

# A Methodology Integrating Petri Nets and Knowledge-based Systems to Support Process Family Planning

Linda Zhang, Qianli Xu, Petri Helo

## ► To cite this version:

Linda Zhang, Qianli Xu, Petri Helo. A Methodology Integrating Petri Nets and Knowledge-based Systems to Support Process Family Planning. International Journal of Production Research, 2011, pp.1. 10.1080/00207543.2011.555787 . hal-00732518

# HAL Id: hal-00732518 https://hal.science/hal-00732518

Submitted on 15 Sep 2012

**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.



## A Methodology Integrating Petri Nets and Knowledge-based Systems to Support Process Family Planning

Journal:	International Journal of Production Research					
Manuscript ID:	TPRS-2010-IJPR-1157					
Manuscript Type:	Original Manuscript					
Date Submitted by the Author:	21-Nov-2010					
Complete List of Authors:	Zhang, Linda; University of Groningen, Operations Xu, Qianli; Nanyang Technological University Helo, Petri; University of Vaasa					
Keywords:	PETRI NETS, INTEGRATION, PROCESS PLANNING					
Keywords (user):	Product family, Process family					



http://mc.manuscriptcentral.com/tprs Email: ijpr@lboro.ac.uk

# A Methodology Integrating Petri Nets and Knowledge-based Systems to Support Process Family Planning

Linda L. Zhang<sup>\*1</sup>, Qianli Xu<sup>2</sup>, Petri Helo<sup>3</sup> <sup>1</sup>Department of Operations, University of Groningen, Groningen, The Netherlands <sup>2</sup>Department of Computer Graphics & Interfaces, Institute for Infocomm Research, A\*STAR, Singapore <sup>3</sup>Logistics Systems Research Group, University of Vaasa, Vaasa, Finland

#### Abstract

Planning production processes for product families has been well recognized as an effective means of achieving successful product family development. However, most existing approaches do not lend themselves to planning production processes with focus on the optimality of the cohort of a product family. This paper addresses process family planning for product families. In view of the advantages of Petri nets (PNs) for modeling large systems, the potential of knowledge-based systems (KBSs) for solving complex problems and the analogy in between, we develop a methodology by integrating PNs and KBSs to support process family planning. An integrated product-process family structure, called IP<sup>2</sup>S, is proposed to organize all data pertaining to a product family and the corresponding process family planning is further developed by integrating the principles of several well-defined PN extensions. Thus, this paper also contributes to visualizing the dynamic behaviors and reasoning behind process family planning. The methodology is applied to process family planning for a truck family. The preliminary results demonstrate the feasibility and potential of using the methodology to support process family planning.

Keywords: Product family, process family, production processes, Petri net, knowledge-based system

<sup>&</sup>lt;sup>6</sup> Corresponding author. E-mail: <u>L.Zhang@rug.nl</u>

#### **1. INTRODUCTION**

A production process describes routings, operations and manufacturing resources (e.g., machines, tools, fixtures, jigs) that are adopted to materialize a design. In accordance with the items located at different levels of a product's BOM (bill of materials) hierarchy, it includes a number of sequenced subprocesses, each of which consists of multiple ordered manufacturing and/or assembly operations. Most existing approaches to planning processes, be they manufacturing processes for parts or assembly processes for assemblies, are of trial and error based on individual planner's experience and intuition due to the lack of well-structured mechanisms (Huang et al., 2004; Tong et al., 2004; Zha, 2004). As a result, it is not uncommon that given the same product, planners develop different production processes. This inevitably causes unnecessary production changeovers, such as variations in routings, operations and manufacturing resources, on shop floors. Production performance in terms of cost, lead-time and quality deteriorates with the inclusion of avoidable production changeovers (Zhang, 2007; Zhang and Rodrigues, 2009). Furthermore, these approaches address the optimality of individual products without considering the impact from/on the production of other products, which are to be produced using the same manufacturing resources existing on shop floors (Gupta and Krishnan, 1998).

To survive, manufacturing companies nowadays strive to develop product families, each of which consists of a number of customized products. Successful product family development relies on the efficiency in both designing and producing product families (Kuttner and Karjust, 2006). On one hand, most of the solution methods reported in the literature can assist companies to quickly design these customized products (or product family members) at low costs (Forza and Salvador, 2002; Simpson, 2004). On the other hand, the key to achieving efficiency in producing these products is to plan production processes that can maintain production stability by eliminating the unnecessary production changeovers and by reducing the complexities in material handling on shop floors (Kuipers et al., 2004; Zhang, 2007). Given the existing manufacturing capabilities, such production processes can only be planned by considering the optimality of the cohort of a product family rather than individual products (Kuttner and Karjust, 2006; Zhang and Rodrigues, 2009). In this regard, the

#### International Journal of Production Research

traditional approaches to planning processes (discussed above) do not lend themselves to realizing product family production efficiency. Consequently, it raises the importance in developing solution methods to plan production processes for a product family by considering the production optimality of the cohort (i.e., process family planning). In this study, we, thus, tackle process family planning by developing a methodology.

Process family planning is inherently complex due to the large number of individual products together with the resulting diverse component items (Hunynen, 1992; Martinez et al., 2000). The complexity is often exacerbated by 1) the multiple feasible operations and the corresponding machines, tools and fixtures; 2) many routing alternatives; and 3) various precedence constraints among operations (Zha, 2004; Zhang and Jiao, 2008). Moreover, the finite manufacturing resources and the often conflicting production performance metrics make the problem more difficult. Other researchers have also reported the difficulties and complexities in planning production processes in the context of product family development (Hayes et al., 2005; Pisano, 1997).

Petri nets (PNs) are able to shed light on the reasoning behind complex processes by modeling and visualizing system's dynamic behaviors (Peterson, 1981). Compared with the complicated symbols, variables, equations, etc. in mathematical models, the graphical representation of PN models offers an easier intuitive understanding of systems in consideration (Zha and Du, 2001). In addition, the underlying mathematical equations enable formal analysis of PN models (Zha et al., 2002). Another advantage of PNs is that extensions and formalisms can be developed based on conventional PNs to meet the modeling requirements of different problem domains. Some well-recognized extensions include colored PNs (CPNs; Jensen, 1997), timed PNs (TPNs; Ramachandani, 1974) and nested PNs (NPNs; Lomazova, 2000), etc.

As a branch of artificial intelligence (AI), knowledge-based systems (KBSs) refer to computer systems that rely on knowledge and reasoning to perform difficult tasks, which are usually undertaken by domain experts (Mockler and Dologite, 1992; Parsaye and Chignell, 1988). They are able to handle the complexities involved in decision making, thereby supporting companies to quickly make better decisions (Haddock and Hartshorn, 1989; Harhalakis et al., 1994; Hynynen, 1992; Liang, 1988; Rao et al., 2005; Singh and Sekhon, 2002). When the processes in consideration are large and involve

complex decision making, companies often resort to KBSs. A KBS not only helps retain the knowledge scattered in the memories of individual domain experts but assists to find solutions using a mechanism by mimicking the reasoning process of domain experts (Mockler and Dologite, 1992). Recognizing the correspondence between PNs and KBSs, researchers have reported integrated solution methodologies based on PNs and KBSs to support modeling and decision making in design, planning and control in manufacturing (e.g., Zha and Du, 2001; Zha et al., 2002; Zha et al., 1998a; 1998b).

In view of the above advantage of PNs for modeling large systems, the potential of KBSs for solving complex problems and the analogy in between, we develop the methodology to support process family planning by integrating PNs and KBSs. In the methodology, an integrated productprocess family structure, termed as IP<sup>2</sup>S, is proposed to organize all data pertaining to the product and process families. The rationale of such an  $IP^2S$  is consistent with that of the generic product-process structure in our previous work (Jiao et al., 2007; Jiao and Zhang, 2008). Since we focus this study on these industries, where design and manufacturing technologies are relatively stable (e.g., automobile industries, PC industries, bicycle industries), the  $IP^2S$  is assumed fixed for a product family during its lifecycle. Built upon the IP<sup>2</sup>S, a formal PN model of process family planning is subsequently developed. The PN model not only captures the elements (e.g., product and process family data) and their relationships inherent in process family planning but visualizes the dynamic behavior of process family planning, i.e., how the production processes for given product family members are planned. To implement the proposed methodology, we further develop a prototype for a truck family. In light of the fact that elaboration on prototype development does not provide additional contribution, we focus on the definitions and concepts involved in the IP<sup>2</sup>S and the formal PN model. By doing so, we also avoid the risk of developing a lengthy and loosely focused paper.

In the rest of this paper, Section 2 presents the literature relevant to planning processes and the applications of PNs and KBSs in manufacturing industries. Section 3 discusses an overview of the proposed methodology. Sections 4 and 5 present the details of the construction of the  $IP^2S$  and the specification of the PN model, respectively. Section 6 introduces the application case and the preliminary results. Section 7 concludes this paper by outlining avenues for future research.

#### **2. RELATED WORK**

#### 2.1 Planning processes for parts and assemblies

As an important link between design and production, planning production processes has received much attention. In the process planning community, researchers present various methods, methodologies and system prototypes to address process planning for parts. Many studies have been reported with focus on specific industries. For example, Choi et al. (1999), Lin and Chang (1997), Lin and Deng (2001) and Tor et al. (2005) present their methods for planning stamping processes in progressive die design. Tong and Kwong (2000), Tong et al. (2004), Yeung and Yuen (1996) and Yeung et al. (1995) apply AI techniques to facilitate process design for transfer molding of electronic packages in the semiconductor industry. In addition, process planning is approached from a variety of aspects, such as plan evaluation and optimization (Huang et al., 2004; Li et al., 2004), integration of planning and scheduling (Aldakhilallah and Ramesh, 1998), planning in one-of-a-kind production (Tu et al., 2000), generic process planning support independent of specific problem domains (Yuen et al., 2003).

Recognizing the importance of assembly operations in production, researchers in the assembly planning community have made endeavors to assembly planning, aiming to generate better assembly plans. Most published articles focus on assembly modeling, sequencing and evaluation (Yuan and Li, 2000; Zha, 2004). In assembly modeling, PN techniques have been intensively investigated to develop models (Ben-Arieh et al., 2004; Zha, 2004; Zha and Du, 2001; Zhang, 1989). In assembly sequencing and evaluation, the advanced problem-solving methodologies in AI have been adopted to obtain assembly plans (Thomas, 1992; Zha et al., 1998a; Zha et al., 1998b; Zha and Du, 2001; Zha, 2004; Zhang, 1989).

In summary, most existing studies address either assembly planning or process planning for assemblies or parts at a lower level with focus on operations details, as shown in Figure 1. Essentially, they attempt to determine specific process parameters (e.g., cutting speed, collision-free path, feed rate). In addition, they focus on the process optimality of individuals, be it a part or an assembly. To ensure the effectiveness of these assembly/process planning activities, the production processes of

final products at a higher level should be optimal in the context of product family production, on which we put our focus in this study, as shown in Figure 1.

#### 2.2 PNs and KBSs

As stated in (Peterson, 1981, Moore and Gupta, 1996), PNs are a very powerful means of modeling, analyzing and designing discrete systems such as assembly systems, production systems and flexible manufacturing systems (FMSs). They also possess the potential to integrate into an AI framework (Tonshoff et al., 1988). In view of the advantage of KBSs in problem solving, many researchers have dedicated themselves to KBS applications in the area of production and manufacturing (Haddock and Hartshorn, 1989; Harhalakis et al., 1994/1995; Hynynen, 1992; Liang, 1988; Rao et al., 2005; Singh and Sekhon, 2002). The benefits of integrating PNs and KBSs for systems design, modeling, scheduling and control have been recognized; and studies addressing different problems relevant to the integration of the two techniques have been reported.

Zha et al. (1998a; 1998b) put forward the use of PNs and KBSs to assist flexible assembly system modeling and automatic generation of assembly plans. In their method, they develop a new knowledge PN formalism for capturing data and information about assembly topology, geometry and constraints. Recognizing the complexities in developing FMS controllers, Zha et al. (2002) adopt the above knowledge PN formalism to first model FMS and subsequently, develop an Internet and web-based system to design and verify FMS controllers. By employing the same knowledge PN model, the group develop KBSs to accommodate electro-mechanical assembly planning (Zha, 2004), mechanical systems and assembly modeling (Zha and Du, 2001) and agent-based collaborative assembly planning (Zha et al., 2003).

Based on the correspondence between PN models and KBSs, Righini (1990) develops FIRST (flexible industrial robotics simulation tool) for simulation of complex distributed manufacturing systems. He also defines modular PNs and discusses how they can assist to develop KBSs for system modeling (Righini, 1993). Based on updated PNs, Harhalakis et al. (1995) design a KBS for controlling and integrating information flows between computer-aided design, computer-aided

 process planning, manufacturing resources planning and shop floor control. In view of the advantages of high level PNs, Muro-Medrano et al. (1998) introduce knowledge representation-oriented PNs (KRONs) for modeling discrete event systems. Based on KRONs, they also report the design of a prototype for scheduling and planning. Similarly, this study attempts to facilitate process family planning by capitalizing on the advantages of integrating PNs and KBSs.

### **3. METHODOLOGY OVERVIEW**

The complexities in process family planning have raised several requirements towards the design of appropriate methodologies. Such methodologies should be able to visualize and automate the generation of a set of production processes in accordance with the set of customized products in a family. In line with this general requirement, more specific requirements are identified as follows:

• A mechanism should be designed to capture existing product and process data and their relationships, thereby anchoring process family planning to a common platform.

• A formal representation should be constructed to model the dynamic behavior of process family planning by handling the involved complexities, thus shedding light on the underlying logic for planning processes families.

• Computer support incorporating product and process family knowledge should be designed to facilitate process family planning. In other words, such support should be able to automatically generate process families for given product families.

To satisfy such requirements, this study proposes a methodology by integrating PNs and KBSs, as shown in Figure 2. In the first step, the IP<sup>2</sup>S is constructed to organize data pertaining to a given product family and data relevant to the corresponding process family as a single structure. In this way, product and process family data and knowledge that are stored previously in different media, such as designer/engineer's memories, design documents, process cards, are captured and maintained in one unified unit, facilitating downstream product development activities as well. Moreover, since process data is mapped to product data, the IP<sup>2</sup>S entails a theoretically sound mechanism. Based on such mechanism, companies are able to plan production processes by considering the existing manufacturing resources and the common and/or similar routings to be adopted on shop floors.

Furthermore, the IP<sup>2</sup>S helps construct the dynamic model of process family planning in the second step and the database in the KBS in the third step by providing the relationships among product and process data.

#### 

The second step aims to construct a representation reflecting the dynamic behavior of process family planning (i.e., how production processes are planned for given product families). Due to the large number of differing products, constructing process family planning model needs to cope with several major issues, as elaborated below.

First, the set of customized products and the resulting various component items require different manufacturing resources and operations. As a result, various data describing products, component items and process elements are involved in process family planning. In this regard, the representation to be constructed should be able to model product and process variety. Second, in spite of the fact that products in a family possess similarities, the production processes of individual products differ from one another in routings, operations precedence, etc. This, in turn, underscores the importance in accommodating process variations in order to build one single model. Third, due to the many component assemblies, subassemblies and parts, it is difficult to plan processes by treating a product as a single unit without decomposing it into several manageable subunits. Consequently, this raises a granularity concern in process family planning. The implication is that processes for product items should be planned at different granularity levels along product hierarchies. Such granularity issue must be addressed for constructing a concise model. Last, in addition to the many restrictions on machine capabilities/capacities and operations precedence, planning production processes for an entire product family needs to satisfy additional constraints, such as the tradeoffs among conflicting production objectives, the similar routings and manufacturing resources to be adopted. In this regard, constrains and how they are satisfied in planning process families need to be captured in the model to be constructed.

By considering the above issues, this study applies PN techniques, more specifically CPNs, NPNs and TPNs, to construct the formal model of process family planning. The system of nested colored

timed PNs model can not only capture the dynamics and reasoning of process family planning but provides readers with an easier intuitive understanding.

The last step attempts to support decision making by automatically generating process families for given product families. (Note, in line with the focus of this study: formulation of the IP<sup>2</sup>S and the corresponding PN model, we generally discuss this last step, which itself is an interesting topic in the filed of decision support systems.) In view of the significance of KBSs in decision making support and the analogy between PNs and KBSs, this step proposes KBS development based on the net system model and the IP<sup>2</sup>S. Similar with most reported KBSs, the major functional modules in the KBS should include a user interface, a knowledge base, a database, an inference engine and a working memory. The user interface is to not only obtain input data from users but output suggested production processes. While the knowledge base is to organize all the knowledge involved in planning production processes for a product family, the database is to keep all data pertaining to the product family and the corresponding process family by following the relationships embedded in the  $IP^2S$ . The inference engine controls the entire executive process of the KBS by applying rules in the knowledge base to input data to derive recommended production processes. Containing facts, such as the initial input data, the working memory changes in the course of planning production processes for given product family members. At last, the outputs of the KBS are a set of production processes corresponding to the set of product family members.

## 4. INTEGRATED PRODUCT-PROCESS FAMILY STRUCTURE

Regardless of the many definitions in the literature, a consensus is that a product family is a set of related products assuming a common product structure and performing a basic function (Fan and Liu, 1999; De Lit, 2000; Simpson, 2004). By following this consensus, this study focuses on such product families that have common product structures. In addition to the common product structure, individual products in a family often possess similar items of same types (i.e., item families). Thus, in line with the common product structure and item families, all data pertaining to a product family can be organized as a generic product structure, as shown in Figure 3(a). Each node in the generic product structure is a generic item in the sense that it represents a class (or family) of similar item variants of

the same type. Each generic item is described by a number of design parameters. For given specific design parameter values, a generic item can be instantiated to a specific item. As with the instantiation of generic items, instantiating the generic product structure with respect to the specifications of a given product produces a unique BOM describing the product.

#### 

A process family in relation to a product family refers to the set of all feasible production processes that can be used to produce individual products in the family. For a given product family, similarities in product structures and items propagate to production processes in the process family, resulting in similarities in routings, operations and manufacturing resources. Thus, a generic process structure can be organized to include all data pertaining to production processes of the corresponding process family, as shown in Figure 3(b). Each node is a generic process in accordance with a generic item in the generic product structure. Each such generic process can be further decomposed into a number of ordered generic operations together with generic manufacturing resources, setups and cycle times. For example, the generic process, AP<sub>4</sub>, corresponds to the generic final product, FP, in Figure 3(a) and, has two generic operations, AO<sub>1</sub> and AO<sub>2</sub>, and the other generic process elements. A process element can be specified by instantiating the corresponding generic process element with respect to given design specifications. A specific process for a given item can be obtained after all specific operations, manufacturing resources, setups and cycle times have been determined through instantiation.

The  $IP^2S$  can be obtained by linking the generic product structure with the generic process structure, as shown in Figure 3(c). The integration follows materials-operations requirement links in general, that is, the output product items of preceding operations become the input material items of succeeding operations. Linking items with other process elements can be accomplished by following their mapping relationships. Such mapping relationships can be identified from the large volumes of existing production data using data/text mining techniques (Jiao et al., 2008).

The IP<sup>2</sup>S provides companies with a well-structured mechanism by organizing product and process family data as a single unified structure. With such a mechanism, production processes for diverse products can be planned by considering similar routings, operations and manufacturing

resources to be adopted on shop floors. In turn, the similar production processes can eliminate the unnecessary changeovers in production, thus contributing to production stability and efficiency.

## 5. NESTED COLORED TIMED PN MODEL OF PROCESS FAMILY PLANNING

To cope with the major issues in process family planning, a formalism of CTPNs (colored timed PNs) is developed based on the principles of CPNs, NPNs and TPNs. In accordance with the generic items in the IP<sup>2</sup>S, a number of CTPNs is specified to model the process families. A multilevel system of nested CTPNs, *SysN*, is further developed to model process family planning.

The elements essential to a PN are places, tokens, transitions, arcs and arc expressions describing pre/post conditions of transition firing. By attaching data (so called colors in CPNs) defining real world objects to tokens, the resulting colored tokens can capture the large variety of product items and process elements in a compact manner. In conjunction with the relationships among product items and process elements, these colored tokens can also model how constraints are satisfied when determining machines, operations and other process elements. Defining nets, which represent processes producing child items, as tokens residing in places of nets, which model processes producing product items, can address the granularity concern in process family planning. A unique reconfiguration mechanism, including inhibitor arcs, reconfigurable transitions and generic machine places, is defined to accommodate process variations while building one single and concise model.

#### 5.1 Net definitions

Definition 1: A CTPN is defined as a 11-tuple:

 $CTPN = (P, T, A, h, \Sigma, \alpha, V, E^T, E, \tau, \mu)$ , where

 $P = P^B \cup P^I \cup P^M \cup P^{GM}$  is a finite nonempty set of places with 4 disjoint subsets:  $P^B$  (buffer places),  $P^I$  (item places),  $P^M$  (machine places) and  $P^{GM}$  (generic machine places);

 $T = T^{L} \cup T^{R} \cup T^{T}$ ,  $P \cap T = \phi$  is a finite nonempty set of transitions with 3 disjoint subsets:  $T^{L}$ 

(logical transitions),  $T^{R}$  (reconfigurable transitions) and  $T^{T}$  (timed transitions);

 $A \subseteq P \times T \cup T \times P$  is a finite nonempty set of normal arcs;

$$h \subseteq P^{GM} \times T^R$$
,  $h \cap A = \phi$  is a finite nonempty set of inhibitor arcs;

 $\Sigma$  is a set of types or color sets;

 $V = \{v_i\}_n \subseteq \Sigma$  is a set of variables;

 $\alpha: P \mapsto \Sigma$  is a color assignment function that maps a place, p, to a color set,  $\alpha(p)$ ;

 $\tau \in \Re^+ \cup 0$  is a set of non-negative real numbers representing time delays;

 $E^{T}$  is a timed arc expression function that maps arcs to timed expressions such that  $Type(E^{T}(t, p)) = \alpha(p) \wedge Type(Var(E^{T}(t, p))) \subseteq \Sigma \wedge \tau, \forall (t, p) \in T^{L} \times P^{I};$ 

*E* is an untimed arc expression function that maps arcs other than  $T^L \times P^I$  to untimed expressions consisting of  $v_i \in V$  such that  $Type(v_i) = \alpha(p), \forall p \notin t \mid t \in T^T$ ; and

 $\mu: P \mapsto \Sigma_{MS}$  is a marking function specifying the distribution of colored tokens in all places  $(\Sigma_{MS}$  is the family of all multisets over  $\Sigma$ ).

A CTPN is defined to model processes producing an item family and involves the families of immediate child items only. In other words, a CTPN deals with parent and child items at two adjacent levels of the generic product structure. Since such an item family can be the final product family, an assembly family or a part family, CTPNs can be assembly processes for final product and assembly families or manufacturing processes for part families.

Tokens are to model various product and process elements based on the host places; and tokens residing in same places belong to same types. More specifically, tokens residing in buffer places are defined to represent items ready to be processed or items that have been produced; tokens in item places model items that are being processed by machines (including tools, fixtures, jigs, etc.); tokens in machines places reflect idle machines; and tokens in generic machine places indicate machines that are selected from multiple alternatives. Since the CTPNs are specified taking into account the IP<sup>2</sup>S of the product family in consideration, the data attached to tokens is consistent with the data organized in the IP<sup>2</sup>S of the product family.

Logical transitions are defined to model the preconditions of operations (e.g., the availability of machines). Timed transitions are used to represent operations which take certain time durations. Reconfigurable transitions are introduced to model the selection of machines. The firing of

#### International Journal of Production Research

reconfigurable transitions and logical transitions is atomic and takes no time delay, whilst the firing of timed transitions incurs time delays which are equal to the cycle times of the operations represented. Inhibitor arcs are introduced to connect generic machine places to reconfigurable transitions. They prevent the firing of an enabled reconfigurable transition when there is a token in the associated generic machine place. In this way, only one machine is selected each time for given material items. Essentially, inhibitor arcs, reconfigurable transitions and generic machine places form the reconfiguration mechanism that accommodates machine selection and process variations.

While inhibitor arcs control machine selection, normal arcs describe process flows involving both items and machines. Variables in arc expressions are used to specify process flows. Each variable is defined to belong to a color type, thus having the corresponding set of color instances as values. Expressions attached to normal arcs are defined to specify pre/post conditions of transition firing. Both timed arc expressions and untimed arc expressions consist of variables, which belong to the color types of associated places. Evaluating timed arc expressions results in colored tokens representing input items, output items to be produced, machines and the corresponding operations to be adopted, cycle times to be incurred as well; evaluating untimed arc expressions specifies the preconditions of firing logical transitions (e.g., the presence of material items) and the post conditions of firing timed transitions (e.g., the machines that are released after completing operations). The marking function describes system states by specifying the distribution of colored tokens in all places; and the initial making  $\mu_0$  shows the initial state in line with the given colored tokens.

Definition 2: A multilevel system of nested CTPNs (SysN) is defined as a 5-tuple:

$$SysN = (SN, SP^B, \beta, A^R, \gamma)$$
, where

 $SN = \{CTPN_i\}_I$  is a set of colored timed PNs;

 $SP^{B} = \{p_{a}\}_{A}, \forall p_{a} \in P^{B} \text{ is a set of buffer places};$ 

 $\beta: SN \mapsto SP^B$  is a place assignment function that maps SN to  $SP^B$  such that  $\forall CTPN_i \neq CTPN_j \in SN, p \in SP^B \cap CTPN_j : \beta(CTPN_i) = p;$   $A^{R} \subseteq T^{L} \times SP^{B}$  is a set of reciprocal arcs connecting  $T^{L}$  to  $SP^{B}$  such that  $\forall CTPN_{i} \neq CTPN_{j} \in SN \ \forall (t, p) \in T^{L} \times SP^{B} : t, p \in CTPN_{i} \land \beta (CTPN_{j}) = p$ ; and

 $\gamma: T^T \mapsto T^L$  is an assignment function that maps a timed transition in a nested net to a logical transition in a host net such that  $\forall CTPN_i \neq CTPN_j \in SN \ \forall \gamma(t_i) = t_j$  $\forall t_i \in T^T \cap CTPN_i, t_j \in T^L \cap CTPN_j, \ \beta(CTPN_i) = p: p \in CTPN_j \land p \in t_j.$ 

An *SysN* is to model the complete production processes of a given product family. Each CTPN in the system represents either the processes of the final product family, an assembly family, or a part family. By following the generic product structure, CTPNs of child item families are defined as tokens nesting in buffer places of CTPNs of immediate parent item families. The place assignment function links the nested nets to the host buffer places. The set of reciprocal arcs are introduced to connect logical transitions to buffer places which host nets of immediate child items. Moreover, timed transitions representing the last operations in nested nets are linked with logical transitions, which are output transitions of the host buffer places. Defining reciprocal arcs and associating timed transitions with relevant logical transitions aim to accommodate system evolution (see Section 5.3).

## 5.2 Graphical representation of CTPNs and the net system

As an illustrative example, for a given product family A, the generic product and process structures and the  $IP^2S$  have been constructed, as shown in Figures 4(a), (b) and (c), respectively. In the figure, a, c and e are three families of purchased parts and d is a part family to be manufactured in house. Assembly family b is formed by d and e. Based on the generic structures and net definitions, CTPNs of three item families – A, b and d – have been constructed, as shown in Figures 5 (a), (b) and (c), respectively. Since each CTPN is independently defined with a unique ID, the IDs of places/transitions and variables can be same in different CTPNs.

The CTPNs are constructed by following the generic processes of the corresponding items shown in Figure 4(b). The arc expressions are formulated based on the mapping relationships among product

and process data embedded in the  $IP^2S$  in Figure 4(c). The color types of each place are underlined and put adjacent to the corresponding places. The declarations of variables and color types are provided in the boxes with dashed lines in the figure. See the formal description of the three CTPNs in Appendix A.

Based on the CTPNs for individual item families and the  $IP^2S$ , the net system has been constructed, as shown in Figure 6. For clarity, the arc expressions and the declarations of variables and color types are not provided. See formal description of the net system in Appendix B.

If the generic product structure of a given product family has n levels, and at each level there are items to be produced in house, the SysN to be constructed should have n levels. Each level contains CTPNs modeling the processes producing the corresponding item families.

#### **5.3 System evolution**

For any individual CTPN, its dynamic behavior is controlled by transition enabling and firing rules. While a transition enabling rule specifies the conditions under which transitions are enabled, a transition firing rule determines how system state changes after firing an enabled transition by removing/adding tokens from/to relevant places.

(1) Transition enabling rule:

A transition  $t \in T$  is *enabled* in a marking  $\mu$  iff each of its input places holds a "sufficient" number of tokens as specified by the arc expressions, i.e., iff  $E(p, t) \subseteq \mu(p)$ ,  $\forall p \in t$ .

(2) Transition firing rule:

When a transition  $t \in T$  fires in  $\mu$ , it creates  $\mu'$  by generating colored tokens in the output places as specified by  $E(t, p)/E^{T}(t, p)$ ,  $\forall p \in t^{\bullet}$  and by removing tokens from the input places as specified by E(p, t),  $\forall p \in t$ .

While through enabling and firing transitions, individual CTPNs can change from one state to another, the evolution of the SysN relies on the interaction among CTPNs. More specifically, the interaction between host nets and nested nets promote the evolution of the system. As with NPNs

(Lomazova, 2000), transition synchronization supports such interaction. In this study, in order to enable transition synchronization, timed transitions representing the last operations in the nested nets are linked with output logical transitions of the host buffer places. Thus, upon the firing of the timed transition in a nested net, the generated colored token together with colored tokens in other input places will enable and fire the associated logical transition in the host net. The host net, then, will follow transition enabling and firing rules to evolve. The reciprocal arcs are introduced to ensure that the firing of logical transitions will not remove the nested nets from the buffer places by sending them back (Yan et al., 1997).

## 6. APPLICATION CASE AND RESULTS

XYZ (a disguised name; due to the company's concern, the real name is not given) is a truck manufacturer that develops a wide range of truck products. We apply the methodology to a specific truck family. Figure 7 shows the generic product structure of the truck family. In general, the truck family consists of three major assemblies, including the cabinet, the chassis and the wheel set assemblies. Each of these immediate child assemblies has its own child components, be they parts or assemblies.

## 

In accordance with the company's practice, the data associated with the set of customized trucks and their production processes are analyzed a priori. Based on the analysis result, the IP<sup>2</sup>S of the truck family is constructed, as shown in Figure 8. Each product item has a quantity per ( $Q_{xy}$ ), representing the unit number required by the parent item at the immediate higher level. The material items are the inputs to the assembly processes (denoted by circles). Each such process involves one or more than one assembly operations. For example, the assembly process APcs forming the chassis has one assembly operation; the final assembly process APta involves two assembly operations.

## 

Based on the analysis, the CTPN model of the truck family is constructed. The CTPN model has multiple levels in accordance with the truck family's product hierarchy and captures trucks' production processes at different granularity levels. The net in the top level models the final assembly

process (i.e., the APta in Figure 9) to produce trucks. It has two timed transitions,  $t_2$  and  $t_6$ , representing the two assembly operations involved. The material items of the first assembly operation represented by transition  $t_2$  are variants of cabinet and chassis. These variants are represented by colored tokens residing in buffer places  $p_1$  and  $p_2$ , respectively. The output item is the Work-In-Process (WIP) sub-assembly of cabinet-chassis (*ccassy*) represented by colored tokens in place  $p_7$ . The variants of *ccassy* and these of the wheel set represented by colored tokens residing in place  $p_8$  are the material items of the second assembly operation represented by transition  $t_6$ . Two alternative sets of manufacturing resources including machines, tools, fixtures and jigs are able to perform the first assembly operation; and they are modeled by the two machine places  $p_4$  and  $p_5$ . Similarly, places  $p_{10}$  and  $p_{11}$  are used to represent two alternative sets of resources carrying out the second assembly operation.

## 

The colored tokens residing in places  $p_1$ ,  $p_2$  and  $p_8$  are themselves CTPNs at the second level, as shown in the figure. These CTPNs represent the three assembly processes – APca, APcs and APfw in Figure 9 – to produce the variants of cabinet, chassis, and wheel set, respectively. The working principle of the nested CTPNs and that at the following lower levels is similar to that of the CTPN at the top level.

While the CTPN model acts as a knowledge entity for production planning, decision support is realized only when the CTPN is transformed into a series of production rules. The rule generation module is responsible for such a transformation. In the system, rule generation is realized in a semi-automatic way. Initially, the CTPN model is analyzed and translated into production rules using the rule generation engine. For the Petri net components that are very complex and contain conflicts, the designer can produce the rules manually. Figure 10 shows an interface for viewing and editing the rules associated with the CTPN.

## 

For a series of trucks that belong to the family, given their specifications indicated in the corresponding customer orders, their production processes are generated by a simulation process based on the same CTPN model. Along the production processes, the production schedules and the

status of manufacturing resources are generated as well, as shown in Figure 11. The operation order pane lists the sequence of individual operations, which are indexed by their work orders. The list also includes the product family in consideration, the items and their quantity per, the machines used, the starting and completion time of the operations, etc.

## 7. CONCLUSIONS

Given the finite manufacturing resources, well-planned production processes to fulfill individual products are important for companies to obtain the efficiency in product family production. To obtain a holistic view of design changes among product family members and the resulting process variations, these production processes should include both manufacturing processes for parts and assembly processes for assemblies. In part due to the complexities involved, planning production processes for product families where both component part and assembly families are considered remains to be an unexplored area of research. This study proposes a methodology integrating PNs and KBSs to support process family planning.

In the methodology, a unified structure, called  $IP^2S$ , is proposed, aiming to provide a wellstructured mechanism for process family planning. The  $IP^2S$  organizes all data pertaining to a given product family and all potential data describing the corresponding process family. Planning production processes for multiple products based on the  $IP^2S$  can thus eliminate the unnecessary changes in routings and manufacturing resources. Based on the  $IP^2S$ , a formalism of CTPNs is developed to build a formal model of process family planning. Thanks to the modeling power of PNs, the constructed model not only visualizes the complex planning processes at different granularity levels but sheds light on the underlying logic of process family planning. In the last step of the methodology, the design of a KBS is generally discussed for the automatic generation of production processes for a given product family. It takes into account both the  $IP^2S$  and the PN model of process family planning. The methodology is applied to a truck family. The application results have demonstrated the methodology's applicability.

While this study concerns the problem of planning process families for given product families and contributes to obtaining production stability, it does not address the integrated design of product and process families. As pointed out in (Jiao et al., 2007), most of the reported integrated product and process models focus on individual products rather than product families. Thus, the integrated design of product and process families may pave a potential avenue for future research. As with product platforms for product family design, a unified product-process platform may be developed to support product family design, production and the integration of design and production. Furthermore, this study is centered around the first step involved in process family planning: the generation of production processes. It does not address the evaluation of alternative production processes and the associated issues, such as computation complexity. In this regard, future research efforts might be put in production process evaluation together with the relevant issues. At last, in view of the fact that this study does not discuss in detail the issues relevant to the design of a KBS, future efforts might be put in designing such a system so that the efficiency of process family planning can be improved.

## REFERENCES

Aldakhilallah, K.A. and Ramesh, R., 1998, An integrated framework for automated process planning: Design and analysis, *International Journal of Production Research*, 36(4), pp. 939-956.

Ben-Arieh, D., Kumar, R.R. and Tiwari, M.K., 2004, Analysis of assembly operations' difficulty using enhanced expert high-level colored fuzzy Petri net model, *Robotics and Computer-Integrated Manufacturing*, 20, pp. 385-403.

Choi, J.C., Kim, B.M. and Kim, C., 1999, An automated progressive process planning and die design and working system for blanking or piercing and bending of a sheet metal product, *International Journal of Advanced Manufacturing Technology*, 15, pp. 485-497.

De Lit, P., 2000, A comprehensive and integrated approach for the design of a product family and its assembly system, PhD Dissertation, CAD/CAM Department, University of Libre de Bruxelles, Brussels, Belgium.

Fan, I.S. and Liu, C.K., 1999, Product family and variants: Definition and models, in *Flexible Automation and Intelligent Manufacturing 1999* (Ashayeri, J., Sullivan, W.G. and Ahmad, M.M., Eds). New York: Begell House, pp. 213-224.

Forza, C. and Salvador, F., 2002, Product configuration and inter-firm co-ordination: An innovative solution from a small manufacturing enterprise, *Computers in Industry*, 49(1), pp. 37–46.

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

Haddock, J. and Hartshorn, T.A., 1989, A decision support system for specific machine selection, *Computers and Industrial Engineering*, 16(2), pp. 277-286.

Harhalakis, G., Lin, C.P., Mark, L. and Muro-Medrano, P., 1994, Implementation of rule-based information systems for integrated manufacturing, *IEEE Transactions on Knowledge and Data Engineering*, 6(6), pp. 892-908.

Harhalakis, G., Lin, C.P., Mark, L. and Muro-Medrano, P., 1995, Structured representation of rule-based specifications in CIM using updated Petri nets, *IEEE Transactions on Systems, Man, and Cybernetics*, 25(1), pp. 130-144.

Hayes, R., Pisano, G., Upton, D. and Wheelwright, S., 2005, *Operations, Strategy, and Technology: Pursuing the Competitive Edge*. John Wiley & Sons, Inc.

Huang, S.H., Liu, Q. and Musa, R., 2004, Tolerance-based process plan evaluation using Monte Carlo simulation, *International Journal of Production Research*, 42(23), pp. 4871-4891.

Hunynen, J., 1992, Using artificial intelligence technologies in production management, *Computers in Industry*, 19, pp. 21-35.

Jensen, K., 1997, Colored Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Vol. 2, New York: Springer.

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(1), pp. 112-129.

Jiao J., Zhang L., Zhang Y. and Pokharel S., 2008, Association rule mining for product and process variety mapping, *International Journal of Computer Integrated Manufacturing*, 21(2), pp. 111-124.

Kuipers, B.S., De Witte, M.C. and van der Zwaan, A.H., 2004, Design or development? Beyond the LP-STS debate; inputs from a Volvo truck case, *International Journal of Operations & Production Management*, 24(8), pp. 840-854.

Kuttner, R. and Karjust, K., 2006, Coordination of complex tasks of engineering product and manufacturing process optimization, *Proceedings of the Estonian Academy of Sciences Engineering*, 12(3-1), pp. 163-175.

Li, W.D., Ong, S.K. and Nee, A.Y.C., 2004, Optimization of process plans using a constraintbased tabu search approach, *International Journal of Production Research*, 42(10), pp. 1955-1985.

Liang, T., 1988, Expert systems as decision aids: Issues and strategies, *Journal of Information Systems*, 2(4), pp. 41-50.

Lin, Z.C. and Chang, Y.C., 1997, Sash sheet metal development and the application of a backpropagation neural network in the calculation of the developed length, *Journal of Materials Processing Technology*, 69(1), pp. 95-105.

#### International Journal of Production Research

Lin, Z.C. and Deng, C.H., 2001, Application of Petri net in the planning of a shearing cut and bending progressive die workstation, *International Journal of Materials and Product Technology*, 16(6-7), pp. 579-591.

Lomazova, I.A., 2000, Nested Petri nets: A formalism for specification and verification of multi-agent distributed systems, *Fundamenta Informaticae*, 43, pp.195-214.

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

Mockler, R.J. and Dologite, D.G., 1992, *Knowledge-based System: An Introduction to Expert Systems*, New York: Macmillan Publishing.

Moore, K.E. and Gupta, S.M., 1996, Petri net models of flexible and automated manufacturing systems: A survey, *International Journal of Production Research*, 34, pp. 161-170.

Muro-Medrano, P.R., Banares, J.A. and Villarroel, J.L., 1998, Knowledge representationoriented nets for discrete event system applications, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 28(2), pp. 183-198.

Parsaye, K. and Chignell, M., 1988, Expert Systems for Experts. John Wiley & Sons, Inc.

Peterson, J.L., 1981, Petri Net Theory and the Modeling of Systems. Prentice-Hall, Englewood Cliffs, N.J.

Pisano, G.P., 1997, *The Development Factory: Unlocking the Potential of Process Innovation*. Harvard Business School Press: Boston, MA.

Ramachandani, C., 1974, Analysis of asynchronous concurrent systems by timed Petri nets. Technical Report MAC TR 120, MIT, Cambridge.

Rao, M.P., Miller, D.M. and Lin, B., 2005, PET: An expert system for productivity analysis, *Expert Systems with Applications*, 29, pp. 300-309.

Righini, G., 1990, FIRST: A Petri net-based system for simulation of complex distributed manufacturing systems, *Computer Integrated Manufacturing Systems*, 3(4), pp. 252-263.

Righini, G., 1993, Modular Petri nets for simulation of flexible production systems, *International Journal of Production Research*, 31(10), pp. 2463-2477.

Simpson, T.W., 2004, Product platform design and customization: Status and promise, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18(1), pp. 3–20.

Singh, R. and Sekhon, G.S., 2002, An intelligent system for optimal selection of dies and tools for sheet metal operations, *Proceedings of Mechanical Engineers*, Vol. 216, Part B: *Journal of Engineering Manufacture*, pp. 821-828.

Thomas, J.P., 1992, A Petri net framework for representing mechanical assembly sequences, *Proceedings of 1992 IEEE/RJS International Conference on Intelligent Robots and Systems*, Raleigh, NC, pp. 2116-2121.

Tong, K.W. and Kwong, C.K., 2000, Process design of microchip encapsulation: A case based reasoning approach, *Journal of Electronics Manufacturing*, 10, pp. 127-138.

Tong, K.W., Kwong, C.K. and Yu, K.M., 2004, Intelligent process design system for the transfer molding of electronic packages, *International Journal of Production Research*, 42(10), pp. 1911-1931.

Tonshoff, H.K., Beckendorff, U. and Schaele, M., 1988, Some approaches to represent the interdependence of process planning and process control, *Proceedings of the 19th CIRP International Seminar on Manufacturing Systems*, State College, PA, pp. 165-170.

Tor, S.B., Britton, G.A. and Zhang, WY., 2005, A knowledge-based blackboard framework for stamping process planning in progressive die design, *International Journal of Advanced Manufacturing Technology*, 26, pp. 774-783.

Tu, Y.L., Chu, X.L. and Yang, W.Y., 2000, Computer-aided process planning in virtual one-ofa-kind production, *Computers in Industry*, 41, pp. 99-110.

Yan, H.S., Wang, N.S., Cui, X.Y. and Zhang, J.G., 1997, Modeling, scheduling and control of flexible manufacturing systems by extended high-level evaluation Petri nets, *IIE Transactions*, 29, pp. 147-158.

Yeung, T.S. and Yuen, M.M.F., 1996, Implementation study of intelligent system for IC transfer molding process, *Proceedings of SPIE – International Society for Optical Engineering*, 2644, pp. 471-476.

Yeung, T.S., Yuen, M.M.F. and Kwok, W.M., 1995, Development of initial machine setting program for IC transfer molding process, *Proceedings of the 3rd International Conference CIM*, 2, pp. 1445-1452.

Yuan, J.T. and Li, L.C., 2000, Integrating assembly and machining planning using graph-based representation models, *International Journal of Production Research*, 38(12), pp. 2619-2641.

Yuen, C.F., Wong, S.Y. and Venuvinod, P.K., 2003, Development of a generic computer-aided process planning support system, *Journal of Materials Processing Technology*, 139, pp. 394-401.

Zha, X.F., 2004, Planning for STEP-based electro-mechanical assemblies: An integrated approach, *International Journal of Computer Integrated Manufacturing*, 17(4), pp. 305-326.

Zha, X.F. and Du, H.J., 2001, Mechanical systems and assemblies modeling using knowledgeintensive Petri nets formalisms, *Artificial Intelligence for Engineering Design, analysis and Manufacturing*, 15, pp. 145-171.

Zha, X.F., Li, L.L. and Zhang, W.J., 2002, A knowledge Petri net model for flexible manufacturing systems and its application for design and verification of FMS controllers, *International Journal of Computer Integrated Manufacturing*, 15(3), pp. 242-264.

Zha, X.F., Lim, S.Y.E. and Fok, S.C., 1998a, Integrated knowledge-based assembly sequence planning, *International Journal of Advanced Manufacturing Technology*, 14, pp. 50-64.

Zha, X.F., Lim, S.Y.E. and Fok, S.C., 1998b, Integrated knowledge-based Petri net intelligent flexible assembly planning, *Journal of Intelligent Manufacturing*, 9, pp. 235-250.

Zha, X.F., Lim, S.Y.E. and Fok, S.C., 2003, A knowledge intensive multi-agent system for cooperative/collaborative assembly modeling and process planning, Transactions of the SDPS, 7(1), pp. 99-122.

Zhang, L., 2007, Process platform-based production configuration for mass customization, Ph.D. Dissertation, Nanyang Technological University, Singapore.

Zhang, L. and Jiao, J., 2008, Modeling production configuration using nested colored objectoriented Petri nets with changeable structures, Journal of Intelligent Manufacturing, In press, 2008.

Zhang, L. and Rodrigues, B., 2009, A tree unification approach to constructing generic processes, IIE Transactions, 41(10), pp. 916-929.

Zhang, W.X., 1989, Representation of assembly and automatic robot planning by Petri net, IEEE Transactions on System, Man, Cybernetics, 29(2), pp. 418-422.

ι, Repr.. stem, Man, Cyber..

## A LIST OF FIGURES

- Figure 1. Process family planning positioning
- Figure 2. A systematic methodology for process family planning
- Figure 3. Integrated product-process family structure
- Figure 4. Generic structures of product family A
- Figure 5. CTPN models of processes of three item families
- Figure 6. The net system of process family planning for product family A
- Figure 7. The generic product hierarchy of the truck family
- Figure 8. The IP<sup>2</sup>S of the truck family
- Figure 9. The CTPN model representing process family planning of the truck family
- Figure 10. Examples of production rules for executing CTPN
- Figure 11. The generated production process based on CTPN simulation



Figure 1. Process family planning positioning



Figure 2. A systematic methodology for process family planning



Figure 3. Integrated product-process family structure



Figure 4. Generic structures of product family A



Figure 5. CTPN models of processes of three item families

Page 27 of 32



Figure 6. The net system of process family planning for product family A



Figure 7. The generic product hierarchy of the truck family



Figure 8. The IP<sup>2</sup>S of the truck family



Figure 9. The CTPN model representing process family planning of the truck family

2	
3	
4	
5	
Â	
7	
1	
8	
9	
10	
11	
12	
12	
13	
14	
15	
16	
17	
18	
10	
19	
20	
21	
22	
23	
24	
25	
20	
20	
27	
28	
29	
30	
31	
22	
32	
33	
34	
35	
36	
37	
20	
38	
39	
40	
41	
42	
43	
11	
44	
45	
46	
47	
48	
49	
50	
50	
51	
52	
53	
54	
55	
56	
50	
5/	
58	
59	
60	

File Edit Dri	awing Run View	Help							
N Component		PN Chart Ru	ile Report						
$\odot$		Level		Chabue	Product	A			
Place	Transition Arc	2		Active -	Truck Cab	Waiting approval			
		1-			THREE COD				
$\square$	8	IF		1	THEN				
Layer	Import	000000000			0.1.1.1	Kara and a second s			
		= Input color	Input color Material A=a1 Mode available			AND			
Properties Cust	tomize	Mode							
Attribute	Value	Measure	minute		Measure	minute	Add rule		
Index	1		1.			1	100000000		
Name	Comp A				la ma	Terre			
Туре	Buffer place	F1 LHS			IYPE	IRHS			
Color	Headlight		enal A=a I	L. LAN	ALLEPT	Output (AB=a1b) OR (AB=a1b0)			
Preset			erial A=a1) AND (M	lachine Mil=milj	AULEPT	Uutput (AL=aIC) UR (AL=aCI)			
Postset	T-AsmCab	3 (Ma	Chine MT=mTJUH (	Machine MZ=m3pj	DENY	Process type PI =concurrent			
SubEntry	none	4 Lea	d time PU2>=100		DENY	process PU2			
Inhibited	N	5 Mat	erial A=a2 UR mate	rial A=a3	SET DEFAULT	Uutput AB=a2/a3			
Duration	22	6 (Ma	erial C=c2) AND (M	laterial E =e I J and (Machine N	14 ALLEPT	[Machine added M=m4-1] AND [Process P1=p1-2]			
Mautokon	10	7 Unite	ria C='min changed	ver'	SET DEFAULT	Process time = min (P1, P2, P3+P4)			
	0	8 (Ma	chine M2:status=bu	sy) AND (Material M3=ready)	ALLEPT	AUTIVATE machine search[]			
Min token U	U	9 (Ma	erial M5>=U1 J AND	(PROCESSING TIME<100)	ALLEPT	Process PU3 or Process PU5			
		10 (Ma	chine M5=6) AND (	Product U.param2=p1)	SET DEFAULT	Processing time P6<25			
		11 (Inte	rior color = blue) AN	ID MODEL <fh< td=""><td>ACCEPT</td><td>Material M9='cloth' AND Color ='blue'</td><td></td></fh<>	ACCEPT	Material M9='cloth' AND Color ='blue'			
		12 (Mo	del <fm-fh) and="" li<="" td=""><td>EAD TIME&lt;100</td><td>DENY</td><td>Seat S9-7= 'inbound'</td><td></td></fm-fh)>	EAD TIME<100	DENY	Seat S9-7= 'inbound'			
		13 (Ma	chine M8= Machine	M7) OR (Processing time M8	SET DEFAULT	Process P9.M=M8)			
		14 Com	fort level='medium'		SET DEFAULT	CAB type C9= 'tinted window=1'			
		15 (Mo	del <fm) and="" interi<="" td=""><td>or color C0='Yellow')</td><td>ACCEPT</td><td>Material S3='leather'</td><td></td></fm)>	or color C0='Yellow')	ACCEPT	Material S3='leather'			

Figure 10. Examples of production rules for executing CTPN

<u>File</u> <u>E</u> dit	Drawing Run View	Hel	р								
N Compon	ient	P	N Chart   F	Rule Report							
⊙ Place	Transition Arc	F	Process F	Plan							
			Operation	Order:	Filter:			•			Compute
Layor			Work ord	er Product fam	Item	Item Qty	Line	Machine	Start time	Complete time	
Properties	Customize		Ata 001	Truck cab	Cabinet	1	ATCLine1	WC_ata_p1	6/9/2008 13:00:20	6/9/2008 18:00	Scheduled Order
Attribute	Value	пE	Ata 002	Truck cab	Chassis	1	ATCLine1	WC_ata_p2	7/9/2008 04:00:10	7/9/2008 06:00	
Index	1		Ata 003	Truck cab	Wheel set	2	ATCLine1	WC_ata_p2	7/9/2008 09:00:20	7/9/2008 20:00	Refresh
Name	Comp A		Aca 001	Truck cab	Body	1	ATCLine1	WC_aca_r1	8/9/2008 13:00:20	8/9/2008 18:00	
Tupe	Buffer place		Aca 002	Truck cab	Office package	1	ATCLine1	WC_aca_r2	8/9/2008 15:00:20	8/9/2008 17:00	Gantt Chart
Type Color	Duner piace		Aca 003	Truck cab	Interior decorati	3	ATCLine1	WC_aca_t1.a	9/9/2008 13:00:20	10/9/2008 13:0	
Descal	meaulight		Aca 004	Truck cab	Resting packag	( 1	ATCLine1	WC_aca_t1.b	9/9/2008 13:00:20	11/9/2008 16:0	
Pestect	T AsmCab		Aca 005	Truck cab	Audio package	1	ATCLine1	WC_aca_t1.a	12/9/2008 12:00:20	12/9/2008 13:0	
Postset	I ASTILIAD		Acs 001	Truck cab	Power train	1	ATCLine1	WC_acs_t1	12/9/2008 21:00:20	14/9/2008 01:0	
SUDENTRY	none		Acs 002	Truck cab	Suspension	1	ATCLine1	WC_acs_em	14/9/2008 15:00:20	15/9/2008 14:0	
Inhibited	N	-	Aap001	Truck cab	Radio	1	ATCLine1	WC_aap_x0	16/9/2008 13:00:20	16/9/2008 19:0 -	
Duration			•							<u>,</u>	
Max token	10		Load capa	acity chart				Capacity cha	art		
Min token	U		here 1		In In				Loa	d Capacity Char	rt
			Line	Machine	Lapacity L	.oad	10	1.2			
			ATCLINET	WC_ata_p1	3/		12	1			
			ATCLINET	WC_ata_pz	30		0				
			ATCLINET	WC_aca_r1	1/0	া	00	36.65			
			ATCL ine1	WC aca r2	20		29	0.6			
			ATCL ine1	WC aca t1 a	10		3	0.4	·		
			ATCI ine1	WC aca t1 b	22		20	0.2			
			ATCLine1	WC aca t1.a	6		5 -	0			
		H	I		1. ŠI.			1	2 3 4	5 6 7	8 9 1

Figure 11. The generated production process based on CTPN simulation

#### **APPENDIX A**

The 3 CTPNs in Figure 5 can be formally described as follows: (1) CTPN A  $P = P^B \cup P^I \cup P^M \cup P^{GM}$  $= \left\{ \left\{ p_{1}, p_{2}, p_{5}, p_{6}, p_{11} \right\}, \left\{ p_{4}, p_{10} \right\}, \left\{ p_{3}, p_{8}, p_{9} \right\}, \left\{ p_{7} \right\} \right\}^{2} \right\}$  $T = T^L \cup T^T \cup T^R$  $= \left\{ \left\{ t_{1}, t_{3} \right\}, \left\{ t_{2}, t_{6} \right\}, \left\{ t_{4}, t_{5} \right\} \right\}^{\frac{1}{2}}$  $A = \begin{cases} (p_{1}, t_{1}), (p_{2}, t_{1}), (p_{3}, t_{1}), (t_{1}, p_{4}), (p_{4}, t_{2}), (t_{2}, p_{3}), (t_{2}, p_{5}), (p_{5}, t_{3}), (p_{6}, t_{3}), \\ (p_{7}, t_{3}), (t_{3}, p_{10}), (p_{10}, t_{6}), (t_{6}, p_{9}), (t_{6}, p_{8}), (p_{8}, t_{4}), (p_{9}, t_{5}), (t_{4}, p_{7}), (t_{5}, p_{7}), \\ (t_{6}, p_{11}) \end{cases};$  $h = \{(p_7, t_4), (p_7, t_5)\};$  $\Sigma = \{A, C, AC, R, B, FA\};$  $V = \{x, y, z, u, v, r_{j}, \\ E = \begin{cases} x & \text{if } arc = (p_{1}, t_{1}) \\ y & \text{if } arc = (p_{2}, t_{1}) \\ m_{1} & \text{if } arc = (p_{3}, t_{1}) \\ \vdots \\ v & \text{if } arc = \begin{cases} (p_{10}, t_{6}) \\ (t_{6}, p_{11}) \end{cases}$  $E^{T} = \begin{cases} case x, y, r \text{ of } \begin{cases} a_{1}, c_{1}, m_{1} \to ac_{1} @+2.7 \\ a_{2}, c_{2}, m_{1} \to ac_{2} @+2.4 \end{cases} \text{ if } arc = (p_{3}, t_{1}) \\ case z, u, r \text{ of } \begin{cases} ac_{1}, b_{1}, m_{2} \to A_{1} @+4.9 \\ ac_{1}, b_{2}, m_{2} \to A_{2} @+5.2 \\ ac_{2}, b_{1}, m_{3} \to A_{3} @+4.7 \end{cases} \text{ if } arc = (t_{3}, p_{10}) \\ ac_{2}, b_{2}, m_{3} \to A_{4} @+5.0 \end{cases}$  $V = \{x, y, z, u, v, r\};$  $\alpha(p) = \begin{cases} \{a_1, a_2\} & \text{if } p = p_1 \\ \{c_1, c_2\} & \text{if } p = p_2 \\ \vdots & & \vdots \\ \{A_1, A_2, A_3, A_4\} & \text{if } p = p_{10} \\ \{A_1, A_2, A_3, A_4\} & \text{if } p = n \end{cases}$  $\tau = \{2.7, 2.4, 4.9, 5.2, 4.7, 5.0\};$  and  $\mu_0 = \left\{0, 0, m_1, 0, 0, 0, 0, m_2, m_3, 0, (A_1, A_2, A_3, A_4)\right\}.$ (2) *CTPN*\_*b*  $P = P^B \cup P^I \cup P^M \cup P^{GM}$  $= \left\{ \left\{ p_1, p_2, p_7 \right\}, \left\{ p_6 \right\}, \left\{ p_4, p_5 \right\}, \left\{ p_3 \right\} \right\};$ 

$$\begin{split} T &= T^{L} \cup T^{T} \cup T^{R} \\ &= \{\{t_{1}\}, \{t_{2}, \{t_{2}, t_{3}\}\}^{T} \\ &= \{\{t_{1}\}, \{t_{4}\}, \{t_{2}, t_{3}\}\}^{T} \\ A &= \{(p, \mu, t_{1}), (p_{2}, t_{1}), (p_{3}, t_{1}), (t_{1}, p_{6}), (t_{2}, p_{3}), (t_{3}, p_{3}), (p_{4}, t_{2}), (p_{5}, t_{3}), (p_{6}, t_{4})\} \\ &+ \{(p, \mu, t_{2}), (p_{2}, t_{3})\}; \\ \Sigma &= \{D, E, B, R\}; \\ V &= \{x, y, z, r\}; \\ E &= \begin{cases} x & \text{if } arc = (p_{1}, t_{1}) \\ y & \text{if } arc = (p_{2}, t_{1}) \\ r & \text{if } arc = (p_{3}, t_{1}); \\ \vdots \\ z & \text{if } arc = (p_{3}, t_{1}); \\ z & \text{if } arc = (p_{3}, t_{1}); \\ z & \text{if } arc = (p_{3}, t_{1}); \\ z & \text{if } arc = (p_{3}, t_{1}); \\ z & \text{if } arc = (p_{3}, t_{2}) \\ d_{2}, e_{2}, m_{2} \rightarrow b_{2} \oplus 2.0 \\ d_{2}, e_{2}, m_{2} \rightarrow b_{2} \oplus 2.3 \\ d_{2}, e_{3}, m_{2} \rightarrow b_{2} \oplus 2.3 \\ \end{bmatrix} \\ &= \begin{cases} \{d_{1}, d_{2}\} & \text{if } p = p_{3} \\ \{d_{1}, d_{2}\} & \text{if } p = p_{3} \\ \{d_{1}, d_{2}\} & \text{if } p = p_{3} \\ \{b_{1}, b_{2}\} & \text{if } p = p_{4}; \\ \{m_{2}\} & \text{if } p = p_{5} \\ \{b_{1}, b_{2}\} & \text{if } p = p_{5} \\ \{b_{1}, b_{2}\} & \text{if } p = p_{5} \\ \{b_{1}, b_{2}\} & \text{if } p = p_{5} \\ \{b_{1}, b_{2}\}, \text{if } p = p_{5} \\ \{b_{1}, b_{2}\}, \text{if } p = p_{5} \\ \{b_{1}, b_{2}\}, (t_{2}, t_{8}), \{t_{8}, t_{8}\}\}^{T} \\ T = T^{-} \cup T^{T} \cup T^{T} \\ T = T^{-} \cup T^{T} \cup T^{T} \\ = \{\{t_{1}, t_{2}, t_{3}\}, \{t_{2}, t_{8}\}, \{t_{4}, t_{3}\}\}^{T} \\ A = \{(p_{1}, t_{1}), (p_{2}, t_{1}), (p_{2}, t_{2}), (p_{2}, t_{2}), (t_{2}, p_{2}), (t_{2}, p_{3}), (p_{3}, t_{3}), (p_{5}, t_{3}), (t_{5}, p_{3}), (t_{5}, p_{5}), (t_{5},$$

$$-2345678911123415678901222222222223333333333344423445678901123455555555560$$

$$E = \begin{cases} x & \text{if } arc = (p_1, t_1) \\ m_1 & \text{if } arc = (p_2, t_1) \\ y & \text{if } arc = (p_3, t_2) \\ \vdots \\ u & \text{if } arc = \begin{cases} (p_{11}, t_8) \\ (t_8, p_{12}) \end{cases}$$

$$E^T = \begin{cases} case \ x, r \ of \ \begin{cases} d_1 \cdot 0, m_1 \to d_1 \cdot 1 @+ 3.7 \\ d_2 \cdot 0, m_1 \to d_2 \cdot 1 @+ 4.0 \end{cases} \quad \text{if } arc = (t_1, p_3) \\ d_1 \cdot 1, m_2 \to d_1 \cdot 2 @+ 1.8 \\ d_1 \cdot 1, m_3 \to d_1 \cdot 2 @+ 1.6 \\ d_2 \cdot 1, m_3 \to d_2 \cdot 2 @+ 2.0 \end{cases} \quad \text{if } arc = (t_3, p_8); \\ case \ z, r \ of \ \begin{cases} d_1 \cdot 2, m_4 \to d_1 @+ 4.0 \\ d_2 \cdot 2, m_4 \to d_2 @+ 3.8 \end{cases} \quad \text{if } arc = (t_7, p_{11}) \end{cases} \\ case \ z, r \ of \ \begin{cases} d_1 \cdot 0, d_2 \cdot 0 \\ d_1 \cdot 1, d_2 \cdot 1 \\ \end{cases} \quad \text{if } p = p_1 \\ \{m_1\} & \text{if } p = p_2 \\ \{d_1 \cdot 1, d_2 \cdot 1 \} & \text{if } p = p_1 \\ \{m_4\} & \text{if } p = p_1 \\ \{m_4\} & \text{if } p = p_1 \\ \{d_1, d_2\} & \text{if } p = \left\{ p_{11} \\ p_{12} \\ \tau = \{3.7, 4.0, 1.8, 1.6, 2.0, 1.7, 4.0, 3.8\}; \text{and} \\ \mu_0 = \{\{d_1 \cdot 0, d_2 \cdot 0\}, m_1, 0, 0, 0, m_2, m_3, 0, 0, m_4, 0, 0\}. \end{cases}$$

## **APPENDIX B**

The net system in Figure 6 can be formally described as follows:  $SysN = (SN, SP^{B}, \beta, A^{R}, \gamma)$   $SN = \{CTPN \_ A, CTPN \_ b, CTPN \_ d\};$   $SP^{B} = \{p_{6} @ CTPN \_ A, p_{1} @ CTPN \_ b\};$   $\beta(CTPN) = \begin{cases} p_{6} @ CTPN \_ A & if CTPN = CTPN \_ b \\ p_{1} @ CTPN \_ b & if CTPN = CTPN \_ d \end{cases};$   $A^{R} = \{(t_{2}, p_{6}) @ CTPN \_ A, (t_{2}, p_{2}) @ CTPN \_ b\}; \text{ and}$ 

$$\gamma(t) = \begin{cases} t_3 @ CTPN \_ A & if \ t = t_4 @ CTPN \_ b \\ t_1 @ CTPN \_ b & if \ t = t_8 @ CTPN \_ d \end{cases}.$$