resource performs the operations to place *Part A* and *Part B* and the assembly operation, the inspection operation differs as will be detailed in section 4.3.

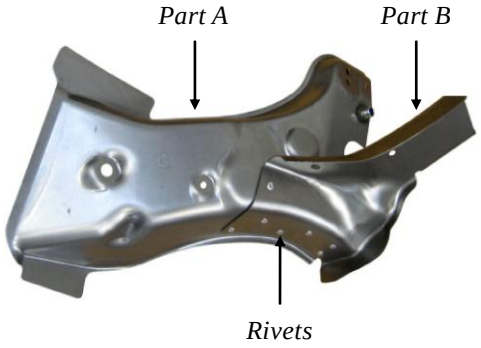

Figure 1. Product used to illustrate the proposed method

2.2. SOP language

The Sequences of Operations (SOP) language [4] is a graphical language used to specify and visualize relations among operations. This SOP language is based on an operation model. Sequences of operations are defined with the help of pre- and post-conditions related to each operation. Figure 2 presents how an operation can be represented using the SOP language.

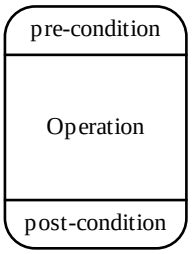

Figure 2. Representation of an operation using SOP language

Operations represented using the SOP language can be translated into Finite State Machines (FSM) [4], [5]. This FSM formalization can hence be used to apply verification and validation techniques and supervisory controller calculation.

The pre- and post-conditions can be related to several design or implementation phases. 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 [6], [7] can also be introduced in order to prioritize a specific assembly sequence.

- Resource booking:

If operation *O1* must be performed by resource *R1*, then *R1* must be booked before *O1* 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 permits 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 [8], etc. This point will be explained in the next subsection.

Thus, instead of explicit and static sequence construction that is hard to handle for large scale systems, the SOP language permits to express relations between operations through pre- and post-conditions. Thus, the SOP language permits more flexibility during the different product and process design, and implementation phases, due to support by an underlying formal computational engine [3], [9] that resolves blocking and deadlocks and guarantees safety.

As a conclusion, this SOP language is based on the fact that a sequence of operations should not be considered as a compulsory input data but as a consequence of relevant pre- and post-conditions on when and how the operations can be executed. Figure 3 and Figure 4 give an example of operations that are related through pre-conditions. Figure 3 represents the set of six operations that must be performed on the product given in Figure 1 and illustrates pre- and post-conditions of operation *Fixate A*. Figure 4 represents explicitly the relations between these six operations. According to the requirements on the product, operations *Place A* and *Fixate A* can be executed in parallel with operations *Place B* and *Fixate B*. Then, operations *Assemble A+B* and *Inspect A+B* must be executed sequentially.

Pre-conditions, and respectively post-conditions, of an operation can be composed of guards and actions. Those guards and actions are defined through variables. A guard is a condition that must be satisfied so that the operation can start (or finish). An action permits to define or change the value of variables when the operation starts (or finishes). For instance, a pre-condition related to the booking of a resource is both a guard and an action: the resource needs to be available (*R1==available*) and the resource is then booked (*R1=booked*). The pre-condition associated to the resource booking is (*R1==available* ∧ *R1=booked*). The fact that both guards and actions can be used in the same condition helps engineers in expressing functional needs.

For the example given in Figure 3, two pre-conditions and one post-condition are defined for the operation *Fixate A*. The first pre-condition (*Place A==finished*) means that operation *Place A* must be finished before operation *Fixate A* can start. The second pre-condition and the post-condition are related to resource booking. This latter pre-condition means that resource *Fixture* must be available (*Fixture==available*) and then booked (*Fixture=booked*) before operation *Fixate A* can start. The post-condition means that resource *Fixture* is unbooked (*Fixture=available*) when operation *Fixate A* is finished.

In Sequence Planner, the modeling of pre- and post-conditions is based on the FSM model that handles the SOP language. Each operation and each resource is represented through a unique variable. This representation eases the definition of pre- and post-conditions related to operation states, resource operating modes and counters (see section 4.1).

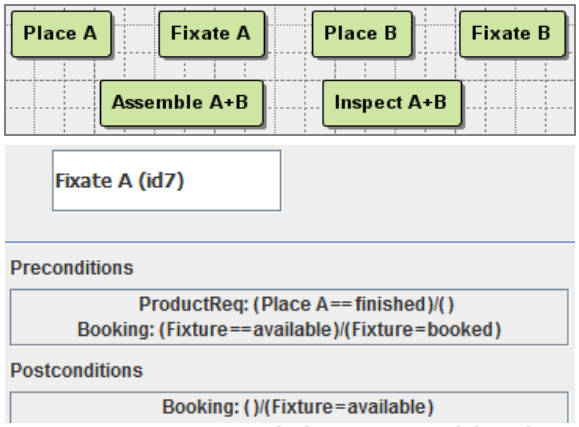

Figure 3. Operations and relations expressed through pre- and post-conditions

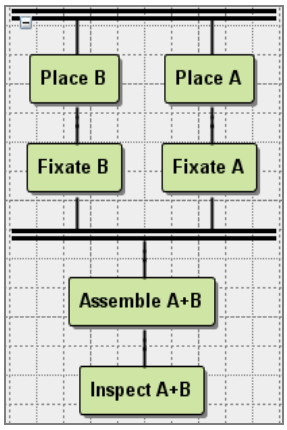

Figure 4. SOP generated according to the conditions on product requirements

2.3. Sequence Planner software

Sequence Planner (SP) is a prototype software tool developed to manage the SOP language and to perform sequence planning [3]. SP handles operations and permits to build Sequences of Operations according to pre- and post-conditions associated to each operation. These sequences of operations can be represented from different points of view. For example, SP can represent SOPs from a *product* point of view (sequences of operations related to one product) or from a *resource* point of view (sequences of operations performed by a specific resource).

To ease SOPs representation, several concepts have been introduced in order to express parallelism, alternatives, arbitrary order, etc. One important concept is *hierarchy*. A hierarchical relation can be used to represent in detail how an operation is performed. This hierarchical representation permits to simplify the representation of an SOP and only display information that are important for the end-user: either a high-level view of a whole system or a low-level view of a part of a system.

The input data of this software are the following:

1. A set of product operations. This set contains operations that should be performed on the product.
2. A set of resources with detailed operations. For each resource, each operation that it can realize is detailed through a hierarchical relation.
3. A resource mapping. This mapping permits to define, for each operation, which resources are able to realize it.

As previously mentioned, since the operations and the resources can be formally defined through a FSM model, supervisory control theory can be applied on the global model. SP is linked to Supremica [9] and permits to generate extra guards that must be added to pre- and post-conditions to avoid forbidden states. These extra guards are generated according to a supervisory model that is defined to ensure system liveness, safety, collision avoidance, etc. For instance, the sequence conditions define when the fixture should close clamps to fixate *Part A* or *B*, and together with interlocking and resource allocation this defines the pre-condition for the close clamp action. The interlocking checks that the resources do not collide with something or someone, while resource allocation is used to manage zone booking or tool allocation.

As a result, the output data of the software are several SOPs that contain the original operations completed with additional guards. These SOPs describe the minimally restrictive behavior of the system, i.e. the behavior with the largest amount of freedom that satisfy pre- and post-conditions and do not lead to blocking or deadlock. These SOPs can be generated from a product point of view (SOP given in Figure 4) or from a resource point of view (SOPs for the human operator, etc.), with different levels of detail, etc.

3. Levels of Automation in an assembly system

There is no simple way to make automation human-oriented and applicable across all domains and types of work. Different processes and domains put different emphasis on flexibility, speed, etc. requiring specific consideration of type appropriate automation [10]. Levels of Automation (LoA) could be defined as *"The allocation of physical and cognitive tasks between humans and technology, described as a continuum ranging from totally manual to totally automatic"* [2].

In order to provide indicators and parameters that can be used to perform and optimize resource allocation (seen from both a cognitive and physical perspective of LoAs) in assembly systems a 7 by 7 matrix has been developed [11], resulting in 49 different possible solutions of varying LoAs. Results from six case studies show that approximately 90 percent of the tasks performed in assembly systems are still performed by humans and by own experience (LoA=(1;1)) [12].

There is a need for a dynamic resource allocation that can take advantage of the access to instantaneous evaluation of the situations to choose the best allocation [13]. A case study that uses dynamic resource allocation, involving changeable LoAs [14], shows that it is possible to change from a human operator to a robot-cell and vice versa in order to achieve volume and route flexibility. The issue to be shown in this paper is how to model and simulate this dynamic allocation when alternative resources could be allocated to some operations.

Difference in LoAs implies that different resources need to be modeled as precisely as possible so that these models correspond to these LoAs and not to a generic resource. Furthermore, models of behavior, knowledge and skills for robots and human must be considered in different ways in order to better fit the real resources.

The aim of the proposed resource modeling is to reduce the gap between a resource and its model, and to take into account human roles in early design phases of an automated system to avoid automation abuse [15].

4. Modeling of human operator and robot resources

In previous works [4], [5], sequence planning has been considered only for the automatic mode of a system. In this context, the resource allocation is often the result of an optimization problem; thus only one resource (or several complementary resources) is allocated to each operation on the product. In this paper, we consider an assembly system in which robots and humans coexist and can cooperate. If we consider the functioning of such an assembly system over a longer period, several unpredicted events can occur (robot breakdown, misassembled products, etc.). These errors imply that the optimal solution found for the automatic operating mode is no longer the optimal one if the system configuration is changed.

The following sections deal with two improvements that can be added to previous systems modeling. The first one considers operating modes of an assembly system; the second one alternative solutions with different LoAs.

4.1. Operating modes

In the automatic operating mode, a resource performs operations on a product. If several products are considered, the resource executes the same operation several times. This modeling means that products and resources are considered in two different ways: the resource performs operations whereas the product needs operations to be performed.

However, over a longer period, maintenance tasks need to be performed. For instance, after a breakdown, a robot needs to be maintained, reprogrammed, set-up, etc. In that case, the robot, which was previously considered as a resource, is now considered as a product, and a human operator performs operations on it.

The SOP language can be used to represent these operating modes. An operation is associated to each mode. Then, using a hierarchy relation, detailed operations are added to each mode according to the resource abilities. For the studied example, five modes can be considered for the robot resource: *Production*, *Unavailable*, *Maintenance*, *Set-Up* and *Ramp-Up*. These operating modes are presented in Figure 5, and a detailed view of the *Production* operating mode is given.

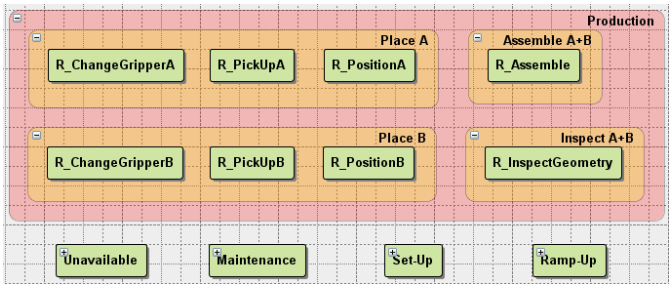

Figure 5. Operating modes of the robot resource

Since the SOP language is used to represent operating modes, relations between the different modes can also be defined through pre- and post-conditions.

For example, if we assume that the robot can assemble *100* products before maintenance is needed, the relations between the operating modes can be defined through the pre- and post-conditions defined as follows:

- post-condition of operating mode *Production* is defined by *count_assemble==100*.
- a part of the pre-condition of operating mode *Unavailable* is defined to be *true* when operation *Production* is finished. This pre-condition is defined by *Production ==finished*.
- a counter *count_assemble* is incremented by 1 each time the operation *R_Assemble* is finished. This implies that a post-condition of operation *R_Assemble* is defined by *count_assemble+=1*.

Besides, the robot can be switched from *Unavailable* mode to *Maintenance* mode if a human operator who has abilities to perform maintenance operations contained in this mode is available. Then, the robot can be switched to *Set-Up*, *Ramp-Up* and *Production*, sequentially.

4.2. Use of resource operating modes

Resources used in flexible assembly systems typically have many abilities. Modeling a resource through several operating modes permits to organize these different abilities according to the operating mode they are related to.

As mentioned previously, a detailed optimal planning is hard to obtain for complex systems and is no longer relevant when an error occurs (resource unavailability, etc.). The hierarchical resource modeling through operating modes can be used to consider sequence planning from two hierarchical levels:

- A detailed optimal planning for each production mode: For example, an optimal sequence planning for the robot resource when it is in its *Production* mode. This planning takes into account detailed operations (*R_PickUpA*, *R_PositionA*, etc.)
- An organizational planning considering only the different operating modes. For example, the robot will produce 100 products being in its *Production* mode, and then a human operator will be needed to perform maintenance in the *Maintenance* mode. Once *Maintenance* is finished, the robot can switch to *Set-Up*, *Ramp-Up*, and then *Production*.

This hierarchical representation can also be used to define policies that must be applied to each mode. For instance, in the *Set-Up* or *Ramp-Up* modes, human operators and robots may use the same area; this means that the safety policy must be on its highest level [16]. On the opposite, in the *Production* mode, the time policy can be the leading policy. Moreover, according to the current operating mode the control code implemented in the robot can be slightly different. For example, in the *Production* mode, since the robot acts automatically extra guards can be added by the supervisory controller in order to take into account collision avoidance with other robots. On the other hand, in the *Set-Up* mode, the robot should execute step-by-step; extra guards can be added so that human validation (through a push button) is needed between two steps. Other policies such as deadlock avoidance, energy efficiency, etc. can also be applied to the different modes.

The same approach can be applied to a human operator resource. The operating modes can be different for each human operator since they all have different abilities. In that case, the objective of operating modes is to define what type of activities the human operator is performing: mental activities, manual activities, production tasks, maintenance tasks, pause, etc. These modes can then be used, for example, to conduct an optimization according to a human operator's workload.

4.3. Levels of Automation in Sequences of Operations

Robots and human operators can sometimes have the same abilities, but in many cases their abilities and capabilities are different (in this paper capabilities are defined as detailed abilities, quality parameters, maximum load, etc.). If they do not have the same abilities, then the matching between resources and operations is simplified. If several resources can perform the same operation, the best one (according to an optimization criterion) is allocated to each operation. However, the result of an optimization problem depends on hypotheses. If these hypotheses are no longer satisfied the whole optimization problem must be reconsidered, and this takes time. In many cases, if a robot breaks down an easy solution is to use human operators as a replacement resource. Unfortunately, if the different alternatives have not been considered during the process design the reallocation of operations is not easy to perform. To tackle this issue, we suggest generating SOPs that keep track of alternatives according to different LoAs.

In what follows, only the *Production* operating mode will be considered. For the example illustrated in Figure 1, the operations presented in Figure 4 are performed using three resources: a robot *R*, a fixture *F* and a human operator *H*.

The fixture *F* can perform operations *Fixate A* and *Fixate B*.

The robot *R* can perform operation *Place A*, *Place B*, *Assemble A+B* and *Inspect A+B* (see Figure 5).

The human operator *H* can perform operation *Place A*, *Place B*, *Assemble A+B* and *Inspect A+B* (see Figure 6).

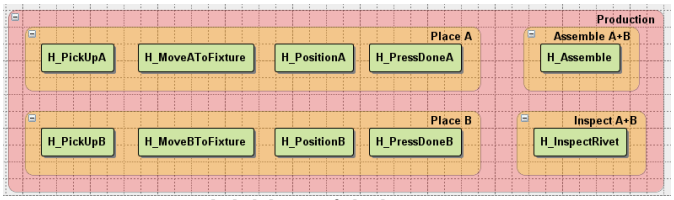

Figure 6. Detailed abilities of the human operator resource

As presented in Figure 5 and Figure 6, even though the robot and the human operator have exactly the same global abilities their detailed abilities differ. Even though both can *Place A*, *Place B* and *Assemble A+B*, they do not perform these operations with the same precision. Thus, they have the same abilities but different capabilities. The following statements illustrate these differences and their consequences:

1. If the human operator performs *Place A* or *Place B*, the geometry of the resulting assembly must be inspected. The robot is supposed to be accurate and repeatable enough so that geometry inspection is not required when both *Place A* and *Place B* are performed by the robot.
2. If the robot performs *Assemble A+B*, the rivets quality must be inspected. If *Assemble A+B* is performed by the human operator, we assume that rivets quality inspection has already been done by this operator during assembly.
3. Geometry inspection is performed by the robot. This inspection is done automatically using a contact sensor.
4. Rivets quality inspection is based on visual inspection. This inspection is performed by the human operator resource.

Statements 3 and 4 are used to define the robot ability *R_InspectGeometry* and the human operator ability *H_InspectRivet*. Statements 1 and 2 are instantiated in the pre-conditions of operations *R_InspectGeometry* and *H_InspectRivet*.

These four statements illustrate an example of an assembly system that requires two LoAs. Neither the human operator nor the robot can perform the whole assembly alone. However, several alternatives are possible, as illustrated in the next section.

5. Simulation and results

Sequence Planner has been used to generate and represent the different alternatives that permit to produce the product shown in Figure 1.

The operations that must be performed have been modeled as shown in Figure 3. Only the necessary conditions have been used to define pre- and post-conditions. The three resources previously presented have been modeled in the software with their detailed abilities, as shown in Figure 5 and Figure 6, using hierarchical relations.

Figure 7 presents an SOP from a product point of view. Since both the robot and the human operator can perform operations *Place A*, *Place B* and *Assemble A+B*, alternatives are proposed for these operations. Alternative sequences are represented by a single horizontal line. Since *Place A* and *Place B* can be performed by different resources (e.g. *Place A* performed by the robot and *Place B* performed by the human operator), they can be executed in parallel. Parallel execution is represented by double horizontal lines. Dashed horizontal lines on top and bottom of operations *H_InspectRivet* and *R_InspectGeometry* mean that these operations are order independent, i.e. *H_InspectRivet* can be executed before or after *R_InspectGeometry*.

However, this graphical representation is not sufficient to describe precisely and unambiguously the assembly system behavior. Pre- and post-conditions associated to each operation permit to define more precisely this behavior. For instance, according to Figure 7, *Fixate A* and *Fixate B* could be executed in parallel; however since they are performed by the same resource (fixture *F*) they cannot be executed in parallel. Pre-conditions of operations *Fixate A* and *Fixate B* permit to define this through resource booking conditions.

The SOP given in Figure 7 does not express an optimal sequence planning but the minimally restrictive behavior of the considered assembly system. This SOP permits to represent alternatives between a high LoA (operations performed by the robot) and a low LoA (operations performed by the human). If possible, alternatives are represented locally; these local alternatives permit to define local triggers to define if the human or the robot should perform operations that follow. The definition of these triggers is not unique. One solution is to consider that the first resource available performs the operations of the alternatives. Another solution is to define additional guards according to the solution of an optimization problem. Since many optimization problems can be defined (global time minimization, resource workload minimization, human workload adjustment, stock in process minimization, route flexibility maximization, etc.), generation of an optimal planning is not considered in this paper. However, the obtained SOP can be used as an input to an optimization problem, since costs can be attached to each operation.

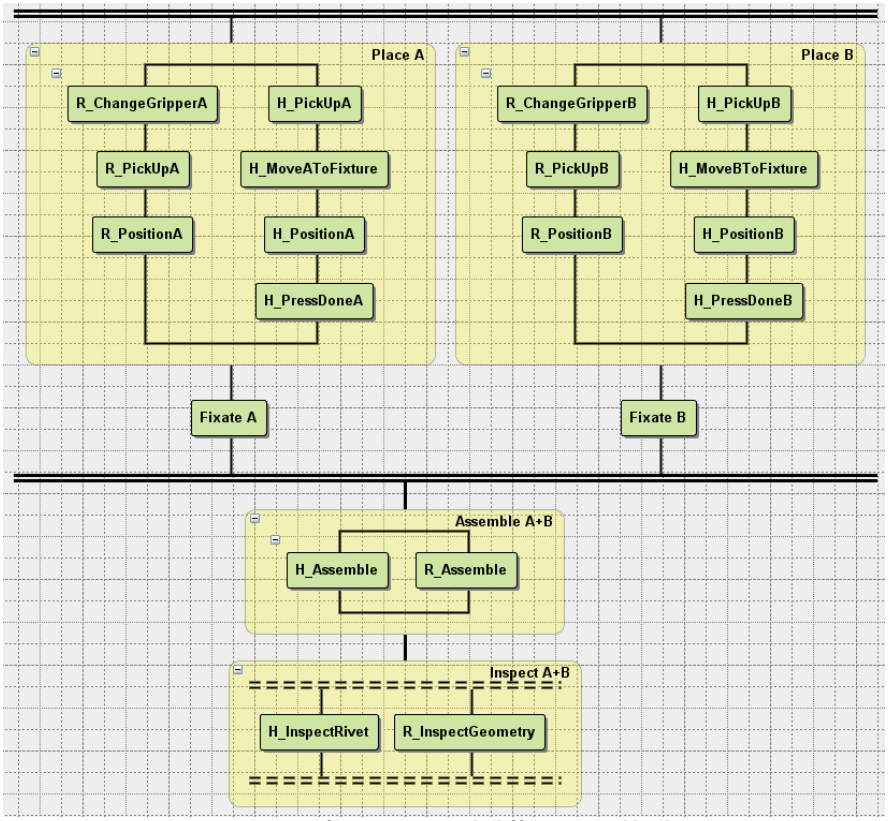

Figure 7. Sequence of Operations with different possible alternatives

6. Conclusion and prospects

This paper has presented a way to consider both human operators and robots as specific resources with their own abilities and capabilities. The modeling language used to define and represent these resource models is flexible and permits to express many relations between operations and resources. The SOP automatically generated permits to represent all alternatives to assemble a product without deadlock.

On-going work considers three topics:

1. Automatic generation of control code for robots and instructions for human operators. The automatic generation of instructions for human operators would aim at reducing the learning time and improving product quality.
2. Sequence planning optimization with regard to flexibility and proactivity.
3. Definition of timed and stochastic models. These models would permit to implement criteria such as Mean Time To Failure, Mean Time Between Failure, etc. and could be used to define relations between operating modes.

References

[1] Marian, R.M., Luong, L.H.S., Abhary, K., 2003, Assembly Sequence Planning and Optimisation using Genetic Algorithms: Part I. Automatic Generation of Feasible Assembly Sequences, Applied Soft Computing, 2(3), p. 223-253.

[2] Frohm, J., Lindström, V., Winroth, M., Stahre, J., 2008, Levels of Automation in Manufacturing, Ergonomia IJE&HF, 30(3).

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

[4] Lennartson, B., Bengtsson, K., Yuan, C. Y. et al., 2010, Sequence Planning for Integrated Product, Process and Automation Design. IEEE Transactions on Automation Science and Engineering, 7 (4), p. 791-802.

[5] Bengtsson, K., Lennartson, B., Yuan, C., 2009, The Origin of Operations: Interactions between the Product and the Manufacturing Automation Control System, 13th IFAC Symposium on Information Control Problems in Manufacturing, INCOM 2009, p. 40-45.

[6] Mounaud, M., Thiebaut, F., Bourdet, P., Falgarone, H., Chevassus, N., 2009, Assembly Sequence Influence on Geometric Deviations Propagation of Compliant Parts, 11th CIRP Conference on Computer Aided Tolerancing.

[7] Wang, H., Ceglarek, D., 2005, Quality-driven Sequence Planning and Line Configuration Selection for Compliant Structure Assemblies, CIRP Annals - Manufacturing Technology, 54(1), p. 31-35

[8] Shoaei, M. R., Lennartson, B., Miremadi, S., 2010, Automatic Generation of Controllers for Collision-Free Flexible Manufacturing Systems, 6th IEEE Conference on Automation Science and Engineering, CASE 2010, p. 368-373.

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

[10] Hollnagel, E., 2003, The Role of Automation in Joint Cognitive Systems. 8th IFAC Symposium on Automated Systems Based on Human Skill and Knowledge.

[11] Fasth, Å. & Stahre, J., 2010, Concept Model towards Optimising Levels of Automation (LoA) in Assembly Systems, 3rd CIRP Conference on Assembly Technologies and Systems.

[12] Fasth, Å., Stahre, J., Dencker, K., 2010, Level of Automation Analysis in Manufacturing Systems, 3rd International Conference on Applied Human Factors and Ergonomics.

[13] Hoc, J.-M., 2000, From Human-Machine Interaction to Human-Machine Cooperation. Ergonomics, 43, p. 833-843.

[14] Fasth, Å., Stahre, J., Dencker, K., 2008, Analysing Changeability and Time Parameters due to Levels of Automation in an Assembly System, 18th Conference on Flexible Automation and Intelligent.

[15] Parasuraman, R., Riley, V., 1997, Humans and Automation: Use, Misuse, Disuse, Abuse, Human Factors, 39 (2), p. 230-253.

[16] Tan, J.T.C., Duan, F., Zhang, Y., Kato, R., Arai, T., 2009, Safety Design and Development of Human-Robot Collaboration in Cellular Manufacturing, 5th IEEE Conference on Automation Science and Engineering, CASE 2009, p. 537-542.