

NESTED COLORED TIMED PETRI NETS FOR PRODUCTION CONFIGURATION OF PRODUCT FAMILIES

Linda Zhang, Brian Rodrigues

▶ To cite this version:

Linda Zhang, Brian Rodrigues. NESTED COLORED TIMED PETRI NETS FOR PRODUCTION CONFIGURATION OF PRODUCT FAMILIES. International Journal of Production Research, 2009, 48 (06), pp.1805-1833. 10.1080/00207540802585329. hal-00548945

HAL Id: hal-00548945

https://hal.science/hal-00548945

Submitted on 21 Dec 2010

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.

International Journal of Production Research



NESTED COLORED TIMED PETRI NETS FOR PRODUCTION CONFIGURATION OF PRODUCT FAMILIES

Journal:	International Journal of Production Research
Manuscript ID:	TPRS-2008-IJPR-0502.R1
Manuscript Type:	Original Manuscript
Date Submitted by the Author:	15-Oct-2008
Complete List of Authors:	Zhang, Linda; University of Groningen, Operations Rodrigues, Brian; Singapore Management University, Lee Kong Chian School of Management
Keywords:	PRODUCTION MODELLING, PETRI NETS
Keywords (user):	PRODUCTION MODELLING, PETRI NETS



NESTED COLORED TIMED PETRI NETS FOR PRODUCTION CONFIGURATION OF PRODUCT FAMILIES

Lianfeng (Linda) Zhang*†, Brian Rodrigues‡

†Department of Operations, University of Groningen, Landleven 5, 9747 AD Groningen, The Netherlands ‡Lee Kong Chian School of Business, Singapore Management University Stamford Road 50, 178899, Singapore

Abstract: Production configuration is as an effective technique to deal with product variety while maintaining production stability and efficiency. It involves a diverse set of process elements (e.g., machines, operations), a high variety of component parts and assemblies and many constraints arising from product and process variety. Production configuration entails the selection and subsequent arrangement of process elements into complete production processes and the final evaluation of configured multiple alternatives. To better understand production configuration and its implementation, we study the underlying logic for configuring production processes using a dynamic modeling and visualization approach. This is accomplished by developing a new formalism of nested colored timed Petri nets (PNs). In view of the inherent modeling difficulties, in the formalism three types of nets - process nets, assembly nets and manufacturing nets - together with a nested net system are defined. Using an industrial example of vibration motors, we show how the proposed formalism can be applied to specify production processes at different levels of abstraction to achieve production configuration.

Keywords: Process platform, production configuration, Petri nets, modeling.

^{*} Corresponding author. Email: L.Zhang@rug.nl

1. Introduction

The importance of planning production processes has been well recognized (Huang et al., 2004). As a vital link between design and production, planning determines how well a product can be fulfilled in terms of cost, lead time and quality (Martinez et al., 2000). Decision making in planning production processes is complex since it involves multiple, often conflicting, production performance metrics, a variety of operations types, operations precedence and alternative manufacturing resources of same kinds (Chan et al., 2001). Production processes of products consist of both manufacturing processes of component parts and assembly processes of component assemblies. Thus, production process planning provides inputs (e.g., parts, assemblies, manufacturing resources) to downstream computer-aided process planning and assembly planning for specifying detailed process parameters such as cutting speed, feed rate and depth of cut (Turner et al., 1992). It also provides inputs for scheduling needs (Martinez et al., 2000).

Manufacturers at large have adopted the strategy of developing product families in attempting to stay competitive through satisfying diverse individual customer requirements while maintaining low costs (Sanderson and Uzumeri, 1996). Successful product family development relies on achieving efficiencies in both designing and producing product families. Some researchers (e.g., Fixson (2005), Huang et al. (2005). Lu and Botha (2006), to name but a few) have pointed out that planning of production processes for product families can help achieve production stability, and eventually mass production efficiency. Production processes planning for product families is more challenging and complex when compared with that for individual products (Martinez et al., 2000). The reasons are threefold. First, the optimal production processes of individual products may not contribute to that of the product family as a whole due to the limited number of manufacturing resources (Zhang, 2007). Second, the solution space of production processes of product families is combinatorial in nature due to the large number of individual products, process types, operations types, and process elements (Martinez et al., 2000). Last, production processes planning for product families involves specific technical and managerial challenges, which are inadequately addressed in the literature (Lu and Botha, 2006).

Among the authors reporting production process planning for product families, some approach this topic from a strategic level and, their solutions provide companies with managerial guidelines for deciding on production/manufacturing structures and investment choices (Blackhurst et al., 2005; Garg, 1999; Huang et al., 2005; Petersen et al., 2005; Raman and Chhajed, 1995). Others address planning manufacturing processes for part families (Schierholt, 2001; Williams et al., 2007) and planning assembly sequences for assembly families (De Lit et al., 2003; Gupta and Krishnan, 1998; Martinez et al., 2000). This suggests that few researchers deal with planning production processes for product families that involves both parts and assemblies at an operational level. In response to the lack of research, configuring production processes from a process platform in relation to a product family (i.e., production configuration) has been proposed as an effective means of achieving product family production stability and efficiency (Jiao et al., 2007; Zhang, 2007).

A process platform organizes all data pertaining to a product family and the corresponding process family as an integrated common process structure. Thus, production configuration provides a generic umbrella to capture and utilize commonality, within which each new product fulfillment is instantiated and extended, thereby anchoring planning to the integrated common process structure. Within a process platform, elements such as processes, operations, machines, tools, fixtures and setups are selected and arranged to form optimal production processes for individual products by considering the cohort of the product family. Production configuration involves product and process variety exhibited by large volumes of different types of product and process family data, and many constraints, which, for example, describe the connections between different types of data. In addition, configuration needs to deal with the many process variations resulting from design changes among members in the product family. Lastly, in accordance with the many assemblies and parts at different levels of product hierarchies, production configuration concerns granularity issues (i.e., configuring processes for product items at different levels of abstraction). To help manufacturers better understand production configuration, this study addresses the underlying reasoning by modeling the process of configuring production processes for a product family from the corresponding process platform.

Due to their executability, graphical representation and mathematical support, Petri nets

(PNs) have been well recognized as a powerful modeling, simulation and evaluation tool for complex flows and processes (Peterson, 1981). As a graphical tool, PNs serve as a visual modeling technique and as a communication aid for describing models. As a mathematical tool, PNs can be exploited to perform qualitative and quantitative analysis of systems being modeled. A number of extensions have been made to classic PNs in order to satisfy different modeling requirements. Among all, colored PNs (CPNs; Jensen, 1995), timed PNs (TPNs; Ramachandani, 1974) and nested PNs (NPNs; Lomazova, 2000; Lopez-Mellado and Almeyda-Canepa, 2003) are of particular interest in this study. CPNs are able to provide a concise, flexible and manageable representation of large systems by attaching a variety of colors to tokens. By including timing, TPNs can capture systems physical behaviors through assuming specific durations for various activities. By considering PNs as tokens, NPNs model large and complex system through refinement and abstraction of PNs.

Towards this end, in this paper, we apply PN techniques, specifically CPNs, TPNs and NPNs, to model production configuration. Accordingly, a new formalism of NCTPNs (nested colored timed PNs) is developed by integrating the basic principles of CPNs, TPNs and NPNs. In accordance with the different types of processes in high variety production and the inherent similarities among them, in the proposed formalism several nets are defined based on a basic PN structure. Product and process variety and configuration constraints are handled by attaching various data regarding product items and process elements to tokens, resulting in colored tokens. A reconfiguration mechanism, consisting of inhibitor arcs, reconfigurable transitions and conceptual machine places, is incorporated into the formalism to cope with process variations. Timing is introduced to facilitate the selection of proper machines, operations and processes. Moreover, a multilevel net system is further developed based on net nesting to tackle explicitly granularity issue in production configuration.

The rest of the paper is structured as follows: Section 2 provides a review of work relevant to PNs modeling applications. The definitions in the proposed formalism are presented in Section 3, following which the multilevel system of NCTPNs is introduced in Section 4. Also presented is the evolution of the net system through defining several rules. In the industrial example in Section 5, the formalism is applied to model production configuration of vibration motors for mobile phones. Also discussed are system analysis and

application results. Further discussions along with some managerial implications are provided in Section 6.

2. RELATED WORK

To capture the hierarchical nature of assembly planning, Thomas et al. (1996) propose a hierarchical PN framework, where each control plan corresponding to an assembly robot operation is viewed as a lower level representation of assembly tasks at the higher level. Adamous et al. (1998) introduce a hierarchical model of flexible assembly system control using object PNs, which is built on hierarchical decomposition of the system through coordination among lower level decomposed system elements. Lomazova (2000) discusses two level NPNs for modeling multi-agent distributed system and the extension to multi-level NPNs. She proves that NPNs are able to maintain the properties of classic PNs. To accommodate qualitative modeling of multiple mobile robot systems, Lopez-Mellado and Almeyda-Canepa (2003) develop a three level scheme of PNs, in which tokens themselves can be defined as PNs as well. In software system modeling, Hiraishi (2000) presents a PN-based model, which adopts the classic PNs to analyze and design multi agent systems. He represents agents using tokens, which themselves are PNs.

Li and Lara-Rosano (1999) develop a hybrid object-oriented colored PNs (OOCPNs) by enhancing OOCPNs with time delay and firing speed. The hybrid OOCPNs are used to model formally batch-based electronic component manufacturing systems. Recognizing the importance of solving deadlocks in flexible manufacturing systems, Abdallaht et al. (2002) put forward a deadlock free scheduling algorithm using TPNs in order to minimize mean flow time. Reddy et al. (1993) present an algorithm for qualitative and quantitative analysis of TPN models in manufacturing systems with focus on real time control and performance evaluation. Liu et al. (2002) propose a workflow modeling language-based CPNs called WFCP-nets (workflow based on CPNs) and apply it to the product development workflow. Chin et al. (2006) put forward methodologies based on integrated definition language and CPNs for modeling and simulating complicated manufacturing processes. Researchers, e.g., Dotoli and Fanti (2004), Nandula and Dutta (2000), Jiang et al. (2000), Ravi Raju and Krishnaiah Chetty (1993), to name but a few, also apply CPNs and TPNs to manufacturing

systems modeling, analysis and control. An observation on available PN models is that most researchers have adopted the basic ideas of different extensions of classic PNs and further extended them to accommodate the different modeling requirements of their own problem domains. Similarly, in this work, we integrate the advantages of NPNs, TPNs and CPNs to accommodate the difficulties in modeling production configuration.

3. DEFINITIONS OF NESTED COLORED TIMED PETRI NETS

Production configuration entails a process of configuring a complete production process for a product along with the product hierarchy (i.e., at different levels of abstraction). Being located at the highest level, the more abstract production process involving the product's immediate child items is configured first. Subsequently, the assembly processes and/or manufacturing processes for producing these immediate child items are configured at the second level, and so on. The manufacturing processes for parts at the lowest level of each branch of the hierarchy are last to be configured.

In each such process, be it a manufacturing process for a part, an assembly process for an assembly or the abstract production process for the final product, a number of alternative machines (including the corresponding tools, fixtures, etc) are able to complete same tasks. While these machines work on same input items for same tasks, they execute different operations and, in most cases, incur different cycle times. In practice, it is a common solution that buffers are adopted to keep raw materials, parts, assemblies and final products. Furthermore, a number of different types of WIPs (work in processes) are involved in a process. Unlike a part or assembly that can be located in the BOM (bill of materials) of a product, a WIP is an intermediate pseudo item formed by sibling parts and/or assemblies and is not specified in the BOM.

Bearing in mind the similarities embedded in processes (e.g., adoption of buffers and alternative machines, involvement of WIPs), we define a basic PN structure first. Accordingly, the basic PN structure reflects the generalized common process elements, which are assumed by different types of nets defined in the proposed formalism.

3.1 Basic Net Structure

Definition 1: A basic PN structure is a directed bipartite graph G = (P, T, h, A), where $P = P^B \cup P^I \cup P^R \cup P^{CR}$ is a finite set of places with four disjoint subsets: P^B representing buffers (be it for raw materials, parts, assemblies, or final products), P^I items

ready to be processed (be it a raw material, part, assembly, or WIP), P^R machines, and P^{CR} the conceptual machines in accordance with alternative machines that can complete same tasks by performing different operations;

 $T = T^L \cup T^R \cup T^T$, $P \cap T = \phi$ is a finite set of transitions with three disjoint subsets: T^L denoting a set of logical transitions, T^R a set of reconfigurable transitions, and T^T a set of timed transitions;

 $h \subseteq P^{CR} \times T^R$ is a finite set of inhibitor arcs that connect a conceptual machine place to a reconfigurable transition and assumes two values: 1 and 0. When h(p,t)=1, $\forall p \in P^{CR}$, $t \in T^R$, there is a token in the conceptual machine place; the associated reconfigurable transition is disabled and cannot fire. When h(p,t)=0, no token resides in the conceptual machine place; the associated reconfigurable transition can fire if it is enabled; and

 $A \subseteq (P \times T \cup T \times P) \cap \overline{h}$ is a finite set of arcs excluding inhibitor arcs that connect places/transitions to transitions/places.

Attempting to capture and model multiple alternative machines in relation to same tasks, P^{CR} is defined in the formalism to represent the corresponding conceptual machines in addition to P^R , which is defined to denote specific machines. The common practice suggests that only one machine works on same items at one time. Thus, in conjunction with P^{CR} , h and T^R are introduced to model this situation, where multiple machines can perform same tasks and only one is used eventually. The firing of T^R leads to the reconfiguration of proper machines. In this way, P^{CR} , T^R and h can address process variations in system models without rebuilding new ones when machines are added and/or removed.

 T^L is defined to capture the logic of system running. Their firing indicates the satisfaction of preconditions of operations, among which the typical one is the presence of material items and machines to be used. T^T is defined to represent operations, which take certain time durations to complete. Accordingly, the firing of timed transitions incurs time delays. Both logical and reconfigurable transitions are untimed. Their firing is instantaneous and takes 0 time delay.

Figure 1 shows a basic PN structure and the corresponding graphical formalism. Based on the basic PN structure, three types of PNs, namely manufacturing nets (*MNets*), assembly nets (*ANets*) and a process net (*PNet*) are defined to address the granularity issue in production configuration. These nets are defined as a type with a marking, with each type

following the basic PN structure and including additional information.

3.2 Manufacturing Net (*MNet*)

Definition 2: A manufacturing net is defined as a tuple $MNet = (ToM, \mu)$, where MNet is a manufacturing net representing the processes of manufacturing a part family; $ToM = (G, \Sigma^M, \alpha, \beta, E, \tau)$ is a manufacturing type with

- G is the basic PN structure;
- Σ^{M} is a finite set of color sets or types, each of which includes a number of individual color instances;
- $\alpha: P \mapsto M_{\Sigma^M}$ is a color assignment function that maps a place, p, to a set of colors, $\alpha(p)$ $(M_{\Sigma^M}$ is the family of all multisets over Σ^M);
- $\beta: T \mapsto (\Sigma^M) \times (\Sigma^M \cup \{\varepsilon\})$ is a color assignment function that maps a transition, t, to a set of color pairs, $\beta(t)$ such that each pair declares that a t must be synchronized externally with transitions of the net where this manufacturing net resides, except if the pair is (Σ^M, ε) with ε representing nil, indicating no synchronization is required;
- $E: A \times \Sigma^M \mapsto \bigvee M_{\Sigma^M} \cup \bigvee (\bigwedge M_{\Sigma^M} \to M_{\Sigma^M} @ + \tau)$ is an arc expression function that defines the timed and untimed arc expressions for arcs with respect to transition colors; \bigvee / \bigwedge denote *Exclusive OR (XOR)/AND* relationships; and \longrightarrow represents an "if—then" relationship;
- $\tau \in \Re^+ \cup 0$ is a set of non-negative real numbers representing time delays;

 $\mu: P \mapsto M_{\alpha(p)}, \forall p \in P$ is a marking function specifying the distribution of colored tokens in all places of an $MNet(M_{\alpha(p)})$ is the family of all multisets over $\alpha(p)$.

In PN models, tokens residing in places are used to represent objects. To reflect diverse product and process variety while building a concise and representative model, colors, i.e., specific data values, are attached to tokens. Each colored token is uniquely defined by a color, and vice versa. Since there is a one-to-one correspondence between colors and colored tokens, hereafter, colors and colored tokens are used interchangeably. While the places representing machines, conceptual machines, WIPs and part buffers are common to an *MNet*, *ANet* and *PNet* (the latter two are defined below), the places denoting raw material buffers are unique to an *MNet*. The set of colored tokens representing raw materials in a raw material

buffer are defined according to the corresponding parts to be produced.

Machines have two statuses - busy and idle. While tokens in idle machine places are defined to represent machines, tokens in busy machine places are defined according to items to be produced. When there is a token residing in the conceptual machine place, the conceptual machine is instantiated to the specific machine represented by the token. A part buffer place contains the set of colored tokens representing the corresponding part family. Following production practice, we define colored tokens of logical and reconfigurable transitions to be the same as those of output places and, similarly, colored tokens of timed transitions as those of input places.

Tielemans (1995) points out that cycle times are a useful performance measure of production. Thus, in this study we adopt cycle times to accommodate selection of proper processes from configured alternatives. Accordingly, time delays, τ , are defined in the formalism to represent operations cycle times. Time delays can be obtained from a process platform of a process family in relation to a product family (Zhang, 2007). See (Bowden, 2000) for a comprehensive review of time representations in PN models.

To cope with the difficulties in modeling diverse cycle times associated with multiple machines and same tasks, an arc expression function, E, is introduced in the modeling formalism. It defines both timed arc expressions and untimed arc expressions. For every arc, E relates transition colors with multisets of place colors establishing pre- and post-conditions for transition firing. A timed arc expression is a set of antecedent-consequent statements with XOR relationships. Each antecedent contains a set of colored tokens with AND relationships. The occurrence of each such colored token may not be 1, and by default, the occurrence of 1 is omitted. The consequent includes a colored token to be generated in the busy machine place together with a time delay representing the corresponding operation cycle time. Untimed arc expressions are defined to specify (1) the input tokens for firing transitions; and (2) output tokens after firing transitions.

Figure 2 shows an example of an MNet. In this net, the set of places is $\{p_1, \cdots, p_7\}$; the set of transitions is $\{t_1, \cdots, t_4\}$; the set of inhibitor arcs is $\{(p_7, t_3), (p_7, t_4)\}$; the set of arcs is $\{(p_1, t_1), (p_2, t_1), \cdots, (t_7, p_1)\}$; and the set of color sets is $\{\{a_1, a_2\}, \{b_1, b_2\}, \{A_1, A_2, A_3\}, \{m_1, m_2\}\}$. Table 1 shows the places, represented system elements and assigned colored tokens.

The color pairs assigned to transitions and the timed/untimed arc expressions with respect to transition colors are shown in the figure (note: for a better understanding, the arc expression functions in accordance with individual color pairs of transitions are also given). The three different time delays in the timed arc expression of (t_1, p_3) : 10, 8 and 12 units of time are the cycle times incurred by m_1 processing a_1 and b_1 , m_2 processing a_1 and b_2 , and m_1 processing a_2 and b_1 , respectively. A_1 , A_2 and A_3 denote the three parts to be produced. The color pairs of t_2 , $\{(A_1, A_1), (A_2, A_2), (A_3, A_3)\}$, indicate that when t_2 is enabled with respect to (A_1, A_1) or (A_2, A_2) or (A_3, A_3) , it must fire simultaneously with transition(s) containing same color pairs of the higher level nets, in which this MNet is contained. The color pairs of other transitions indicate that their firing is autonomous and transition synchronization is not required.

The marking of the four places containing tokens is $\mu(p_1) = (a_1)$, $\mu(p_2) = (b_2)$, $\mu(p_6) = (m_2)$ and $\mu(p_7) = (m_1)$ and the marking of the *MNet* is $(a_1, b_2, 0, 0, 0, m_2, m_1)$.

3.3 Assembly Net (ANet)

Definition 3: An assembly net is defined as a tuple $ANet = (ToA, \mu)$, where

ANet is an assembly net representing the processes of producing a family of assembly variants;

 $ToA = (G, SoTma, \Sigma^A, \alpha, \beta, E, \tau)$ is an assembly type with

- G is the basic PN structure;
- SoTma is a set of manufacturing types and assembly types;
- Σ^A is a finite set of color sets or token types;
- $\alpha: P \mapsto SoTma \cup M_{\Sigma^A}$ is an assignment function that maps a place, p, to manufacturing or assembly types or a set of colors, $\alpha(p)$;
- $\beta: T \mapsto (\Sigma^A \cup \{\varepsilon\}) \times (\Sigma^A) \times (\Sigma^A \cup \{\varepsilon\})$ is an assignment function that maps a transition, t, to a set of color triples, $\beta(t)$, such that each triple declares that the transition, t, must be synchronized 1) internally with transitions of the nested net; and 2) externally with transitions of the host net where this assembly net is contained, except if the triple is $(\varepsilon, \Sigma^A, \varepsilon)$ with ε representing nil;

- $E: A \times \Sigma^A \mapsto \bigvee M_{\Sigma^A} \cup \bigvee (\bigwedge M_{\Sigma^A} \to M_{\Sigma^A} @+ \tau)$ is an arc expression function that defines the timed and untimed arc expressions for arcs with respect to transition colors; and
- $\tau \in \Re^+ \cup 0$ is a set of non-negative real numbers representing time delays;

 $\mu: P \mapsto M_{\alpha(p)}, \forall p \in P$ is a marking function specifying the distribution of colored tokens, manufacturing and assembly nets in all places of an ANet.

An *ANet* is defined to represent the processes of producing a family of assemblies. Unlike raw material buffer places in an *MNet*, buffer places in an *ANet* are defined for the immediate child items of assemblies. Tokens in such part/subassembly buffers are MNets / ANets of child items. The assignment function, α , specifies the allocation of such MNets / ANets to buffer places and colored tokens to other places.

Figure 3 shows an example of an ANet. Also shown is the MNet (in Figure 1) nested in place p_9 of the ANet. In this net, the set of places is $\{p_8, \dots, p_{14}\}$; the transition set is $\{t_5, \dots, t_8\}$; the arc set is $\{(p_8, t_5), (p_9, t_6), \dots, (t_8, p_{14})\}$; and the set of color sets is $\{(A_1, A_2, A_3), \{B_1, B_2\}, \{X_p, X_2, X_3\}, \{m_3\}\}$. A_1, A_2 and A_3 represent variants of part family A; B_1 and B_2 represent variants of part family B; and X_p, X_2 and X_3 represent variants of assembly family X. Table 2 lists the set of places, represented system elements and tokens and nested nets.

Both the color triples assigned to transitions and the timed/untimed arc expressions with respect to transition colors are shown in the figure. For example, the color triple assigned to t_6 of the $ANet:\{(A_1,A_1,\varepsilon),(A_2,A_2,\varepsilon),(A_3,A_3,\varepsilon)\}$ indicates that if enabled, t_6 must fire simultaneously with the nested nets at a lower level, whilst t_8 needs to be synchronized externally with transitions of the host nets at the higher level. The marking of two places of the ANet is $\mu(p_9)=(MNet)$ and $\mu(p_{12})=(m_3)$ and the current marking of the ANet is $(0, MNet, 0, 0, m_3, 0, 0)$.

3.4 Process Net (*PNet*)

Definition 4: A process net is defined as a tuple $PNet = (ToP, \mu)$, where

PNet is a process net representing more abstract production processes of producing a family of end-products;

 $ToP = (G, SoTma, \Sigma^P, \alpha, \beta, E, \tau)$ is a process type (a special assembly type) with

- G is the basic PN structure;
- SoTma is a set of manufacturing types and assembly types;
- Σ^P is a finite set of color sets or token types;
- $\alpha: P \mapsto SoTma \cup M_{\Sigma^P}$ is an assignment function that maps a place, p, to manufacturing or assembly types or a set of colors, $\alpha(p)$;
- $\beta: T \mapsto (\Sigma^P \cup \{\varepsilon\}) \times (\Sigma^P)$ is an assignment function that maps a transition, t, to a set of color pairs, $\beta(t)$, such that a pair declares that the transition t must be synchronized internally with transitions of the nested nets, except if the pair is $(\{\varepsilon\}, \Sigma^P)$ with ε representing nil;
- $E: A \times \Sigma^P \mapsto \vee M_{\Sigma^P} \cup \vee (\wedge M_{\Sigma^P} \to M_{\Sigma^P} \circledast + \tau)$ is an arc expression function that defines the timed and untimed arc expressions for arcs with respect to transition colors; and
- $\tau \in \Re^+ \cup 0$ is a set of non-negative real numbers representing time delays;

 $\mu: P \mapsto M_{\alpha(p)}, \forall p \in P$ is a marking function specifying the distribution of nested nets and colored tokens in all places.

A *PNet* is defined to represent the more abstract production processes of producing a family of end-products. Essentially, a *PNet* is a special kind of *ANet*. It involves only the child items at the immediate lower level of BOM structures of end-products. As with an *ANet*, material buffer places in a *PNet* are defined for parts and/or assemblies. They contain tokens, which are *MNets | ANets* of their child parts/assemblies.

Figure 4 shows an example of a *PNet*. In this net, the set of places is $\{p_{15}, \cdots, p_{28}\}$; the transition set is $\{t_9, \cdots, t_{17}\}$; the set of arc is $\{(p_{15}, t_9), \cdots, (t_{17}, p_{28})\}$; and the set of color sets is $\{\{Y_1, Y_2\}, \cdots, \{FP_1, FP_2, FP_3\}\}$. Y_1 and Y_2 represent two assembly variants of assembly family Y; $X_p X_2$ and X_3 are defined for assembly variants of assembly family X; Z_1 and Z_2 describe two part variants of part family Z; XY_1, XY_2 and XY_3 denote WIPs, which are formed by the two assembly families X and Y; and FP_1, FP_2 and FP_3 represent 3 end-products to be produced. Table 3 lists the set of places, represented system elements and tokens, nested nets.

Both the color pairs assigned to transitions and the timed/untimed arc expressions with respect to transition colors are shown in the figure. The color pairs of t_9 , t_{10} and t_{15} indicate that if enabled, they must fire simultaneously with the corresponding transitions of the lower level nested nets. The marking of the places of the *PNet* is $\mu(p_{16}) = (ANet)$, $\mu(p_{19}) = (m_4)$, $\mu(p_{21}) = (m_5)$ and $\mu(p_{27}) = (m_6)$, and the current marking of the *ANet* is $(0, ANet, 0, 0, m_4, 0, m_5, 0, 0, 0, 0, m_6, 0, 0)$.

4. NESTED NET SYSTEM AND SYSTEM EVOLUTION

Consistent with the hierarchical structure of an end-product, production configuration can be regarded as a recursive process of configuring process elements through various levels of abstraction. For an end-product with an *N*-level hierarchy, production configuration is carried out for various product items at each level. Figure 5 concludes a parallelism between the product hierarchy and production configuration. For an end-product with an *N*-level hierarchy, the iterative process refinement will form a process hierarchy with *N* levels, where the processes at the *N*-th level are the detailed manufacturing processes for parts at the *N*-th level of the product hierarchy. Since parts can be at each level of the product hierarchy except the first level (i.e., Level 1).

While *MNets*, *ANets* and *PNets* are defined to model processes of parts, assemblies and end-products, where only the immediate child items are considered, a multilevel nested net system is defined to capture complete production processes of end-products. By complete, we mean that all processes for component items listed in a product's BOM hierarchy are considered.

4.1 Nested Net System

Definition 5: A multilevel nested net system is defined as a triple, MINNS = (PNet, M, A), where

MINNS is the multilevel nested net system for modeling the complete production processes of end-products;

PNet is the process net describing the abstract production processes of end-products;

 $M = \{MNet_i\}_{N^M}$ is a finite set of MNets, $M = M_1 \cup M_2 \cdots \cup M_{N^M}$, where each

 $M_o = \{MNet_j\}_{N^{M_o}}$, $\forall o \in [I, N^M]$ is a finite set of *MNets* nested in such places that are in the same nets at the immediate higher levels; and

 $A = \left\{ANet_i\right\}_{N^A} \quad \text{is a finite set of} \quad ANets \; , \quad A = A_I \cup A_2 \cdots \cup A_{N^A} \; , \quad \text{where each} \\ A_p = \left\{ANet_j\right\}_{N^{A_p}}, \; \forall p \in \left[I, N^A\right] \; \text{represents the set of} \quad ANets \; \text{ that are nested in the same nets} \\ \text{at the immediate higher levels}.$

Performing as an abstraction mechanism, an *MINNS* facilitates the configuration of processes with right amount of details. Within *MINNS*, the highest level is the *PNet*, while a number of *MNets* and *ANets* are located at the second level. Each of these nets provides more details for processes of input items involved in *PNet*. The nets at any lower level provide detailed descriptions of assembly and manufacturing processes nested in places of nets at their immediate higher level. At the lowest level of each path, all nets are *MNets*, whilst a mixture of *MNets* and *ANets* can be found at any arbitrary level. In accordance with an *N* level BOM structure, Figure 6 demonstrates an *N* level net system with nested *MNets* and *ANets*. For clarity, not all nested nets and arc expressions are shown.

4.2 System Evolution

Following production practice, in an *MINNS*, *MNets* nest in places of *ANets* and/or *PNet* and *ANets* nest in places of higher level *ANets* and/or *PNet*. Consequently, configuration of a complete production process for an end-product necessitates interaction of nested nets and host nets. In this work, transition synchronization is introduced to enable interaction of nets at different levels. Transition synchronization is declared by the color pairs/triples of net transitions.

Lomazova (2000) states that transitions of nets at different levels may fire autonomously or are synchronized in NPNs. More specifically, two or more transitions with same color pairs and/or triples must be synchronized. Bearing in mind these characteristics, we define rules for enabling and firing transitions in the net system. While transitions of any net can fire iff they are enabled, different enabling rules are applied for *MNets*, *ANets* and the *PNet*, as elaborated below.

4.2.1 Transition enabling in MNets

Rule 1: A transition, t, of an MNet is enabled with respect to a color pair, cp, and fires autonomously without additional conditions if $\forall p, E((p,t), cp) \subseteq \mu(p)$ and

$$cp = (\Sigma^M, \varepsilon).$$

For example, in the *MNet* in Figure 2, the marking of the three input places of transition, t_1 , is $\mu(p_1) = \{a_1\}$, $\mu(p_2) = \{b_1\}$ and $\mu(p_7) = \{m_1\}$, respectively. In addition, $E((p_1, t_1), (A_1, \varepsilon)) \subseteq \mu(p_1) = \{a_1\}$, $E((p_2, t_1), (A_1, \varepsilon)) \subseteq \mu(p_2) = \{b_1\}$ and $E((p_7, t_1), (A_1, \varepsilon)) \subseteq \mu(p_7) = \{m_1\}$. Thus, t_1 meets the above enabling condition and fires autonomously.

Rule 2: A transition, t, of an MNet is enabled and fires simultaneously with an output transition of the place, where the MNet is nested, of the higher level ANet or PNet if $\forall p, E((p,t), cp) \subseteq \mu(p)$ and $cp = (\Sigma^M, \Sigma^M)$.

For example, in Figure 3, the *MNet* is nested in place p_9 of the *ANet* and t_6 is the output transition of p_9 . In the *MNet*, the marking of the input place, p_3 , of the transition, t_2 , is $\mu(p_3) = \{A_i\}$. Further, $E((p_3, t_2), (A_i, A_i)) \subseteq \mu(p_3) = \{A_i\}$. Thus, t_2 fulfills the simultaneous enabling and firing condition with respect to the transition color pair, (A_i, A_i) .

Rule 3: A transition, t, of an ANet is enabled with respect to a color triple, ct, and fires autonomously without additional conditions if $\forall p, E((p,t),ct) \subseteq \mu(p)$ and $ct = (\varepsilon, \Sigma^A, \varepsilon)$.

In Figure 3, if two more tokens, B_I and A_I , are added into the two input places, p_{I0} and p_{I1} , of transition, t_7 , of the *ANet*, respectively, $\mu(p_{I0}) = \{B_I\}$, $\mu(p_{I1}) = \{A_I\}$ and $\mu(p_{I2}) = \{m_3\}$. Moreover, $E((p_{I0}, t_7), (\varepsilon, X_I, \varepsilon)) \subseteq \mu(p_{I0}) = \{B_I\}$, $E((p_{II}, t_7), (\varepsilon, X_I, \varepsilon)) \subseteq \mu(p_{I1}) = \{A_I\}$ and $E((p_{I2}, t_7), (\varepsilon, X_I, \varepsilon)) \subseteq \mu(p_{I2}) = \{m_I\}$. Accordingly, t_7 satisfies the enabling rule and fires autonomously.

Rule 4: A transition, t, of an ANet is enabled and fires simultaneously with a transition bearing a $cp = (\Sigma^A, \Sigma^A)$ or $ct = (\varepsilon, \Sigma^A, \Sigma^A)$ of the lower level MNet or ANet that nests in an input place of t, if $\forall p, E((p,t), ct) \subseteq \mu(p)$ and $ct = (\Sigma^A, \Sigma^A, \varepsilon)$.

In Figure 3, t_6 of the *ANet* meets the firing rule and fires simultaneously with t_2 of the nested *MNet*.

Rule 5: A transition, t, of an ANet is enabled and fires simultaneously with an output transition of a place, where the ANet is nested, of the higher level ANet or PNet, if $\forall p, E((p,t), ct) \subseteq \mu(p)$ and $ct = (\varepsilon, \Sigma^A, \Sigma^A)$.

In Figure 3, if a token, X_I , is added into the input place, p_{I3} , of transition, t_8 , of the *ANet*, $\mu(p_{I3}) = \{X_I\}$ and $E((p_{I3}, t_8), (\varepsilon, X_I, X_I)) \subseteq \mu(p_{I3}) = \{X_I\}$. Thus, t_8 with respect to (ε, X_I, X_I) meets the enabling condition and fires simultaneously with other transitions of the host nets.

4.2.3 Transition enabling in PNet

Rule 6: A transition, t, of a PNet is enabled with respect to a color pair, cp, and fires autonomously without additional conditions if $\forall p, E((p,t), cp) \subseteq \mu(p)$ and $cp = (\varepsilon, \Sigma^P)$.

In the *PNet* in Figure 4, if two tokens Z_1 and XY_1 are added into the two input places p_{24} and p_{25} of transition t_{16} of the *PNet*, respectively, t_{16} meets the above enabling condition and fires autonomously. The same can be applied to other transitions, e.g., t_{11} , t_{17} .

Rule 7: A transition, t, of a PNet is enabled and fires simultaneously with a transition bearing a $cp = (\Sigma^M, \Sigma^M)$ or $ct = (\varepsilon, \Sigma^A, \Sigma^A)$ of the lower level MNet or ANet that nests in an input place of t, if $\forall p, E((p,t), cp) \subseteq \mu(p)$ and $cp = (\Sigma^P, \Sigma^P)$.

In the *PNet* in Figure 4, a color pair assigned to t_{10} is (X_1, X_1) . Suppose the *ANet* in Figure 3 is nested in p_{16} of the *PNet*. If t_8 in the *ANet* in Figure 3 is enabled, t_{10} and t_8 fire simultaneously.

4.2.4 Transition firing

While the firing of transitions of any net does not provoke the transfer of nested nets, that is, the nested nets remain in the same places before and after simultaneous transition firing, tokens, other than the nested nets, are created and removed as follows:

Rule 8: The firing of an enabled transition, t, in nets at any arbitrary level modifies markings by 1) generating tokens in the output places as specified by E((t, p), cp) or E((t, p), ct); and 2) removing tokens from the input places as specified by E((p, t), cp) or E((p, t), ct).

For instance, in Figure 3, the firing of t_6 generates a token (either A_1 , A_2 or A_3) in P_{11} . While the *MNet* remains in P_9 after firing t_2 , it evolves into a new state, i.e., a token is generated in P_4 and a token is removed from the input place P_5 .

5. INDUSTRIAL EXAMPLE

The NCTPNs formalism developed in this study is tested using an industrial example of vibration motors for mobile phones. Although vibration motors themselves are not complex, they are typical examples of customized products. The high variety of individualized mobile phones assumes different technical specifications, which, in turn, lead to unique design requirements of vibration motors (e.g., dimensions, components).

5.1 Construction of PNet, ANets, MNets and Net System

Figure 7 shows the common product structure of a motor family. This motor family assumes several part types, including: rubber holder (Rh), weight (Wt), frame (Fm), bracket a (Ba), bracket b (Bb), terminal (Tl), magnet (Mt), coil (Cl), shaft (St), commutator (Ct) and tape (Tp). For each part type, there are a number of variants catering for specific mobile phone requirements. In turn, the different combinations of different/same specific parts form a variety of assemblies belonging to same families. According to design requirements, individual motors may not contain a specific variant from each part family. In other words, not all part types are assumed by each individual motor variant.

By following the common process structure underpinning the motor family's process platform, the PNet is constructed, as shown in Figure 8. In the PNet, the items involved include AAs (armatureassies), FAs (frameassies), BAs (bracketassies), ABs (abassies; WIPs formed by AAs and BAs), MBs (mainbodies; WIPs formed by ABs and FAs) and VMs (final vibration motors). Three different machines (including the corresponding tools and fixtures) are used to process AAs and BAs for forming Abs; two different machines to process ABs and FAs for forming MBs; one workstation (including 2 operators, tools and fixtures) to assemble final motors from MBs, Rhs and/or Wts. As shown in the PNet, first one of the 3 alternative machines (m_1 , m_2 and m_3) is used to process AAs and BAs, then one of the 2 machines (m_4 and m_5) is used to work on the formed WIPs (i.e., ABs) and FAs, at last final motors are assembled at the workstation (m_6).

The places, represented system elements and contained colored tokens/lower level nets, are listed in Table 4. The color pairs assigned to each transaction are shown in Figure 8. As shown in the figure, it takes different/same machines different time durations to complete operations on same/different input items. For illustrative simplicity, the colored tokens in

relation to 3 motor variants (VM_1, VM_2) and VM_3 are shown in Figure 8 and Table 4, and in the figures and tables in the following case study as well. The timed arc expression of arc (t_{13}, p_{20}) shows that VM_1 assumes both Wt and Rh, VM_2 contains a Wt, and VM_3 includes an Rh.

As shown in Figure 7, the motor family has 4 assembly families: BA, FA, AA and CA. Similar with constructing the *PNet*, the *ANet* of the 4 assembly families have been constructed. Figure 9 shows the *ANet* of BA family. Table 5 lists the places, colored tokens/lower level nets and the represented system elements.

Also shown in Figure 9 are the color triples assigned to each transition. According to these colored triples, transitions of this ANet fires autonomously, or simultaneously with either transitions of nested nets residing in places p_{22} , p_{25} and p_{31} or transitions of the PNet in Figure 8. For example, the set of color triples of t_{26} : $\{(\varepsilon, BA_1, BA_1), (\varepsilon, BA_2, BA_2), (\varepsilon, BA_3, BA_3)\}$ indicates that t_2 fires simultaneously with t_{26} of the PNet in Figure 8 after being enabled by either one of the 3 tokens: BA_1 , BA_2 and BA_3 which represent 3 BA variants.

In general, two types of assembly operations are involved in producing BAs, as shown in Figure 9. Either of the 2 alternative machines (m_7 and m_8) is used to form Babs (WIPs) by joining Bas and Bbs first. Then, one of another three alternative machines (m_9 , m_{10} and m_{11}) is used to process Babs and Tls for assembling BAs. However, if a BA variant, e.g., BA_1 in the figure, consists of a Ba and Bb variants, the second type of assembly operations involving Tls is not included. To capture and model this kind of process change, a 0 time delay is defined in the timed arc expression to describe the non-execution of the corresponding operations. For instance in Figure 9, since BA_1 does not include a Tl variant, the corresponding statement: $Bab_1 \rightarrow BA_1 @+0$ is defined as a choice of the timed arc expression of arc (t_{22}, p_{37}) . Accordingly, transition t_{26} fires immediately with the presence of Bab_1 . If enabled by Bab_2 or Bab_3 along with the different combination of m_9 , m_{10} and m_{11} , t_{26} fires after the elapse of time durations indicated by the corresponding time

delays.

In a similar way, *MNets* have been constructed in accordance with part families produced in house. Figure 10 shows the *MNet* of Ba family and Table 6 lists the places, colored tokens and the corresponding system elements. As shown in the *MNet*, the color pairs assigned to each transition require synchronized and autonomous transition firing. For instance, as indicated by the color pairs: $\{(Ba_1, Ba_1), (Ba_2, Ba_2), (Ba_3, Ba_3)\}$, t_{32} must fire simultaneously with transition t_{15} of the *ANet* in Figure 9 with the presence of either Ba_1 or Ba_2 or Ba_3 , whilst t_{31} fire autonomously after being enabled by either $Ba_1.1$ or $Ba_2.1$ or $Ba_3.1$ together with m_{14} .

Manufacturing Ba variants requires two types of operations, as shown in the *MNet*. While two alternative machines $(m_{12} \text{ and } m_{13})$ are available to execute the first operation, only one machine (m_{14}) is able to carry out the second operation. The tokens, $Ba_1.1, Ba_2.1$ and $Ba_3.1$, in the raw material buffer p_{39} represent raw materials ready to be processed. They are defined according to the 3 Ba variants to be produced: Ba_1, Ba_2 and Ba_3 .

Based on the constructed *MNets*, *ANets* and *PNet*, the multilevel system of production configuration is obtained as shown in Figure 11. Due to the space constraints, the timed/untimed arc expressions and the color triples/pairs assigned to transitions are not shown. As shown in the figure, the *PNet* is at the highest level; 3 *ANets* for producing assembly families AA, BA and FA are located at the second level; 1 *ANet* for producing CA family and 3 *MNets* for manufacturing part families Ba, Bb and Tl are at the third level; and 2 *MNets* for fabricating part families Tp and Cl are located at the lowest level. Each lower level nested net is linked with the corresponding tokens residing in places of higher level nets.

5.2 System Analysis

After construction, all models are analyzed to check 1) whether or not they are correct; and 2) whether or not they reflect logically the corresponding system operations. Jensen (1995) introduces several methods to verify models with respect to dynamic properties.

Among these, P-invariant analysis, which we adopt in this study, is of particular interest to most researchers due to its easy-understandability and implementation. Several P-invariant can be identified in each *MNet*, *ANet* and the *PNet*. The total number of busy machines and idle machines gives a P-invariant. In other words, in any system state, the total number of tokens appearing in idle machine places, conceptual machine places and machine processing input item places is always the same. Another P-invariant relates to the input items ready to be processed and the corresponding output items. This P-invariant is obtained through mapping the output items to the input items, i.e., representing the output items by the input items.

In addition, in view of the impact of deadlock and conflicts on the logical operations of system models (Jiang et al., 1999), we perform deadlock and conflict analysis in this work. Following practice in high variety production, this study does not draw attention to the sharing of a single machine or resources among multiple tasks - a typical characteristic of previous studied manufacturing systems, e.g., flexible manufacturing systems (Kaighobadi and Venkatesh, 1994). Accordingly, the potential conflicts resulting from resources sharing do not exist in the constructed PN models. Moreover, the definition of colored tokens and arc expressions resolves possible conflicts regarding the firing of transitions with respect to different tokens in all *MNets*, *ANets* and the *PNet*. In other words, transitions fire according to the match between colored tokens in the input places and these are assigned to transitions. Further, with the presence of alternative colored tokens that can enable the same transitions, transition firing can be determined by adopting preferred rules, e.g., the shortest delay times.

Wang and Wu (1998) describe a procedure of Deadlock Detection Algorithm (DDA) for deadlock analysis, as shown in Figure 12. It starts from setting an initial state marking μ_0 and a goal state marking μ_g for a PN model being analyzed. The incidence matrix W is constructed based on the difference of arc expressions of output arcs and input arcs from places to transitions with respect to colored tokens. When the system model is in a state k, the enabled transitions along with the colored tokens are identified. Subsequently, one

enabled transition is selected to fire in accordance with the colored tokens. The characteristic vector $\overline{S_k}$ of a transition firing sequence S_k when the model is in a state k is obtained through setting an entry (i.e., the corresponding colored tokens) for the transition selected to fire and 0 for all other transitions. With μ_k (the marking of the system in a state k) and $\overline{S_k}$, the system marking of the next state μ_{k+l} can be obtained. If $\mu_{k+l} = \mu_g$ in a certain state k, the procedure ends. The conclusion that the model is live and deadlock free can then be drawn. The procedure also indicates that if no enabled transitions can be identified and μ_g is not achieved yet when the PN model is in a state k, the model is not correct and needs to be modified.

In this study, we apply the above DDA procedure to the constructed MNets, ANets and the PNet. For each net, a feasible firing sequence has been obtained. Each sequence starts from a predetermined initial state marking and ends at a predetermined goal state marking through firing enabled transitions. Figure 13 shows a transition firing sequence by applying the DDA procedure to the MNet in Figure 10. This firing sequence leads to the goal state $\mu_{15} = (0, m_{13}, m_{12}, 0, 0, 0, 0, m_{14}, 0, Ba_1 + Ba_2 + Ba_3)$ from the initial state: $\mu_0 = (Ba_1.1 + Ba_2.1 + Ba_3.1, m_{13}, m_{12}, 0, 0, 0, m_{14}, 0, 0)$. Also shown in the figure are the fired transitions along with the corresponding colored tokens.

5.3 Application Results

As indicated by the nets in Figures 8, 9 and 10, more than one production process consisting of different combination of machines is feasible to fulfill each of the 3 motor variants: VM_1 , VM_2 and VM_3 in Figure 8. In this regard, minimizing the completion time of the last operation of producing the three vibration motors is chosen as the production objective and thus the selection is made based on this objective. In the application, we adopt the shortest delay times to fire transitions, which are enabled by alternative colored tokens. Table 7 presents the result of configured production processes in terms of machines in line with items in the nets in Figures 8, 9 and 10.

6. FURTHER DISCUSSIONS

Recognizing the significance of production configuration for producing product families while maintaining production stability and efficiency, this paper studies the underlying logic for configuring production processes from a process platform. In view of the importance of PN techniques in shedding light on system dynamic behaviors, we apply PN techniques to model production configuration. More specifically, we develop a new formalism of NCTPNs by integrating the advantages of CPNs, TPNs and NPNs.

In the formalism, the adoption of colored tokens and time delays of transition firing facilitates the modeling of variety handling, constraint satisfaction and process selection. The definition of a reconfiguration mechanism, consisting of reconfigurable transitions, inhibitor arcs and conceptual machine places, tackles the many inherent process variations when building system models. The incorporation of net nesting addresses granularity issues in production configuration. Moreover, three types of nets: *MNets*, *ANets* and a *PNet* are defined to represent the processes of manufacturing parts, producing assemblies and forming final products, respectively. And a multilevel nested net system is defined to capture the entire process of production configuration. The results of the industrial example of vibration motors for mobile phones show the potential of the NCTPNs formalism in revealing the logic of production configuration. The advantages of the proposed formalism along with some managerial implications are further discussed as follows:

1) NCTPNs possess a combined power of top-down decomposition and bottom-up implementation through net nesting. In the realistic applications, top-down decomposition and bottom-up implementation are important approaches for system design and planning and, they provide managers with an ability to carry out system analysis and synthesis in an interactive and integrative way. This especially true when the system in consideration is large and complex. Due to the high product and process variety, their relationships and the many constraints/restrictions, production configuration is complex. Therefore, NCTPNs are convenient and promising for modeling production configuration. They also enable managers to make right decisions in configuration by presenting an overall

interactive picture of higher level more abstract processes and lower level more detailed processes.

- 2) The nested *ANets* and *MNets* in the proposed formalism are independent of one another. There is no direct connection between two nets that are nested in different places of nets at a same level of the configuration hierarchy. As a result, one nested net can be changed without causing any structural changes to other nets. Hence, it is easy for managers to maintain and modify the production configuration models based on the proposed NCTPNs.
- As a graphical modeling tool, the NCTPNs provide managers with a visualization of the entire process of production configuration. Transaction synchronization is used to capture the interactions between lower level nested nets and higher level host nets. Thus, the impact on the relevant higher level nets caused by the changes of lower level nets, be it a change of net structures or of process elements configured, can be intuitively captured. This, in turn, offers an easier understanding of the impacts caused by configuration changes on production configuration to managers.
- 4) NCTPNs can be used to facilitate the development of a production rule-based computational system which implements production configuration. There are some correspondence between NCTPNs and rule-based systems. The tokens carrying important information pertaining to the represented objects are equivalent to facts in a rule-based system. Transaction firing rules relate to rules in a rule-based system. Transition firing corresponds to rule execution in a computational system. Therefore, the analogy between the NCTPNs and rule-based systems can facilitate the development of a computational system to implement production configuration.

The scope of this study is limited to the development of a proper formalism to model production configuration, and does not extend to the computational implementation of production configuration based on the proposed model. In addition, this study does not address the impact of the different combinations of production objectives and transition firing rules on the final configuration results in that different companies may adopt different combinations according to their own production characteristics. Accordingly, future research may be extended to address them. And more efforts might be made to develop a rule-based

computational system to implement production configuration based on the NCTPNs.



REFERENCES

Abdallaht, I.B., Elmaraghyj, H.A. and Elmekkawyi, T., 2002, Deadlock-free scheduling in flexible manufacturing systems using Petri nets, *International Journal of Production Research*, **40**(12), pp. 2733-2756.

Adamous, M., Zerhouni, S.N. and Bourjault, A., 1998, Hierarchical modeling and control of flexible assembly systems using object-oriented Petri nets, *International Journal of Computer Integrated Manufacturing*, **11**(1), pp. 18–33.

Blackhurst, J., Wu, T. and O'Grady, P., 2005, PCDM: A decision support modeling methodology for supply chain, product and process design decisions, *Journal of Operations Management*, **23**, pp. 325-343.

Bowden, F.D.J., 2000, A brief survey and synthesis of the roles of time in Petri nets, *Mathematical and Computer Modeling*, 31, pp. 55-68.

Chan, F.T.S., Zhang, J. and Li, P., 2001, Modeling of integrated, distributed and cooperative process planning system using an agent-based approach, *Proceedings of the Institution of Mechanical Engineers*, Part B: *Journal of Engineering Manufacture*, **215**, pp. 1437-1451.

Chin, K.S., Zu, X., Mok, C.K. and Tam, H.Y., 2006, Integrated definition language 0 (IDEF) and colored Petri nets (CPN) modeling and simulation tool: A study on mould-making processes, *International Journal of Production Research*, **44**(15), pp. 3179-3205.

De Lit, P., Delchambre, A. and Henrioud, J.M., 2003, An integrated approach for product family and assembly system design, *IEEE Transactions on Robotics and Automation*, **19**, pp. 324–333.

Dotoli, M. and Fanti, M.P., 2004, Colored timed Petri net model for real-time control of automated guided vehicle systems, *International Journal of Production Research*, **42**(9), pp. 1787-1814.

Fixson, S.K., 2005, Product architecture assessment: A tool to link product, process, and supply chain design decisions, *Journal of Operations Management*, **23**, pp. 345-369.

Garg, A., 1999, An application of designing products and processes for supply chain management, *IIE Transactions*, **31**, pp. 417-429.

Gupta, S. and Krishnan, V., 1998, Product family-based assembly sequence design methodology, *IIE Transactions*, **30**, pp. 933–945.

Hiraishi, K., 2000, A Petri-net-based model for the mathematic analysis of multi-agent systems, *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC'2000)*, Oct. 2000, Nashville, TN, 4, pp. 3009–3014.

Huang, G.Q., Zhang, X.Y. and Liang, L., 2005, Towards integrated optimal configuration of platform products, manufacturing processes, and supply chains, *Journal of Operations Management*, **23**, pp. 267-290.

Huang, S.H., Liu, Q. and Musa, R., 2004, Tolerance-based process plan evaluation using Monte Carlo simulation, *International Journal of Production Research*, **42**, pp. 4871-4891.

Jensen, K., 1995, Colored Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Vol. 2, New York: Springer.

- Jiang, Z., Zuo, M.J., Fung, R.Y.K. and Tu, P.Y., 2000, Temporized colored Petri nets with changeable structure (CPN-CS) for performance modeling of dynamic production systems, *International Journal of Production Research*, **38**, pp. 1917–1945.
- Jiang, Z., Zuo, M.J., Tu, P.Y. and Fung, R.Y.K., 1999, Object-oriented Petri nets with changeable structure (OPNs-CS) for production system modeling, *International Journal of Advanced Manufacturing Technology*, **15**, pp. 445-458.
- Jiao, J., Zhang, L. and Pokharel, S., 2007, Process platform planning for variety coordination from design to production in mass customization manufacturing, *IEEE Transactions on Engineering Management*, **54**, pp. 112-129.
- Kaighobadi, M. and Venkatesh, K., 1994, Flexible manufacturing systems: An overview, *International Journal of Operations & Production Management*, **14**(4), pp. 25-49.
- Li, X.O. and Lara-Rosano, F., 1999, Modeling an electronic component manufacturing system using object oriented colored Petri nets, *Proceedings of the 6th IEEE International Conference on Electronics, Circuits, and Systems*, Pafos, Cyprus.
- Liu, D.S., Wang, J.M., Chan, S.C.F., Sun, J.G. and Zhang, L., 2002, Modeling workflow processes with colored Petri nets, *Computers in Industry*, **49**(3), pp. 267-281.
- Lomazova, I.A., 2000, Nested Petri nets: A formalism for specification and verification of multi-agent distributed systems, *Fundamenta Informaticae*, **43**, pp.195-214.
- Lopez-Mellado, E. and Almeyda-Canepa, H., 2003, A three-level net formalism for the modeling of multiple mobile robot systems, *International Journal of Computer Integrated Manufacturing*, **18**(2-3), pp. 122–136.
- Lu, Q. and Botha, B., 2006, Process development: A theoretical framework, *International Journal of Production Research*, **44**(15), pp. 2977-2996.
- Martinez, M.T., Favrel, J. and Ghodous, P., 2000, Product family manufacturing plan generation and classification, *Concurrent Engineering: Research and Applications*, **8**, pp. 12-22.
- Nandula, M. and Dutta, S.P., 2000, Performance evaluation of an auction-based manufacturing system using colored Petri nets, *International Journal of Production Research*, **38**, pp. 2155–2171.
- Peterson, J.L., 1981, *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, Englewood Cliffs, N.J.
- Petersen, K.J., Handfield, R.B. and Ragatz, G.L., 2005, Supplier integration into new product development: Coordinating product, process and supply chain design, *Journal of Operations Management*, **23**, pp. 371-388.
- Ramachandani, C., 1974, Analysis of asynchronous concurrent systems by timed Petri nets. Technical Report MAC TR 120, MIT, Cambridge.
- Raman, N. and Chhajed, D., 1995, Simultaneous determination of product attributes and prices, and production processes in product-line design, *Journal of Operations Management*, **12**, pp. 187-204.
- Ravi Raju, K. and Krishnaiah Chetty, O.V., 1993, Design and evaluation of automated guided vehicle systems for flexible manufacturing systems: An extended timed Petri net-based approach, *International Journal of Production Research*, **31**(5), pp. 1069-1096.
- Reddy, G.B., Murty, S.S.N. and Ghosh, K., 1993, Timed Petri net: An expeditious tool for modeling and analysis of manufacturing systems, *Mathematical and Computer Modeling*, **18**(9), pp. 17-30.

Sanderson, S.W. and Uzumeri, A.V., 1996, *Managing Product Families*. Irwin Press.

Schierholt, K., 2001, Process configuration: Combining the principles of product configuration and process planning, *Artificial Intelligence for Engineering Design*, *Analysis and Manufacturing*, **15**, pp. 411-424.

Thomas, J.P., Nissanke, N. and Baker, K.D., 1996, A hierarchical Petri net framework for the representation and analysis of assembly, *IEEE Transactions of Robotics and Automation*, **12**(2), pp. 268–279.

Tielemans, P.F.J., 1995, Lead Time Performance in Manufacturing Systems. PhD Dissertation, Rotterdam School of Management, Eburon, Delft, The Netherlands.

Turner, W.C., Mize, J.H., Caser, K.E. and Nazemetz, J.W., 1992, *Introduction to Industrial and Systems Engineering*, 3rd Edition. Prentice Hall.

Wang, L.C. and Wu, S.Y., 1998, Modeling with colored timed object-oriented Petri nets for automated manufacturing systems, *Computers & Industrial Engineering*, **34**(2), pp. 463-480.

Williams, C.B., Allen, J.K., Rosen, D.W. and Mistree, F., 2007, Designing platforms for customizable products and processes in markets of non-uniform demand, *Concurrent Engineering: Research and Applications*, **15**, pp. 201-216.

Zhang, L., 2007, Process Platform-based Production Configuration for Mass Customization. PhD Dissertation, Division of Systems and Engineering Management, Nanyang Technological University, Singapore.

LIST OF TABLES

Table 1: Places, system elements and colored tokens in the net in Figure 2

Places	System Elements	Colored Tokens
p_{I}	A raw material buffer	$\alpha(p_1) = \{a_1, a_2\}$
p_2	A raw material buffer	$\alpha(p_2) = \{b_1, b_2\}$
p_3	A busy machine	$\alpha(p_3) = \{A_1, A_2, A_3\}$
p_4	A part buffer	$\alpha(p_4) = \{A_1, A_2, A_3\}$
p_5	An idle machine	$\alpha(p_5) = \{m_1\}$
p_6	An idle machine	$\alpha(p_6) = \{m_2\}$
p_7	A conceptual machine	$\alpha(p_7) = \{m_1, m_2\}$

Table 2: Places, system elements and colored tokens in the net in Figure 3

Places	System Elements Nets/Colored Tokens	
p_{s}	An input part buffer	An MNet for producing part family A
p_{g}	An input part buffer	An MNet for producing part family B
p_{10}	Parts ready to be processed	$\alpha(p_{10}) = \{B_1, B_2, B_3\}$
$p_{{\scriptscriptstyle II}}$	Parts ready to be processed	$\alpha(p_{II}) = \{A_1, A_2, A_3\}$
$p_{_{12}}$	An idle machine	$\alpha(p_{12}) = \{m_3\}$
p_{I3}	A busy machine	$\alpha(p_{13}) = \{X_1, X_2, X_3\}$
<i>p</i> ₁₄	An assembly buffer	$\alpha(p_{14}) = \{X_1, X_2, X_3\}$

Table 3: Places, system elements and the colored tokens/nets in the net in Figure 4

Places	System Elements	Nets/Colored Tokens
<i>p</i> ₁₅	An input assembly buffer	An ANet for producing assembly family Y
$p_{{\scriptscriptstyle 16}}$	An input assembly buffer	An ANet for producing assembly family X
<i>p</i> ₁₇	Assemblies ready to be processed	$\alpha(p_{ij}) = \{Y_i, Y_2\}$
$p_{{}_{I8}}$	Assemblies ready to be processed	$\alpha(p_{18}) = \{X_1, X_2, X_3\}$
<i>p</i> ₁₉	A conceptual machine	$\alpha(p_{19}) = \{m_4, m_5\}$
P 20	An idle machine	$\alpha(p_{20}) = \{m_{_{4}}\}$
<i>P</i> 21	An idle machine	$\alpha(p_{2i}) = \{m_{s}\}$
P 22	A busy machine	$\alpha(p_{22}) = \{XY_1, XY_2, XY_3\}$
<i>p</i> ₂₃	An input part buffer	An MNet for producing part family Z
P 24	Parts ready to be processed	$\alpha(p_{24}) = \{Z_1, Z_2\}$
P 25	WIPs formed by two assembly families	$\alpha(p_{25}) = \{XY_1, XY_2, XY_3\}$
$p_{_{26}}$	A busy machine	$\alpha(p_{26}) = \{FP_{1}, FP_{2}, FP_{3}\}$
<i>p</i> ₂₇	An idle machine	$\alpha(p_{27}) = \{m_{6}\}$
P 28	An end-product buffer	$\alpha(p_{28}) = \{FP_1, FP_2, FP_3\}$

Table 4: The places, represented system elements and colored tokens/nets

Places	System Elements	Nets/Colored Tokens
$p_{_{1/4/11}}$	Input assembly buffers for AAs/BAs/FAs	ANets for producing assembly families AA/BA/FA
$p_{_2}$	AAs are ready to be processed	$\alpha(p_2) = \{AA_1, AA_2, AA_3\}$
$p_{_3}$	BAs are ready to be processed	$\alpha(p_2) = \{BA_1, BA_2, BA_3\}$
$p_{\scriptscriptstyle 5/13}$	Conceptual machines for assembling ABs/MBs	$\alpha(p_s) = \{m_1, m_2, m_3\} / \alpha(p_{13}) = \{m_4, m_5\}$
$p_{_{6/7/8}}$	Alternative idle machines for forming ABs	$\alpha(p_s) = \{m_i\} / \alpha(p_\tau) = \{m_i\} / \alpha(p_s) = \{m_s\}$
p_{9}	Alternative machines processing AAs & BAs	$\alpha(p_{\scriptscriptstyle g}) = \{AB_{\scriptscriptstyle I}, AB_{\scriptscriptstyle 2}, AB_{\scriptscriptstyle 3}\}$
$p_{{\scriptscriptstyle I}{\scriptscriptstyle 0}}$	ABs are ready to be processed	$\alpha(p_{10}) = \{AB_1, AB_2, AB_3\}$
$p_{{}_{I2}}$	FAs are ready to be processed	$\alpha(p_{12}) = \{FA_1, FA_2\}$
<i>p</i> _{14/15}	Alternative idle machines for assembling MBs	$\alpha(p_{\scriptscriptstyle 14}) = \{m_{\scriptscriptstyle 4}\} / \alpha(p_{\scriptscriptstyle 15}) = \{m_{\scriptscriptstyle 5}\}$
$p_{_{16}}$	Alternative machines processing ABs & FAs	$\alpha(p_{16}) = \{MB_1, MB_2, MB_3\}$
$p_{{}_{I7}}$	MBs are ready to be processed	$\alpha(p_{17}) = \{MB_1, MB_2, MB_3\}$
$p_{{}_{I8}}$	A buffer for Wts and Rhs	$\alpha(p_{18}) = \{Wt_1, Wt_2, Rh_1, Rh_2\}$
$p_{{}_{I9}}$	A workstation for assembling final motors	$\alpha(p_{\circ}) = \{m_{\circ}\}$
p_{20}	Final motors are being assembled at the workstation	$\alpha(p_{20}) = \{VM_1, VM_2, VM_3\}$
p_{21}	A buffer for final motors	$\alpha(p_{2i}) = \{VM_1, VM_2, VM_3\}$

Table 5: The places, represented elements and colored tokens/nets

Places	System Elements	Nets/Colored Tokens
$p_{_{22/25/31}}$	Input part buffers for Bas/Bbs/Tls	MNets for producing part families Ba/Bb/Tl
$p_{_{23}}$	Bas are ready to be processed	$\alpha(p_{23}) = \{Ba_1, Ba_2, Ba_3\}$
<i>p</i> ₂₄	Bbs are ready to be processed	$\alpha(p_{24}) = \{Bb_1, Bb_2, Bb_3\}$
p _{26/33}	Conceptual machines for assembling Babs/BAs	$\alpha(p_{26}) = \{m_7, m_8\} / \alpha(p_{33}) = \{m_9, m_{10}, m_{11}\}$
P 27/28	Alternative idle machines for forming ABs	$\alpha(p_6) = \{m_1\} / \alpha(p_7) = \{m_2\} / \alpha(p_8) = \{m_3\}$
p 29	Alternative machines processing Bas & Bbs	$\alpha(p_{29}) = \{Bab_1, Bab_2, Bab_3\}$
$p_{_{30}}$	Babs are ready to be processed	$\alpha(p_{30}) = \{Bab_1, Bab_2, Bab_3\}$
$p_{_{32}}$	Tls are ready to be processed	$\alpha(p_{\scriptscriptstyle 32}) = \{Tl_{\scriptscriptstyle I}, Tl_{\scriptscriptstyle 2}\}$
p _{34/35/36}	Alternative idle machines for producing BAs	$\alpha(p_{34}) = \{m_9\} / \alpha(p_{35}) = \{m_{10}\} / \alpha(p_{36}) = \{m_{11}\}$
<i>p</i> ₃₇	Alternative machines processing Babs & Tls	$\alpha(p_{37}) = \{BA_1, BA_2, BA_3\}$
$p_{_{38}}$	A buffer for BAs	$\alpha(p_{38}) = \{BA_1, BA_2, BA_3\}$

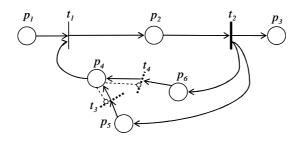
Table 6: The places, represented elements and colored tokens/nets

Places	System Elements	Colored Tokens
$p_{_{39}}$	A raw material buffer for Bas	$\alpha(p_{39}) = \{Ba_1.1, Ba_2.1, Ba_3.1\}$
P 40	A conceptual machines for manufacturing Bas	$\alpha(p_{40}) = \{m_{12}, m_{13}\}$
p _{41/42}	Alternative idle machines for manufacturing Bas	$\alpha(p_{41}) = \{m_{12}\}/\alpha(p_{42}) = \{m_{13}\}$
<i>p</i> ₄₃	Alternative machines processing Ba raw materials	$\alpha(p_{43}) = \{Ba_1.2, Ba_2.2, Ba_3.2\}$
P 44	Ba WIPs are ready to be processed	$\alpha(p_{44}) = \{Ba_1.2, Ba_2.2, Ba_3.2\}$
<i>p</i> ₄₅	The second machine for manufacturing Bas	$\alpha(p_{45}) = \{m_{14}\}$
P 46	The second machine processing Ba WIPs	$\alpha(p_{46}) = \{Ba_1, Ba_2, Ba_3\}$
P ₄₇	A buffer for Bas	$\alpha(p_{47}) = \{Ba_1, Ba_2, Ba_3\}$

Table 7: The configured production processes for the 3 motor variants

Machine(Operation; Output item)			
VM ₁	VM ₃		
m_{I3} (Manufacturing operation; Ba_{I})	m_{13} (Manufacturing operation; Ba_2)	m_{12} (Manufacturing operation; Ba_3)	
m_{14} (Manufacturing operation; Ba_1)	m_{14} (Manufacturing operation; Ba_2)	m_{14} (Manufacturing operation; Ba_2)	
m_{23} (Manufacturing operation; Bb_1)	m_{22} (Manufacturing operation; Bb_2)	m_{23} (Manufacturing operation; Bb_3)	
m_{24} (Manufacturing operation; Bb_1)	m_{24} (Manufacturing operation; Bb_2)	m_{24} (Manufacturing operation; Bb_3)	
m_{26} (Manufacturing operation; Tl_1)	m_{25} (Manufacturing operation; Tl_2)	m_{25} (Manufacturing operation; Tl_2)	
m_7 (Assembly operation; BA_1)	m_8 (Assembly operation; Bab_2)	m_7 (Assembly operation; Bab_3)	
Nonexist	m_{10} (Assembly operation; BA_2)	m_{10} (Assembly operation; BA_3)	
m_{27} (Manufacturing operation; Tp_1)	m_{28} (Manufacturing operation; Tp_2)	m_{27} (Manufacturing operation; Tp_1)	
m_{29} (Manufacturing operation; Cl_1)	m_{29} (Manufacturing operation; Cl_1)	m_{29} (Manufacturing operation; Cl_2)	
m_{30} (Manufacturing operation; Cl_1)	m_{30} (Manufacturing operation; Cl_1)	m_{30} (Manufacturing operation; Cl_2)	
m_{21} (Assembly operation; CA_1)	m_{20} (Assembly operation; CA_2)	m_{20} (Assembly operation; CA_3)	
m_{16} (Assembly operation; AA_{1})	m_{15} (Assembly operation; AA_2)	m_{16} (Assembly operation; AA_{1})	
m_{18} (Assembly operation; FA_{1})	m_{17} (Assembly operation; FA_2)	m_{19} (Assembly operation; FA_3)	
$m_{_{I}}$ (Assembly operation; $AB_{_{I}}$)	m_3 (Assembly operation; AB_2)	m_2 (Assembly operation; AB_3)	
m_s (Assembly operation; MB_1)	m_5 (Assembly operation; MB_2)	m_4 (Assembly operation; MB_3)	
$m_{_{6}}$ (Assembly operation; $VM_{_{1}}$)	m_6 (Assembly operation; VM_2)	m_6 (Assembly operation; VM_3)	

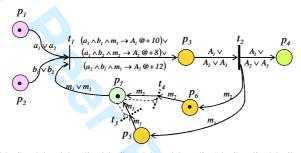
A LIST OF FIGURES



○: Place; ---- : Inhibitor arc; — : Logical transition

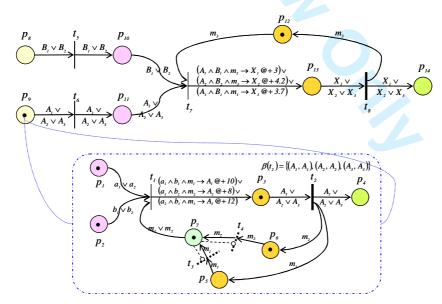
->: Arc; -: Timed transition;: Reconfigurable transition

Figure 1: A basic PN structure and the graphical formalism



$$\begin{split} &\beta(t_1) = \{(A_1,\varepsilon),(A_2,\varepsilon),(A_3,\varepsilon)\}; \ \beta(t_2) = \{(A_1,A_1),(A_2,A_2),(A_3,A_3)\}; \ \beta(t_2) = \{(m_1,\varepsilon)\}; \ \beta(t_2) = \{(m_2,\varepsilon)\}; \\ &E((p_1,t_1),(A_1,\varepsilon)) = a_1; \ E((p_1,t_1),(A_2,\varepsilon)) = a_1; \ E((p_1,t_1),(A_3,\varepsilon)) = a_2; \ E((p_2,t_1),(A_1,\varepsilon)) = b_1 \\ &E((p_2,t_1),(A_2,\varepsilon)) = b_2; \ E((p_2,t_1),(A_3,\varepsilon)) = b_1; \ E(t_2,p_2),(m_1,\varepsilon)) = m_1; \ E(t_4,p_2),(m_2,\varepsilon)) = m_2 \\ &E((t_1,p_3),(A_1,\varepsilon)) = a_1,b_1,m_1 \to A_1 @+10; \ E((t_1,p_3),(A_2,\varepsilon)) = a_1,b_2,m_2 \to A_2 @+8 \\ &E((t_1,p_3),(A_3,\varepsilon)) = a_2,b_1,m_1 \to A_3 @+12; \ E((p_3,t_2),(A_1,A_1)) = A_1; E((p_3,t_2),(A_2,A_2)) = A_2 \\ &E((p_3,t_2),(A_3,A_3)) = A_3; E((t_2,p_4),(A_1,A_1)) = A_1; E((t_2,p_4),(A_2,A_2)) = a_2; E((p_3,t_2),(A_1,A_1)) = m_1; E((t_2,p_3),(A_3,A_2)) = m_2; E((p_3,t_2),(A_3,A_2)) = m_2; E((p_3,t_2),(A_3,A_2)) = m_1; E((p_3,t_2),(A_3,A_2)) = m_2; E((p_3,t_2),(A_3,A_2)) = m_1 \end{split}$$

Figure 2: An MNet



 $\beta(t_s) = \{(B_1, B_1, \varepsilon), (B_2, B_2, \varepsilon)\}; \beta(t_s) = \{(A_1, A_1, \varepsilon), (A_2, A_2, \varepsilon), (A_3, A_3, \varepsilon)\}; \beta(t_\tau) = \{(\varepsilon, X_1, \varepsilon), (\varepsilon, X_2, \varepsilon), (\varepsilon, X_3, \varepsilon)\}; \beta(t_s) = \{(\varepsilon, X_1, X_1), (\varepsilon, X_2, X_2), (\varepsilon, X_3, X_3)\}; \beta(t_s) = \{(\varepsilon, X_1, X_2), (\varepsilon, X_2, X_2), (\varepsilon, X_3, X_3)\}; \beta(t_s) = \{(\varepsilon, X_2, \varepsilon), (\varepsilon, X_3, \varepsilon), (\varepsilon, X_3, \varepsilon), (\varepsilon, X_3, \varepsilon)\}; \beta(t_s) = \{(\varepsilon, X_1, X_2), (\varepsilon, X_2, \xi), (\varepsilon, X_3, \varepsilon)\}; \beta(t_s) = \{(\varepsilon, X_2, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_2, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi), (\varepsilon, X_3, \xi)\}; \beta(t_s) = \{(\varepsilon, X$

Figure 3: An ANet including the nested MNet

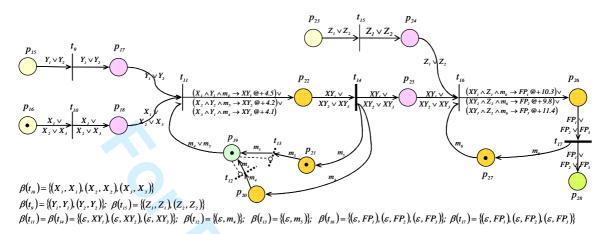


Figure 4: A PNet

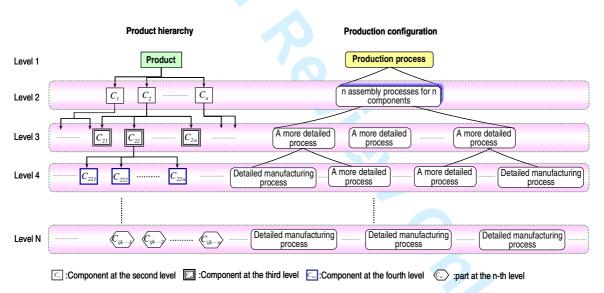


Figure 5: Parallelism between product hierarchy and production configuration

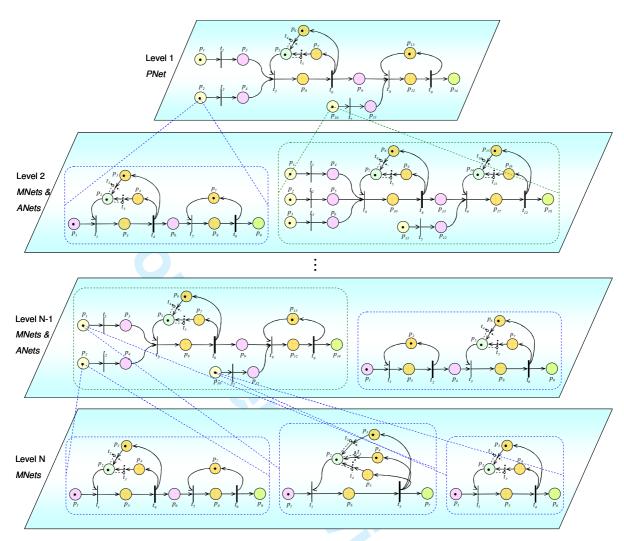


Figure 6: An example of multilevel nested net system

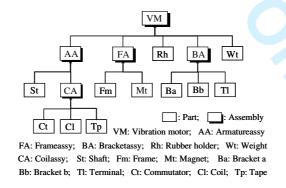


Figure 7: The common product structure of the motor family

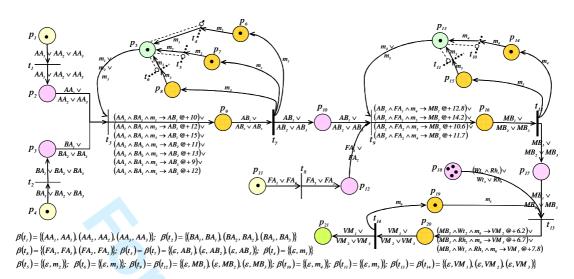
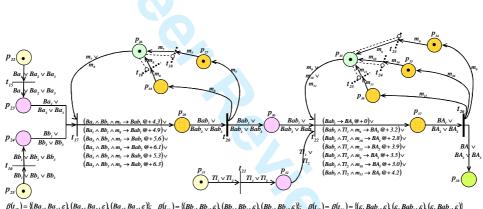
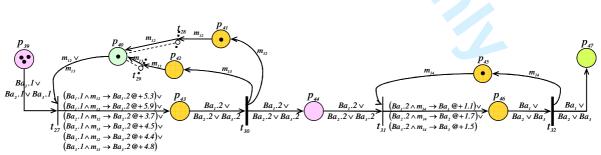


Figure 8: The *PNet* of the motor family



 $\beta(t_{12}) = \{ (Ba_1, Ba_1, \varepsilon), (Ba_2, Ba_2, \varepsilon), (Ba_3, Ba_2, \varepsilon) \}; \quad \beta(t_{16}) = \{ (Bb_1, Bb_1, \varepsilon), (Bb_2, Bb_2, \varepsilon), (Bb_3, Bb_3, \varepsilon) \}; \quad \beta(t_{17}) = \beta(t_{20}) = \{ (\varepsilon, Bab_1, \varepsilon), (\varepsilon, Bab_2, \varepsilon), (\varepsilon, Bab_2, \varepsilon), (\varepsilon, Bab_3, \varepsilon) \}; \quad \beta(t_{17}) = \{ (\varepsilon, m_1, \varepsilon) \}; \quad \beta(t_{17}) = \{ (\varepsilon, Ba_1, \varepsilon), (\varepsilon, Ba_1, \varepsilon), (\varepsilon, Ba_2, \varepsilon), (\varepsilon, Ba_2, \varepsilon) \}; \quad \beta(t_{12}) = \{ (\varepsilon, m_1, \varepsilon) \}; \quad \beta(t_1$

Figure 9: The ANet of the bracketassy family



 $\beta(t_{_{27}}) = \beta(t_{_{30}}) = \{(Ba_{_{1}}.2,\varepsilon), (Ba_{_{2}}.2,\varepsilon), (Ba_{_{3}}.2,\varepsilon)\}; \beta(t_{_{28}}) = \{(m_{_{12}},\varepsilon)\}; \beta(t_{_{29}}) = \{(m_{_{13}},\varepsilon)\}; \beta(t_{_{31}}) = \{(Ba_{_{1}},\varepsilon), (Ba_{_{2}},\varepsilon), (Ba_{_{3}},\varepsilon)\}; \beta(t_{_{32}}) = \{(Ba_{_{1}},Ba_{_{1}}), (Ba_{_{2}},Ba_{_{2}}), (Ba_{_{3}},Ba_{_{3}})\}; \beta(t_{_{32}}) = \{(Ba_{_{1}},S), (Ba_{_{2}},S), (Ba_{_{3}},S), (Ba_{_{3$

Figure 10: The *MNet* of the bracket a family

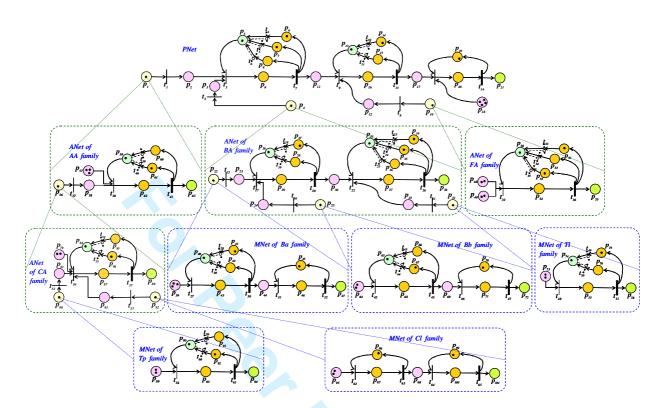


Figure 11: The multilevel nested net system of production configuration

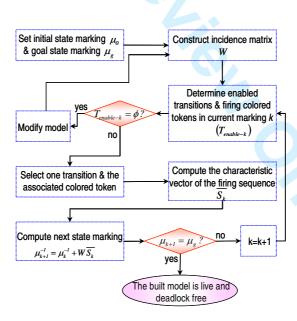


Figure 12: Procedure of deadlock detection algorithm (adapted from Wang and Wu, 1998)

```
\mu_0 = (Ba_1 \cdot 1 + Ba_2 \cdot 1 + Ba_3 \cdot 1, m_{13}, m_{12}, 0, 0, 0, m_{14}, 0, 0)
                                                                                                                \mu_8 = (0, 0, 0, m_{13}, Ba_{\frac{3}{1}} \cdot 2, Ba_1 \cdot 2, 0, Ba_2, 0)
                  t_{27} / Ba_1.2
 \mu_{I} = (Ba_{2} \cdot I + Ba_{3} \cdot I, 0, m_{I_{2}}, 0, Ba_{I} \cdot 2, 0, m_{I_{4}}, 0, 0)
\downarrow t_{28} / m_{I_{2}}
                                                                                                              \mu_{9} = (0, m_{13}, 0, 0, Ba_{3} \cdot 2, Ba_{1} \cdot 2, 0, Ba_{2}, 0)
\downarrow t_{32} / Ba_{2}
\mu_2 = (Ba_2 \cdot I + Ba_3 \cdot I, m_{12}, 0, 0, Ba_1 \cdot 2, 0, m_{14}, 0, 0)
\downarrow t_{27} / Ba_2.2
                                                                                                             \mu_{10} = (0, m_{13}, 0, 0, Ba_3 \cdot 2, Ba_1 \cdot 2, m_{14}, 0, Ba_2)
\downarrow t_{31} / Ba_1
                                                                                                             \mu_{II} = (0, m_{I3}, 0, 0, Ba_3, 2, 0, 0, Ba_1, Ba_2)
\downarrow t_{32} / Ba_1
\mu_3 = (Ba_3 \cdot 1, m_{12}, 0, 0, 0, Ba_2 \cdot 2 + Ba_1 \cdot 2, 0, m_{14}, 0, 0)
\mu_{3} - (Ba_{3} \cdot 1, m_{12}, 0) = 1
\downarrow t_{30} / Ba_{2}.2
\mu_{4} = (Ba_{3} \cdot 1, 0, m_{12}, 0, Ba_{1} \cdot 2, Ba_{2} \cdot 2, m_{14}, 0, 0)
                                                                                                            \mu_{12} = (0, m_{13}, 0, 0, Ba_3 \cdot 2, 0, m_{14}, 0, Ba_1 + Ba_2)
t_{30} / Ba_3 \cdot 2
\mu_5 = (Ba_3 \cdot 1, 0, m_{12}, 0, Ba_1 \cdot 2, 0, 0, Ba_2, 0)
                                                                                                           \mu_{13} = (0, m_{13}, m_{12}, 0, 0, Ba_3, 2, m_{14}, 0, Ba_1 + Ba_2)
\downarrow t_{31} / Ba_3
                                      \int t_{28} / m_{12}
\mu_{6} = (Ba_{3} \cdot I, m_{12}, 0, 0, Ba_{1} \cdot 2, 0, 0, Ba_{2}, 0)
\downarrow t_{27} / Ba_{2} . I \wedge m_{12}
\downarrow t_{17} / Ba_{2} . I \wedge m_{12}
                                                                                                           \mu_{14} = (0, m_{13}, m_{12}, 0, 0, 0, 0, Ba_3, Ba_1 + Ba_2)
\downarrow t_{32} / Ba_3
 \mu_7 = (0, 0, 0, 0, Ba_3 \cdot 2 + Ba_1 \cdot 2, 0, 0, Ba_2, 0)
                                                                                                              \mu_{15} = (0, m_{13}, m_{12}, 0, 0, 0, m_{14}, 0, Ba_3 + Ba_1 + Ba_2)
```

Figure 13: A feasible firing sequence leading to the goal state