HAL
archives-ouvertes.fr

Product Specification in a Service-Oriented Holonic Manufacturing System using Petri-Nets

Francisco Gamboa Quintanilla, Sylvain Kubler, Olivier Cardin, Pierre Castagna

▶ **To cite this version:**

Francisco Gamboa Quintanilla, Sylvain Kubler, Olivier Cardin, Pierre Castagna. Product Specification in a Service-Oriented Holonic Manufacturing System using Petri-Nets. Intelligent Manufacturing Systems, 2013, São Paulo, Brazil. 10.3182/20130522-3-BR-4036.00094 . hal-01116281

HAL Id: hal-01116281
http://hal.univ-nantes.fr/hal-01116281

Submitted on 12 Feb 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Product Specification in a Service-Oriented Holonic Manufacturing System using Petri-Nets

Francisco Gamboa Quintanilla*, Sylvain Kubler, Olivier Cardin*, Pierre Castagna***

** LUNAM Université, IUT de Nantes – Université de Nantes, IRCCyN UMR CNRS 6597 (Institut de Recherche en Communications et Cybernétique de Nantes), 2 avenue du Pr Jean Rouxel – 44475 Carquefou (e-mail: francisco.gamboa/olivier.cardin/pierre.castagna@irccyn.ec-nantes.fr)*

*** Research Centre for Automatic Control of Nancy, Campus sciences, BP-70239, F-54506 Vandœuvre-lès-Nancy Cedex, France (e-mail: sylvain.kubler@cran.uhp-nancy.fr)*

Abstract: Service-Oriented Holonic Manufacturing Systems are now well known and widely studied. Every reference architecture of literature contains a holon dedicated to the description of the recipes of products to be manufactured. Typically, this description is a list of services to perform in order to obtain a finished good out of raw materials. This paper introduces an innovative way to describe product recipes using Petri nets. This description enables multiple variant recipes to increase flexibility in a Service oriented Holonic Manufacturing System.

Keywords: Petri-nets, Holonic Manufacturing System, Mass Customization, Service Oriented Architecture, Manufacturing Recipe.

1. INTRODUCTION

Mass customization (MC) refers to a business strategy that combines two different business practices: mass production and craft production. The MC concept is relatively fresh in international business, first discussed by (Davis, 1987). Its development lagged behind because customer needs did not have effective means, i.e. technology, to be expressed and reached by product and service manufacturers. In the recent decade, changing economic and social environments gave the push for the demands of individualized products and services. Companies are now becoming more and more customer-centric. The major objective of MC is to improve the ability of companies to react faster to changing customers' needs and to address the heterogeneity of demand more efficiently. Nonetheless, the MC concept requires new approaches due to the small volume/high variety order management and the maximization of the profitability of the firms (Blanc et al., 2008). (Molina et al., 2005) argue in this sense that the next generation manufacturing systems must therefore be able to provide increased levels of flexibility, re-configurability and intelligence to allow them to respond to product variety. These concerns are challenging the Intelligent Manufacturing Systems (IMS) community from two decades, through a worldwide industry-led research aiming at setting into practice agile Business to Manufacturing systems based on a networked heterarchy of autonomous units. This type of organization appears as more suitable to meet robustness to disturbances, adaptability to rapid changes and efficient use of available resources, which are still weak points of conventional manufacturing systems (Morel et al., 2003). The difficulty is to combine the ability and the capacity to communicate of these units so that they can interact by creating and executing manufacturing plans to process both physical and informational customized goods. A way to express this challenge is to combine the Multi Agents System (MAS) paradigm with the Holonic Manufacturing Systems (HMS) paradigm (Valckenaers, 1998). Recent years have witnessed many proposals (Babiceanu et al., 2006). These paradigms require giving abilities to the products to interact with their environments such as data storage or decision-making abilities (e.g. products may achieve their own routing/re-routing through the supply chain) (Sallez et al., 2009).

Based on these concepts, this paper explores an approach using product family design in order to generate a feasible production process based on a process description allowing multiple variants. Section 2 introduces product and process family designs. Section 3 introduces the notion of services adapted to manufacturing systems. Finally, section 4 presents the main proposal to use Petri-Nets in order to represent product recipes, and its application to a simple study case.

2. MANUFACTURING PRODUCT SPECIFICATION

2.1 Product Customization

The main objective of product customization is to increase a company's product variety offer as to increase its attractiveness and therefore sales. A first step is to define the product offering's customization level. For the context Product Driven Production Systems (PDS), two types of customization will be considered: scalable and modular.

Scalable customization creates variants by varying the capacity of a certain quantifiable product feature (Simpson et al., 2001) e.g. a computer's hard disk capacity. Modular customization, on the other hand, creates variants through the configuration of existing modules (Meyer et al., 1997b) which are structural features of the product. As a result of product customization, product variety increases. Unfortunately, as product variety increases the law of diminishing returns means the benefits obtained in terms of sales do not keep pace. This is due to an increase of internal complexity (Child et al., 1991). Such problem implies a variety management issue that companies must cope by optimizing its external variety with respect to the internal complexity resulting from product differentiation. (Tseng et al., 1996)

As a solution to this complexity problem and to achieve economy of scale, product families' development has been well recognized and adapted by companies as a means to optimize internal complexity and external variety (Meyer et al., 1997a). Added to this, product family design recognizes the existence of scalable and configurable product family platforms (Jiao et al., 2007) which adapt properly to the level of customization addressed in this paper.

2.2 Product Families

According to (Suh, 2001), Product Family Data design encompasses five design domains. Fig.1 presents the fundamental issues encountered while designing product families (Suh, 2001). The first stages of product family design, from customer needs identification all the way up to the design parameters, correspond to activities out of the scope of this paper. However, this paper addresses particularly the back-end issues corresponding to the process and logistics domains. The former refers to the design of processes and their respective variables for the realization of the Design Parameters (DPs). The later domain covers the production logistics aspects i.e. production chain configuration, resource allocation, etc. An approach will be proposed for the mapping of DPs, into the manufacturing domain (process + logistics domains) in a form that will facilitate the exploration of all production alternatives for the realization of a derived family member.

Product family refers to a set of individual products that share a set of common structural characteristics and yet are differentiated one another by certain specific features (Meyer et al., 1997b). Each of these individual products, derived from a product family, is referred as a product variant or product family instance. Product families are based on the commonality that exists between the product variants that can be derived from them. It is this communality that entails the difference between the architecture of product families from that of a single product. On the other hand it is modularity that allows the characterization of product variants. Through modularity, product structures/architectures can be split into modules. Such modules, representing physical or conceptual grouping of components sharing some characteristics, can then be used as building blocks to form a product respecting certain architectural rules. What is important in characterizing modularity in product families is the interaction among modules. This means that, for their modular aspect to be feasible, there is a need of integrity among modules. Such integrity refers to the standardization of module interfaces and the specification of architectural rules within the product platform.

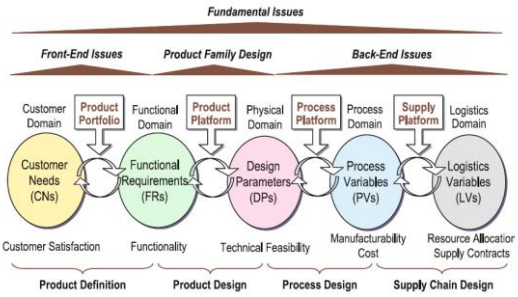

Fig. 1. Holistic view of product family design and development (Suh, 2001).

In summary, product variants are constituted of a collection of modules representing structural features with a certain configuration according to product architectural rules. Such rules are implicitly defined by the modules' interfaces. It is the principle of reutilization of proven and standardized elements/modules that engenders significant benefits to companies such as: reduction in component inventory, ease on component type handling, reduced development risks and faster development time (Fisher et al., 1999).

2.3 Process Families

According to the ideas presented in (Martinez et al., 2000) and (Schierholt, 2001), the similarity found in the products' structure translates into a similarity of operations, processes and sequences among the different product family members. Thus, a common product structure and a common process structure exist within a product family data (Jiao et al., 2007). This common process structure will be referred as a process family. Process families possess the same characteristics/attributes as product families in terms of commonality, modularity, reutilization and scalability (Jiao et al 2007) (Meyer et al 1997a) (Simpson et al 2001). A process family is therefore a collection of manufacturing tasks that respond to the realization of the corresponding feature modules of the product modular architecture. By decoupling the production process of a certain product variant into manufacturing tasks and relating those tasks to the corresponding structural modules of product families the reutilization principle can be then translated to the process domain. Such manufacturing tasks can be standardized and reutilized for the production of other products giving the following advantages:

- Enhanced responsiveness. Faster time to production once the product is developed as the required manufacturing tasks realizing the existing product features are already available and validated..
- Reduction in process variety due to manufacturing tasks reutilization.

- The decoupling of the manufacturing process allows the option of sequence reconfiguration which can be exploited by the production control system.

2.4 Product Modelling

Product family data design comprises several challenges. First, the organization of product data, instead of being a collection of individual product variants should explicate the relationships between the variants. Second, an individual product variant should be defined in terms of the parameters of the product family data (Jiao et al., 1998). This last means that the generation of the specific description of a product variant is a function of both a customer specification and a product family description (Jiao et al., 1998). Fig.2 illustrates, using the Unified Modelling Language (UML), a product manufacturing model intended to welcome the processes family description based on manufacturing modules called manufacturing-services (M-Ser) and the customer specification through the setting of design parameters.

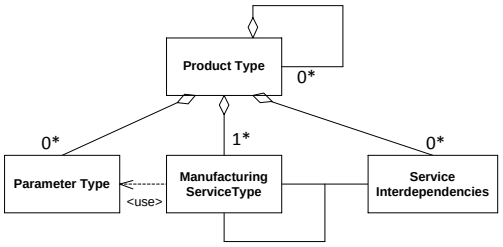

Fig. 2. UML Product ManufacturingModel

The manufacturing model of a specific product family type, i.e. process family type, presented above, represents all the necessary information for the manufacture of a product in a given production platform. This model is a direct mapping of the information defined by the product model of the ISA SP-95 norm and was divided in three elements.

- *Product Parameter*: represents a variable that will be derived from the customization process. This parameter corresponds to a process variable derived from the customization choices made in the physical domain. According to its cardinality a product might have no or several parameters to be defined according to the level of customization attributed. These introduce the scalable character into the process domain.
- *Manufacturing-Service (M-Ser):* Represents a manufacturing task module resulting from the mapping of a product structural feature into the process domain. They correspond to descriptions of manufacturing capabilities with no regard to the methods for their implementation.
- *Service Interdependencies/Service Recipe:* Information explicating the relation and interdependencies among the different M-Sers that comprises a process family (i.e. a product family). It defines the precedence rules between the M-Ser Modules for the orchestration of production workflows. Its cardinality includes zero considering the possibility of an uncoupled production process represented by a single M-Ser, hence no need for precedence.

There exist three important relations in the model: the dependency (or use case) relation between parameter and the M-Ser, the auto-aggregation relation of the product family and the relation between the service interdependencies class and the M-Ser class. The dependency relation indicates that each of the parameters is linked to a M-Ser as to complete its specification for execution. The auto-aggregation relation suggests the possibility of the composition of a product by various sub-products that can have themselves a product model specification declared in the system. Finally there is the relation between the interdependencies class and the M-Ser class which indicates that the interdependencies class defines the relations between the different M-Sers with a certain type of formalism/methodology. Moreover, the parameter type class, besides having an identification, also contains a range of permissible values or choices for its instance. Up to now, the M-Ser Interdependencies are considered to be proprietary of the process family to which they belong and defined by the process family designer, and not as properties of the M-Sers themselves.

In short, product differentiation is achieved by both the specification of product parameters and the configuration of the different M-Ser modules by their addition, subtraction and/or substitution. It is the instantiation of each of the three elements that completely determines the production information required for the realization of a product variant. Such specification is independent of the production platform as the M-Sers are mere descriptions of the manufacturing tasks with no regard of the resources or methods implementing them. This quality makes the manufacturing product specification compatible with all types of resource models as long as they can provide the required manufacturing services. In this way, this manufacturing model contains the process family description through the collection of M-Sers and their interdependencies as well as the customer specification through the collection of parameters and the M-Sers modules selected (in case of modular choices).

3. MANUFACTURING SERVICES

3.1 Concept of services

As presented on section 2, the proposed product specification is based on the structural repeatability and reusability in product families. As it was mentioned, process families map the products' structural components into manufacturing process modules shared between products. In this way a bank of standardized manufacturing process modules can be created for their further reutilization given the case of an existing commonality with other process families in the same way as for structural modules in product families.

In software development, SoA (Norihisa,2006) standing for Service oriented Architecture, is a decentralized architecture that decomposes computational process into sub-processes for later distribute these among the different treatment resources available tacking advantage of their capabilities as well as of the inherent parallelism of the process. The functions treating these sub-processes are called services. According to the definition found in (Grönoos, 2001), a service is a single activity or a series of activities of a more or less intangible nature that normally takes place in the interactions between costumer and service provider, provided as a solution to achieve the customer's desired end results. SoA, within its principles, also considers the possible orchestration of different service sequences. This is of great use in the context of PDSs as they normally comprehend multiple process alternatives i.e. different workflow orchestrations. In HMS this means, more than one Resource Holon (RH) can execute one same manufacturing task and more than one workflow can produce the same product. Moreover, RHs providing the same services not necessarily use the same technology i.e. different internal models. RHs are the virtual representations of one (group of) machine(s) for which manufacturing functions have been pre-programmed according to their internal model and technologies.

By adopting the concept of services and SoA's principles, the manufacturing modules can be represented in the form of services, having a proper identification and description, which thanks to their modularity can be orchestrated in different ways thus giving a higher level of flexibility to the system. In this way, the resources' capability is delimited by the catalogue of manufacturing services (M-Ser) that they offer to the system, based on the tasks' nature itself, rather than on the identification of a specific function proprietary to the resource that has to be known a priori during process design. Thus, integrating services facilitates the integration of new resources and a process design independent from the production platform knowledge.

3.2 Manufacturing services modelling

Translating the concept of services to the manufacturing context gives rise to a specific type of service: the Manufacturing Service (M-Ser) which in turn needs of a specific model. Fig. 3 shows the model of a M-Ser conceived to welcome product customization. Like it is illustrated, a M-Ser is composed of two main elements (classes):

- *Operation:* represents the activity related to the M-Ser. From the consumer perspective: descriptions of the transformations made on the product. From the provider's perspective: the function with the algorithms that execute the M-Ser. Such algorithms are dependent of the resource's technology and are proprietary to the RH providing it. Therefore, the operation type is unique to the service customers but there can exist different instances that are internal to the provider RH and that are not visible to the rest of the system. Each M-Ser has one single operation class as it's cardinality shows, i.e. there is no need to declare more than one function for the execution of a M-Ser. Examples of operation types are: {perforate, paint, weld, etc.} which need of some parameters specification.
- *Parameter*: It can come in two forms: variables or materials. In the former case, it represents a variable with a range of permissible values corresponding to a design parameter from the physical domain during customization e.g. Element X position in x coordinate = {0 - 10} cm or {0, 2, 4, 6, 8, 10} cm. In the latter case, they indicate the category of the component to be added to the main product by the operation. The selection of the material or sub product is done inside a range of component variants of the same category, e.g. *Category*: Hard Disk, *Range*: {200Gb, 300Gb, 400Gb,1Tb}. Its cardinality indicates that a M-Ser can comprise zero or more parameters depending of its flexibility level to reproduce different results out of the same operation function.

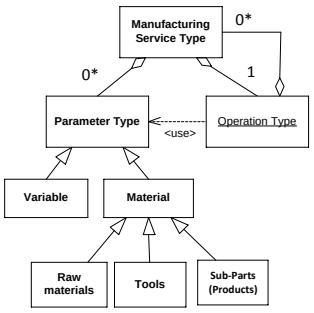

Fig. 3. UML Manufacturing Service Type Model

In the M-Ser model there exist two important relations: the dependency relation between parameter and operation and the mutual aggregation relation between the operation and the M-Ser class. The dependency relation (from the provider's perspective) indicates the algorithms' necessity of the M-Ser's parameter specification for their possible execution. On the other hand, there is the mutual aggregation relation. (Grönoos, 2001) definition implies that a service can be itself composed of other more granular services. Such services are referred as Compound Manufacturing Services (CM-Ser). Such relation can be seen as indirect auto-aggregation done thought the operation class. The question still remaining is: why not better make a direct auto-aggregation of the MSer class with itself? The reason is that the content of the operation class, from the provider's perspective, depends on the provider's internal model and not on the service definition itself. Hence, one same M-Ser provided by different RHs can be considered either as a single service or as a compound service depending on the RH providing it. It might be the case that the RH could implement an internal holonic platform to provide a M-Ser by the orchestration of more granular services instead of a single hard programmed function. It is precisely the decoupling of parameters from operation that allows bringing product customization to the manufacturing domain. Therefore, as product customization is based on the reutilization of structural features, the M-Sers

used to produce such features can be adapted to the different products variants and/or families through service parameterization. The group of Manufacturing Services that can be derived from this model can be seen as a family sharing a same operation description but differentiated by the values in their parameterization. In this way, the instance of a manufacturing service type will be issued from the instantiation of the Parameters Type class which is in turn determined by product customization.

4. MODELING PROCESS FAMILIES WITH PETRI-NETS

In the context of a PDS, the methods and techniques for representing product manufacturing specification is one of the key design issues as the control of the system is based on products. Such method of representation should satisfy the following needs identified from the product manufacturing model:

- Contain all the information of the product manufacturing-model i.e. parameters, M-Sers and service interdependencies.
- Capable of expressing all the possible service choreographies (workflows) that can realize the specified product.
- Must facilitate the online edition of the product specification.
- Must be simple to understand and to program.
- Its memory requirement must be minimal for possible embedded applications as in the implementation of intelligent products level 2 (Wong et al 2002).

Petri Nets (Murata, 1989), a well-known modelling formalism in the academic and industrial domain, turns out to be a very good candidate for this purpose. This is mainly due to its characteristic ability to capture the synchronous and asynchronous aspects of a process in a simple manner.. Thanks to this, and to the evolution mechanisms of the formalism, petri nets have the advantageous capacity of representing a great number of sequence combinations with a single net. This is of great importance as the main goal is to design a product manufacturing informational model that allows the exploitation of the flexibility to the PDS control strategies. This flexibility should translate into the exploration of all the possible production workflows realizing a specified product. Traditionally, process design involves the implementation of static models which specify a single predefined production workflow. The use of Petri Nets has already been recognized by (Mendez et al 2010) by enriching the process model by considering the existence of different alternative services for a given production state with the objective of involving decision engines in the system. The modelling strategy presented in this paper has the intention enrich the process model, still using Petri Nets in order to increase the decision area by putting in question the ejection order of services. By fusing together the Petri-Net formalism, product family design and the concept of manufacturing services a net with the following characteristics is obtained:

- M-Sers are represented by transitions each with its identifier. Being $S:T \rightarrow \{s1,s2,..,sn\}$ the finite set of M-Sers associated to the corresponding transitions.
- Services are expressed in the form *M-Ser_id(parameters)*. Parameters are attributes of the M-Sers.
- The production states of a product are indicated by the Nets' marking. Such marking implicitly indicates the services that have been executed at a given point.
- The service interdependencies are defined by the set of arcs connecting places and transitions and the evolution rules of the Petri-Net formalism. This information is inherently contained in the Net's structure.

To better understand the Petri-Net approach, Fig. 6 shows an illustrative example of a theoretical process family using Legos. It consists of a Lego base that will be used to represent the transporter of the product and a set of blocks each one standing for an instance of a certain M-Ser type. The idea is to use the structural dependencies in the Lego structure to illustrate a process family structure and how this can be represented with a Petri-net. The Lego process family, Fig.4, comprises two family members: variant 1 and variant 2. Table 1 contains the Lego block representation of the different M-Ser types.

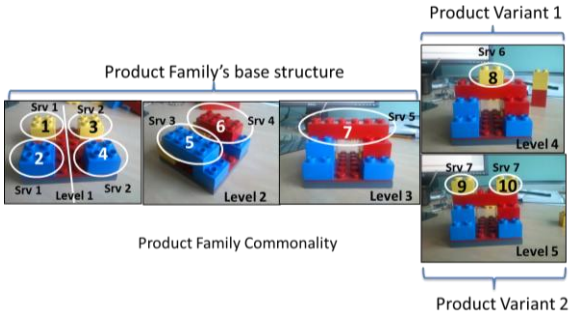

Fig. 4. Example: Process configuration illustration

M-Ser Type	Lego id
Type 1 (Sr 1)	1&2
Type 2 (Sr 2)	3&4
Type 3 (Sr 3)	5
Type 4 (Sr 4)	6
Type 5 (Sr 5)	7
Type 6 (Sr 6)	8
Type 7 (Sr 7)	9&10

Table 1. Example: Equivalence Table

Out of the Lego configuration it can be seen a series of interdependencies between the M-Sers. These interdependencies are illustrated in Table 2. Serving from the interdependencies table, the following Petri-Net structure can be derived (Fig. 5). As mentioned before, the production state of the product in question is given by the net's marking which enables the triggering of certain transitions. Thanks to this, at a certain production state, it can be known the allowed M-Sers to execute next, that will respect the M-Ser interdependencies. This allows the exploration of the alternatives given by the independence between certain

services. For example, services corresponding to Legos 1 through 4 can be executed at any time in the production state as long as they haven't been made thanks to their lack of dependencies. On the other hand, Sr3 is dependent of the previous execution of both Sr1. In short, from a single Petri-Net, a state-automaton can be generated with all the possible workflows, lower memory consumption and more straight forward programming.

M-Ser Type	Uphill Dependancies	Downhill Dependencies
Service-type 1(Sr 1,1)	-	Sr3
Service-type 1(Sr 1,2)	-	Sr3
Service-type 2(Sr 2,1)	-	Sr4
Service-type 2(Sr 2,2)	-	Sr4
Service-type 3(Sr 3)	(Sr1,1 & Sr1,2)	Sr5
Service-type 4(Sr 4)	(Sr2,1 & Sr2,2)	Sr5
Service-type 5(Sr 5)	(Sr3 & Sr4)	(Sr6) | (Sr7,1 & Sr7,2)
Service-type 6(Sr 6)	Sr5	-
Service-type 7(Sr 7,1)	Sr5	-
Service-type 7(Sr 7,2)	Sr5	-

Table 2. Example: Manufacturing-Service interdependencies

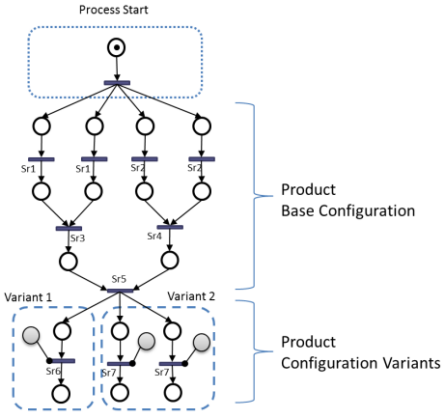

Fig. 5. Example: Petri-Net representing the Product Manufacturing Model

5. CONCLUSION

This paper introduces an innovative way of describing products intended to be used in Service-oriented HMS. This new representation is useful for systems where multiple variants of the same recipes are required. It is based on Petri-Nets, enabling a compact and exhaustive representation of all the possible variants of the same recipe. The application of these concepts is made on a simple test bed made of assemblies of Lego blocks. This study case has the particularity to clearly illustrate the possible variants in manufacturing, where some operations can be performed before others without impact on the quality of the final product. Future works deal with the implementation of these nets in a Service-oriented HMS. Indeed, it is necessary to introduce new negotiation protocols between holons to be able to take into account the flexibility added by the variants in the recipes.

REFERENCES

Babiceanu, R.F. and Chen, F.F. (2006). Development and applications of holonic manufacturing systems: a survey, *Journal of Intelligent Manufacturing*, 17(1), 111-131

Blanc, P., Demongodin, I. and Castagna, P. (2008). A holonic approach for manufacturing execution system design: An industrial application, *Engineering Applications of Artificial Intelligence*, 21(3), 315-330

Child, P., R. Diederichs, F.H. Sanders and S. Wisniowski (1991). Sloan Management Review. SMR forum: The management of complexity. 33, 73-80.

Davis, S., (1987). *Future Perfect*, Addison Wesley, Reading, MA.

Fisher, M., Ramdas, K., & Ulrich, K. (1999). Component sharing in the management of product variety: A study of automotive braking systems. Management Science, 45(3), 297–315.

Grönroos, C. Service Management and Marketing. A customer relationship management approach. 2nd edition, Chichester: Wiley, 2001.

Jiao, J., Tseng, M., Duffy, V., Lin, F. (1998). Product Family Modeling for Mass Customization. Computers and Industrial Engineering, Vol. 35 (3-4), 495-498.

Jiao, J., Simpson, T.W. (2007). Product family design and platform-based product development: a state-of-the-art review. Journal of intelligent Manufacturing, Vol. 18, 5-29.

Martinez, M.T., Favrel, J., & Fhodous, P. (2000). Product family manufacturing plan generation and classification. Concurrent Engineering: Research and Applications, 8(1), 12–22.

Mendes, J. M., Leitão, P., Restivo F., Colombo, A. W., 2010. Process Optimization of Service-Oriented Automation Devices Based on Petri Nets, 978-1-4244-7300-7 IEEE, pp. 274-279

Meyer, M., & Lehnerd, A. P. (1997a). The power of product platform– building value and cost leadship. New York: Free Press.

Meyer, M.H., P. Tertzakian, J.M. Utterback. (1997b). Metrics for managing research and development in the context of product family. Management Science, 43, 88-111.

Morel G., and Grabot, B. (2003) Editorial of special issue, *Engineering Applications of Artificial Intelligence*, 16(4)

Murata, T., 1989. Petri nets: Properties, Analysis and Applications, IEEE, vol. 77, pp. 541-580.

Sallez, Y., Berger, T. and Trentesaux, D. (2009) A stigmergic approach for dynamic routing of active products in FMS, *Computers in Industry*, 60(3)

Schierholt, K. (2001). Process configuration: Combining the principles of product configuration and process planning. AIEDAM, 15(5), 411–424.

Simpson, T.W., Maier, J.R.A., and Mistree, F. (2001a). Product platform design: Method and application. Research in Engineering Design, 13(1), 2–22.

Suh, N. P. (2001). Axiomatic design: Advances and applications. New York: Oxford University Press.

Tseng, M. M., & Jiao, J. (1996). Design for mass customization. CIRP Annals, 45(1), 153–156.

Valckenaers, P. (1998). Special issue on intelligent manufacturing systems, *Computers in Industry*, 37(3)

Wong, C. Y., McFarlane D., Zaharudin A. A., and Agarwal V.,2002. «The Intelligent Product Driven Supply Chain », IEEE Systems Man and Cybernetics, Hammammet, Tunisie.