# A FRAMEWORK BASED ON A HIGH CONCEPTION LEVEL TO GENERATE CONFIGURATIONS IN PRODUCTION SYSTEMS

Marwa Kanso, Pascal Berruet and Jean-Luc Philippe
*Lab-STICC, UBS Saint Maude, Lorient, France*

Keywords: Reconfigurable manufacturing systems (RMSs), Model Engineering, Scheduling, Configuration, Recovery & reconfiguration.

Abstract: In this paper, a framework is presented to generate configurations in reconfigurable manufacturing systems (RMS). This framework is based on a high conception level of reconfigurable manufacturing systems, and uses a multicriteria decision algorithm to select operations to carry out the request. Provisional product scheduling is then used to define the "simple configuration". This paper focuses on the generation process of the "near-most appropriate" configuration to carry out a request. To do so, the same multicriteria decision algorithm is used to select "reserved operations" to improve the generated configuration.A case of study illustrates our approach and demonstrates how we can use this framework to generate the "near-optimal" configuration and improve reconfigurability.

## 1 INTRODUCTION

Manufacturing flexibility is the key for markets facing increasing client demands, frequent volume changes and product requirement. In such a system, choices at the system organization level can be delayed until exploitation. Indeed, this organization must be taken into account at the conception level. To help a designer choose (resources, operations, ...), a set of analysis and evaluations must be applied to the system. The system description must specify all possible information for this analysis. New scheduling techniques which help to answer more complex demands are needed. On the other hand, the main goal in today's markets is the reaction of manufacturing systems when unexpected events occur. Recently, new ideas related to design, description, sequencing products, planning, loading and scheduling policies have been introduced. Architecture design for manufacturing systems, equiped with the correct level of flexibility to face the specific production problem, are introduced in (Nucci and Grieco, 2008) and in (Terkaj et al., 2009). (Kurnaz et al., 2005) considers that the order in which products are produced can have a considerable impact on primary performance metrics. Sequencing decisions can impact customer responsiveness. In (Dpto et al., 2002), loading and scheduling processes evolves jointly in Felxible Manufacturing Cell (FMC). (Lamotte, 2006) proposes DeSyRe, a language to describe reconfigurable manufacturing systems. In this paper, we propose improving the DeSyRe language to take into account and improve the reconfiguration concept at the description level. We also consider how disruptions (machine breakdowns, customer order changes, etc.) impact the generated configuration in a reconfigurable manufacturing system producing multiple products. The rest of this paper is organized as follows. Section 2 gives an extended presentation of the high conception level of RMS's. Section 3 presents the configuration generation process. A use case illustrates the use of our framework in section 4, before moving on to the conclusion in section 5.

## 2 A HIGH CONCEPTION LEVEL FOR RMS

Manufacturing systems can generally be described from multiple levels of granularity. Fractal manufacturing systems (Ryu et al., 2003) takes into account this granularity by using a multi-level representation. To represent and manipulate reconfigurable manufacturing system concepts, we use a description language called "DeSyRe" basically developed and introduced by (Lamotte, 2006). Metamodels show relations between different aspects of a RMS. Models



Kanso M., Berruet P. and Philippe J. (2010).
A FRAMEWORK BASED ON A HIGH CONCEPTION LEVEL TO GENERATE CONFIGURATIONS IN PRODUCTION SYSTEMS.
In *Proceedings of the 7th International Conference on Informatics in Control, Automation and Robotics*, pages 244-248
DOI: 10.5220/0002943102440248

combine a horizontal decomposition between architecture and configuration and a vertical one between logical and physical descriptions. We apply this high level conception to add new dynamic aspects and improve the RMS description. In the next subsections, we present the extended description language version "DeSyRe_E".

## 2.1 The represented Aspects

In "DeSyRe_E", the reconfigurable system is broken down according to the decomposition illustrated in Figure 1. A horizontal axis separates the architecture of the system from its configuration. The architecture consists of all system elements (functions, or resources) and their potential connections. It is presented in section 2.2. These components are parameterized and inter-connected through the configuration presented in section 2.3. The vertical axis separates the logical architecture from the physical one. The logical part consists in the functions and their associations to form logical operating sequences. The physical part consists in the resources and the transport between these resources. The physical part provides the structure on which the logical part is executed. Operations are in the center of the model description. Each operation implements a function on a resource creating a link between the logical and the physical architecture. It also links the configuration to the architecture since operations are defined in the architecture and used in the configuration.

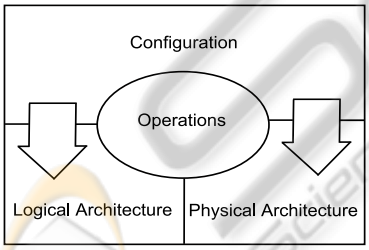


Figure 1: The new decomposition of RMSs.

## 2.2 A Meta-model to represent Architectures

The architecture of a reconfigurable manufacturing system should represent all system potentialities (see figure 2). It is separated into two parts: the logical architecture, which is constituted by functions and function sequences applied to products that can be performed on the system, and the physical architecture which describes the physical elements of the system (resources, connections, ports, etc...). To achieve a complete architecture representation, operating modes of the whole system should be provided. The role of such representation is to help carry out decisions about resource modes and configuration changing. A lot of information is needed to manage resource modes, see (Hamani et al., 2009), and perform performance and cost analysis. This information is introduced using a simple diagram graph for each resource to describe their modes and transitions between these modes. For the sake of simplicity, here, we propose to take into account the following modes: the mode "stop" when the power is off, the mode "in production" when the resource is in use, the mode "in preparation" when the resource is not in use, nor ready to execute an operation, the mode "break down" when a failure occurs on the resource and the mode "idle" when the power is on and the resource is ready to use.

Logical architecture is mapped onto physical architecture through potential operations, which link a function with a resource.

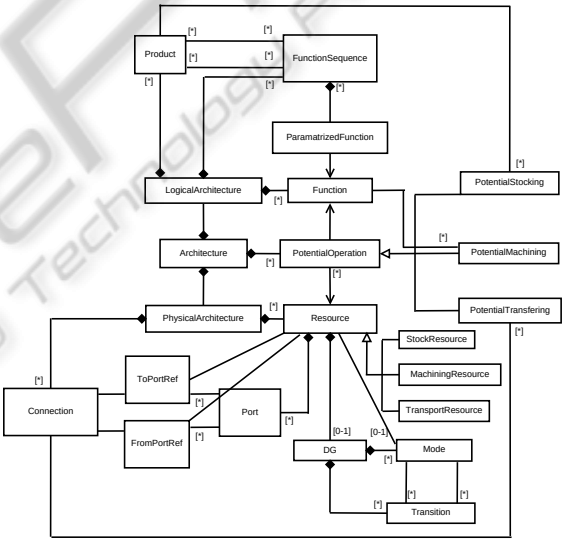


Figure 2: Extended architecture metamodel.

## 2.3 A Meta-model to represent Configurations

The concept of configurations lay on the separation of the component content from its use in the architecture. A configuration is constituted by function instances to realize a machining operation, connection instances to realize a transfer operation, operations which refer to an operation defined in the architecture model and operation sequences which realize a function sequence. The metamodel representing the configuration is given in figure 3.

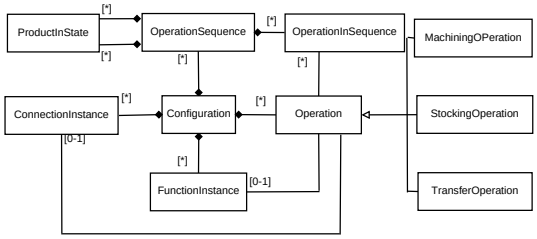


Figure 3: Extended configuration metamodel.

Several types of configurations are defined depending on the use of architecture components. A minimal configuration contains the needed operations to carry out a mission. A simple configuration is defined as the set of "used" operations which lead to the completion of a mission. A flexible configuration is defined by a set of "used" and "reserved" operations to simplify the reconfigurability aspect. Finally, a maximal configuration is defined as the set of potential operations which lead to carrying out a mission. In the next section we present the Configuration Generation module.

# 3 CONFIGURATION GENERATION

As can be seen in the figure 4, three technologies are used: Model driven engineering (MDE), object oriented programmation and multiCriteria Decision Making (MCDM). Modeling concepts are introduced to describe architectures and configurations. The Model Driven engineering approach (MDE) is then used, a transformation module is used to generate the maximal configuration from the architecture description model. This transformation is expressed in a transformation language such as ATL (Bézivin et al., 2005).

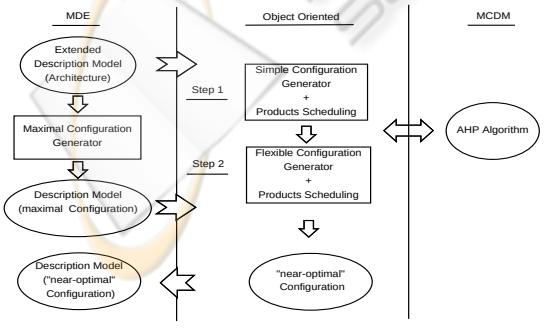


Figure 4: A framework to generate configurations.

Generated code is based on the architecture and maximal configuration description models. Object oriented development (using Java) is then used to carry out the "near most-appropriate" configuration. At runtime, both steps (1 and 2) use the MultiCriteria Decision Making approach (MCDM) to solve indeterminism due to the high flexibility of our system and to carry out the request in the configuration. For more details about the MCDM approach see (Kanso et al., 2009). A simple configuration is generated in step 1. The obtained configuration contains only the "used" operations to carry out the mission. It means that, in case of failure there is no garantee to replace the failed operation. Our framework tries to avoid the reconfiguration process as much as possible, by improving the generated configuration with a compatible degree of flexibility. We deal with two kinds of unexpected events: changes at the request level (case of customer demand changes) and changes at the system level itself (case of failures and system errors). When unexpected events occur, errors, failures or customer demand changes provide the need of additional operations to replace the failed ones. Operations in need are evidently defined in the architecture. To answer this need, a flexible configuration is generated in step 2. This last configuration consists of the simple configuration improved by adding "reserved" operations using the MCDM approach. The obtained configuration description model is then generated using MDE.

# 4 SAMPLE EXPERIMENTATION

We present, in this section, a simple example to demonstrate how our framework can be used to generate the "near optimal" configuration. The purpose of this experimentation is not to generate optimal suggestions, but rather to provide a simple demonstration to help users taking their decisions during production. To do so, we consider a simple reconfigurable manufacturing system with two products. The "logical architecture" contains two functions ($F_1$ and $F_2$) and three function sequences ($S_1$, $S_2$ and $S_3$). $S_1$ consists in implementing $F_1$. $S_2$ consists in implementing $F_2$. $S_3$ consists in implementing $F_1$ then $F_2$.

The "Physical Architecture" of the proposed system is defined to set up our experiments (see figure 5). It consists of convoyers ($CV_1$ - $CV_6$), buffers ($IN$ and $OUT$), machining areas ($M_1$ - $M_4$) and a robot ($Rbt_1$). To increase the flexibility of the system, this architecture consists of the two-directional convoyers ($CV_2$ and $CV_5$). In the description model of the architecture, we refine, for each resource, its capacity, its active mode, the list of the concerned resource modes and the transitions between these modes.

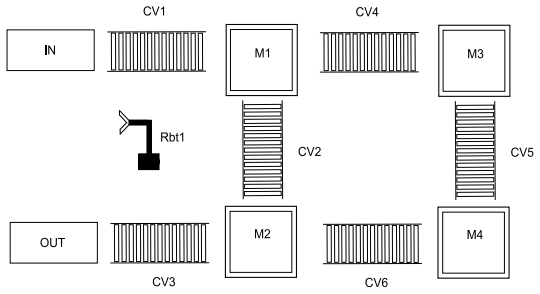


Figure 5: Physical Architecture of the proposed system.

Eight machining operations have been defined to assign the defined functions to the machining areas: $O_{M1,F1}$, $O_{M1,F2}$, $O_{M2,F1}$ and $O_{M2,F2}$, etc...

Fourteen transfer operations have also been defined to realize a transfer using a transfer ressource: $O_{t_{Cv1,INM1}}$, $O_{t_{Rbt,INM1}}$, $O_{t_{Rbt,INM2}}$, $O_{t_{Cv2,M1M2}}$, $O_{t_{Cv2,M2M1}}$, etc...

## 4.1 Maximal Configuration Generation

In this subsection, we present the transformation module used to generate the "maximal configuration" using the architecture description model. Based on the specified mission (product types and quantities), some operations cannot be used in the system. The concerned resources are idle during production. To optimise the configuration cost, we must generate the "maximal configuration". At this stage, we take into consideration the system part which contains the possible operations to carry out the request. If all potential operations can be used to realize our mission, the maximal configuration represents the use of the whole system. For reasons of simplicity, in our example, the maximal configuration represents the use of the whole system to carry out two different product types ($P_4$ and $P_5$). Each operation in the architecture model is projected to generate an operation in the configuration model. The link to a function instance or a connection instance in the configuration model is based on the type of the operation in the architecture. A function sequence can be realized by different operation sequences. For example, $S_1$ is the function sequence which consists of applying the function $F_1$ on a product. It can be realized on the machining resource $M_1$, $M_2$, $M_3$ or on $M_4$. These four solutions are defined in the configuration model as a set of 88 operation sequences, since transfer may be realized by different transfer sequences too. Indeed, we generate 674 possible operation sequences based on the function sequences defined in the architecture model.

It should be noticed that by using all the operations and operation sequences of the system in the configuration at runtime, the reconfiguration time, in case of failure can be shorten (it depends on calculation and decision making time), and the cost of the configuration is surely much higher since a lot of resources are idle. Therefore, we propose to generate the "near-optimal" configuration which answer a better cost-performance trade-off. In the next subsection, we refine the "Simple Configuration generation" process based on the architecture and the maximal configuration description models.

## 4.2 Simple Configuration Generation

The specified mission consists of producing three lots of $P_4$ and two lots of $P_5$. Decisions, of which transporter (or machining resource) is used, are taken by applying the 'AHP' algorithm. For more details about the proposed algorithm please check (Kanso et al., 2009). The "Simple Configuration Generation" process consists of selecting, for each product instance, an operation sequence taken into account the system state (resource availability, waiting time, preparation time, mode changes, transfer between two different machining resources). We use the necessary information in the architecture to decide which is the most appropriate operation to realize next. This process has been implemented using Java in eclipse environment. Once the application is launched, the selection process starts. The architecture of the proposed system is updated depending on each resource capacity and modes in the system while executing an operation on a product. The "simple configuration" is obtained when each product instance specifies its operation sequence and its provisional scheduling. We define at this level the "used operation" as an operation used by a product to realize a function in a configuration. For the proposed example we get the configuration composed of the followed operation sequences:

1. to realize $P_4$, $S_1$ must be applied. Therefore, the operation sequence chosen is as follows:
   - $OS_{1,S_1}$: $\{O_{t_{Cv1,INM1}}, O_{M1,F1}, O_{t_{Rbt,M1OUT}}\}$
2. to realize $P_5$, $S_3$ must be applied. Therefore, the operation sequence choosed is as follows:
   - $OS_{12,S_3}$: $\{O_{t_{Cv1,INM1}}, O_{M1,F1}, O_{M1,F2}, O_{t_{Rbt,M1OUT}}\}$

## 4.3 Flexible Configuration Generation

In case of production problems and more precisely in case of failure when unexpected events occur (demand changes, resource breakdowns, unavailable tools, etc...). Previously generated simple configurations may no longer work. Operations used in the "simple configuration" may be unique and no another

operation, defined at the configuration level, can realize the same function. The obvious solution is to reconfigure the system. Although, we propose to improve the generated configuration and optimise the request completion time. Therefore, we need to extend the "simple configuration" and add "reserved operations". At this level, we define a "reserved operation" as an operation chosen to replace a failed operation in a configuration. This will help us carrying out the request in the system without going through the reconfiguration process. To do so, "reserved operations" are selected using the 'AHP' algorithm. When the "reserved operations" are specified, we generate the corresponding operation sequences to fill out the generated configuration. In other words, we generate reserved operation sequences to extend the obtained "simple configuration" and get the "flexible configuration" to improve system responsiveness. Projecting this extension on our example, we get the "flexible configuration" by duplicating operations since each function (or transfer) can be realized using four resources ($M_1$ - $M_4$) for a function, and the robot or a convoyer for a tranfer). We note that in our case the obtained "flexible configuration" corresponds to "162" operation sequence of the "maximal configuration" using two machining resources ($M_1$ and $M_2$) and "used operations" are realized in priority.

## 5 CONCLUSIONS

Reconfigurable Manufacturing Systems are faced with frequent disruptions that impact the production quality. A low-cost solution is needed to recover the system. In this paper, the use of a high conception level for reconfigurable manufacturing systems is presented to include different features and describe both architecture and configuration. The suggested framework produces "near-optimal " configuration by generating the corresponding "flexible configuration" based on the generated "simple configuration" and the architecture. A simple example illustrating the way we can use the proposed framework is presented in section 4. First, the production environment has been presented. Secondly, the "maximal configuration" generation is presented by refining the transformation module. Third, the "simple configuration" generation process is described before moving on to presenting the "flexible configuration" generation process. Configuration choices usually depend on preferences, on choices obtained at the scheduling level, and on constraintes defined by clients at the request level. "Minimal" and "Simple" configurations are not always the best solutions. Configurations with a higher degree of flexibility may respond to requests with better measures, and may answer a better cost-performance trade-off. Although the existing framework provides opportunity for many different types of analyses, additional extensions will be beneficial as well. These include inventory management during production and transfer sequence management; new configuration extension rules; and more specific criteria to improve operation choice. More generally, we would like to permit a broader class of configurations, such as serial parallel lines with crossover, so that the framework can be applied in an even greater number of circumstances.

## ACKNOWLEDGEMENTS

The authors are pleased to acknowledge the support by the associate professor at Polytech'Marseille Fouzia Ounnar.

## REFERENCES

Bézivin, J., Jouault, F., Rosenthal, P., and Valduriez, P. (2005). The AMMA platform support for modeling in the large and modeling in the small.

Dpto, A. G., Gmez, A., Fuente, D. D. L., Parreo, J., and Puente, J. (2002). Scheduling in flexible manufacturing systems. *Applied Artificial Intelligence*, 15:949–963.

Hamani, N., Dangoumau, N., and Craye, E. (2009). Reactive mode handling of flexible manufacturing systems. *Mathematics and Computers in Simulation*, 79(5):1421–1439.

Kanso, M., Berruet, P., and Philippe, J. (2009). Multicriteria decision making approach for reconfigurable manufacturing systems. *In proceeding ISBN of the 28th EAM Conference on Human Decision-Making and manual Control, SEPT. 3-4, 2009*, pages 37–44.

Kurnaz, S., Cohn, A., and Koren, Y. (2005). A framework for evaluating production policies to improve customer responsiveness.

Lamotte, F. D. (2006). *Proposition d'une approche haut niveau pour la conception, l'analyse et l'implantation des systmes reconfigurables 2006*. PhD thesis. Universit de Bretagne Sud, Lorient, France.

Nucci, F. and Grieco, A. (2008). The operational strategies in focused flexible manufacturing systems. *ISC'08 International Supercomputer Conference. Heidelberg, Germany, June 18 20.*

Ryu, K., Son, Y., and Jung, M. (2003). Modeling and specifications of dynamic agents in fractal manufacturing systems. *Comput. Ind.*, 52(2):161–182.

Terkaj, W., Tolio, T., and Valente, A. (2009). Designing manufacturing flexibility in dynamic production contexts. In *Design of Flexible Production Systems*, pages 1–18.