Graph Theory-based Logic Relationship Programming of Product Development Process

Hong-Zhong Huang and Ying-Kui Gu

Concurrent Engineering 2007 15: 275
DOI: 10.1177/1063293X07083086

CONCURRENT ENGINEERING: Research and Applications

Graph Theory-based Logic Relationship Programming of Product Development Process

Hong-Zhong Huang[1],* and Ying-Kui Gu[2]

[1]School of Mechatronics Engineering, University of Electronic Science and Technology of China
Chengdu, Sichuan, 610054, China
[2]School of Mechanical & Electronical Engineering, Jiangxi University of Science and Technology
Ganzhou, Jiangxi, 341000, China

Abstract: In order to optimize product development process, essential process improvement or process reorganizing activities are needed. Therefore, it is necessary to analyze the relationships among processes when design information is incomplete and imprecise. However, it is very difficult to program and execute the logic relationships among development processes because of the complexity, fuzziness, and dynamic uncertainty of product development processes. In this article, the relationships of product development processes and their properties are analyzed. Based on the analysis results, a novel process net is developed where the processes and the logic constraints are viewed as nodes and verges of the net, respectively. Graph theory is used as a tool to develop the process net. The method of the process net simplification for identifying the core process and its affiliated processes are also presented.

Key Words: product development, process programming, logic relationship, process net, fuzzy logic.

1. Introduction

A process is the basic unit of activities carried out during a product's life cycle [1]. In this sense, the whole development process can be viewed as a set of sub-processes whose physical meanings are varied continuously along with time. These processes are not isolated. There exist complicated relationships among processes, where the logic relationship is one of the important ones [2].

Allen first defined the temporal logic relations of tasks [3]. According to his research, there is a set of primitive and mutually exclusive relations that could be applied over time intervals. The Allen's temporal logic is defined in a context where it is essential to have properties such as the definition of a minimal set of basic relations, the mutual exclusion among these relations and the possibility to make inferences over them. Raposo, Magalhaes, Ricarte, and Fuks made some improvements to Allen's basic relations by adding a couple of new relations and making some variations of those originally proposed [4]. Da Cruz, Raposo, and Magalhaes used seven basic relations of Allen to develop a logic relation graph of tasks and proposed two properties of relation graph [5]. Li, Liu, and Guo defined the concept of process template and classified the logic relations of processes into five categories, i.e., before, meet, start, equal, and finish [6]. Gu, Huang, and Wu analyzed the preconditions for executing these logic relations strictly [7]. Graph theory provides a stronger tool for process modeling and analyzing. Alocilja presented process network theory (PNT) [8]. According to his research results, the generic properties of process networks can provide a practical analytical framework for the systematic analysis, design and management of physical production system, including material flows, technical costs, and so on. Kusiak used graph theory as a tool to develop a dependency network for design variables and analyzed the dependencies between design variables and goals [9]. Da Cruz, Raposo, and Magalhaes presented a methodology to analytically and graphically express the interdependencies among tasks realized in a collaborative environment [5]. Durai Prabhakaran, Babu, and Agrawal took the advantages of the graph theoretic approach to considering all the design aspects together in a single methodology with the help of matrix algebra and permanents [10].

In this article, we present an approach to analyze the logic relations among processes based on graph theory. The structure of the paper is as follows. Section 2 briefly introduces the definitions of process net, relation degree of process, core process, and generalized core process. Section 3 analyzes the logic relations and their

*Author to whom correspondence should be addressed.
E-mail: hzhuang@uestc.edu.cn

properties of product development processes, and a process net is developed using graph theory. Section 4 proposes the method to simplify the process net, which helps identify the core processes and the logic relations between affiliated processes of the core processes. The method can be used to increase the amount of information provided to the designers for programming, improving or reorganizing the development processes. Section 5 gives an illustrative example. Conclusions are drawn in Section 6.

2. Several Definitions

Definition 2.1: Process net
Process net P_{net} is composed of a finite nonempty process variable set $P = \langle P_1, P_2, \ldots, P_n \rangle$ and a logic constraint relationship set $R = \langle R_1, R_2, \ldots, R_m \rangle$. The process net can be formed by connecting the process variables based on constraint relationships, where the process variables are nodes and logic constraint relationships are edges of the net, respectively. $R_i = \langle \psi, \delta, r \rangle$, where ψ represents the quantitative influence of one process on the other process. δ is the degree of influence. r is the logic relationship between any two processes.
Definition 2.2: Relation degree of process
Relation degree of process degree(p_i) expresses the number of processes that has direct logic relationship with p_i. If p_i has direct relationships with n processes, its relation degree degree(p_i) = n. Generally speaking, there is no isolated process in process net. Therefore, the relation degree of process satisfies the condition degree(p_i) > 0.
Definition 2.3: Core process
Given a process net p_{net} with n processes, calculating degree(p_i), $i = 1, 2, \ldots, n$, the process with maximal $\max_{i \in n}$ degree(p_i) is defined as core process of the net and is denoted by p_c. The processes that have direct relationships with p_c and their relation degree is 1 are defined as the affiliated processes of p_c.
Definition 2.4: Generalized core process
The processes in the reduced process net are defined as generalized core processes of the upper process net. Though their relation degrees are not big, the generalized core processes have very important influence on the performance of the whole process net.

Both core processes and generalized core processes have the properties of hierarchy and nonuniqueness.

3. Developing Network of Development Processes

3.1 Logic Relationships between Any Two Development Processes

The set of logic relations adopted in this work is based on the temporal logic proposed by Allen [3]. The relations can be classified into seven different primitive categories and shown in Figure 1.

1. BEFORE. A BEFORE B. It is a serial relation. Process A must be performed before process B. The ending of process A is the precondition of staring process B. After A is completed, it can switch to other processes, and process B may not be carried out immediately. That is to say, the completion of process A is the necessary but not sufficient condition of carrying out process B.
2. MEETS. A MEETS B. It is a sequential relation. After process A is completed, process B must be carried out immediately. The completion of process A is the necessary and sufficient condition of carrying out process B.
3. STARTS. A STARTS B. It denotes that after a synchronous process unit, process A and process B can be started at the same time.
4. FINISHS. A FINISHS B. It denotes that after processes A and B are completed, they can switch to a synchronous unit.
5. EQUALS. A EQUALS B. Process A and process B must occur simultaneously.
6. OVERLAPS. A OVERLAPS B. Process A and process B occur in the same unit partly.
7. DURING. A DURING B. Process B is executed during the execution of process A.

The relationships above could be expressed by $\langle b \rangle$, $\langle m \rangle$, $\langle s \rangle$, $\langle f \rangle$, $\langle e \rangle$, $\langle o \rangle$, and $\langle d \rangle$, respectively.

3.2 Properties of Logic Relationship

Logic relationship is a special constraint. It has the general properties of constraint, such as relativity and inheritance. Except its general properties of

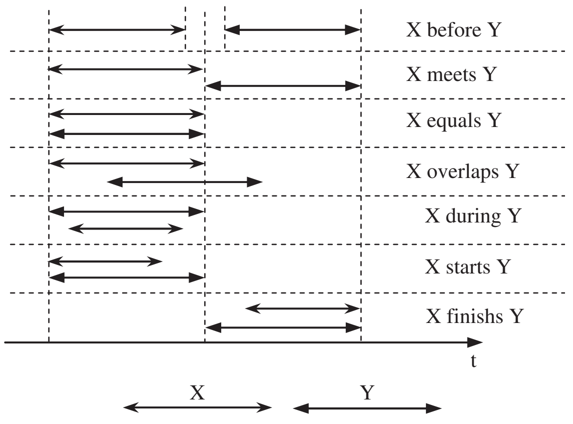

Figure 1. The logic relationships between any two processes X and Y.

constraint, logic relationship has the following special properties:

1. **Transitivity**. Given three different processes p_i, p_j and p_k, if there exist logic relations $p_i\langle r_m\rangle p_j$ and $p_j\langle r_m\rangle p_k$, then there also exists $p_i\langle r_m\rangle p_k$.
2. **Containment**. Given two different processes p_i and p_j, ${}^{f}p_i^n$ and ${}^{f}p_j^m$ are their affiliated processes, respectively. If there exists logic relation $p_i\langle r\rangle p_j$, then there also exists logic relation ${}^{f}p_i^n\langle r\rangle {}^{f}p_j^m$.
3. **Aeolotropism**. The logic relationships among product development processes have aeolotropism. Given two different processes p_i and p_j, the direction of their logic relationship $p_i \xrightarrow{R} p_j$ is unambiguous. This is decided by the properties of product development process.
4. **Irreversibility**. The irreversibility of product development processes decides that their logic relationships have irreversibility. Generally speaking, given two different processes p_i and p_j, there exists $r\langle p_i, p_j\rangle \neq r\langle p_j, p_i\rangle$. Only with the logic relationship $r = \text{EQUAL}$ can the equation $r\langle p_i, p_j\rangle = r\langle p_j, p_i\rangle$ be true.

3.3 Developing a Process Net

The whole development process consists of many interrelated sub-processes. These sub-processes form a net, and the net is called as a process net. The process net can be expressed by graphical representation. There exist matching relationships between elements and nodes of the graph, and between process and edge exist corresponding relationships. $P_{\text{net}} = \langle P, R\rangle$. $P = \langle P_1, P_2, \ldots, P_n\rangle$ is the node set, which represents the nonempty set of processes and their information. $R = \langle R_1, R_2, \ldots, R_m\rangle$ is the edge, which represents the constraint relationships among process nodes.

To develop a process, a net is to express the elements of process using special method. That is to seek the following matching relationships:

1. $P_s \rightarrow P_{\text{net}}$, where P_s is the process set and P_{net} is the process graph.
2. $P_v \rightarrow P$, where P_v is process variables and P is the nodes of the process graph.
3. $P_R \rightarrow R$, where P_R is the logic constraint relationships and R is the edge of process graph.

The logic relationship between two process units is simply binary relations, i.e., it relates only two processes. However, it is possible to overcome this limitation in the whole net. In general, the logic relations among processes are multivariate relationships, i.e., degree$(p_i) \geq 1$. The probability of complete concurrency or partial concurrency of the processes with degree$(p_i) = 1$ is much higher than that of the processes with degree$(p_i) \geq 1$ in logic.

The steps of developing a process net are as follows:

1. Developing the graph of binary logic relations for all processes.
2. Developing the relationship matrix of processes based on the logic relationships among processes. In the matrix, '0' represents that there is no direct logic relationship between the two processes, and '1' represents that there is direct logic relationship between the two processes.
3. On the basis of the relationship matrix, the process net can be developed through connecting all the edges in turn, where the process variables are viewed as nodes, and the logic constraints among variables are viewed as verges.

To process set $P = \langle p_i\rangle, i = 1, 2, \ldots, 12$, there exist logic relationships $m\{p_2, p_1\}$, $b\{p_2, p_4\}$, $m\{p_4, p_3\}$, $s\{p_4, p_5\}$, $s\{p_7, p_5\}$, $d\{p_6, p_7\}$, $e\{p_7, p_8\}$, $o\{p_7, p_9\}$, $f\{p_{11}, p_7\}$, $f\{p_{10}, p_{11}\}$, $b\{p_{11}, p_{12}\}$, and $m\{p_1, p_4\}$. The adjacent matrix M of process relationships and the process net can be developed as follows:

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

The process net is described in Figure 2.

4. Simplifying the Process Net

The main purpose of process net simplification is to identify the core processes, their affiliated processes and their dependencies in order to create the foundation for process improvement. Process simplification also

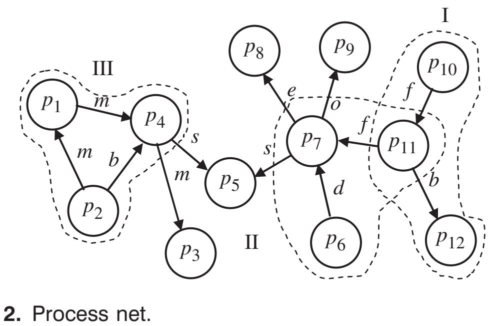

Figure 2. Process net.

benefits the improvements of the concurrency degree of processes and the elimination of float processes to a great extent.

4.1 The Structure Style of Processes

As shown in Figure 2, the process structure can be classified into three categories: serial structure (I), parallel structure (II) and loop structure (III). The serial structure and parallel structure are relatively simple. However, to loop structure, it is impossible to shape the complete closed-loop structure (i.e., there has not entrance and exit of process.) because of the irreversibility of product development process. Therefore, the following two methods may be adopted to deal with the process loop before the process net is simplified:

1. **Equivalent**. As shown in Figure 3(a), the whole loop can be viewed as a compositive process.
2. **Breaking the loop**. As shown in Figure 3(b), the loop can be untied at the entrance. After the loop is untied, the loop structure is transformed into parallel structure.

Both of the methods above have their advantages and disadvantage. The former has its advantages in structure, but the design information could be missed easily. Only the process loop at the most external layer can the equivalent method be adopted. The redundant design information would appear when the second method is adopted, which can ensure the integrality of information, but increase the complexity degree of process simplification.

4.2 Simplification of the Process Net

The simplification of process net adopts an 'out-in' approach. It starts with the most external processes and ends to the most internal ones. The steps for simplifying a process net can be described as follows:

Step 1: Calculating degree(p_i) for all processes, $i=1,2,\dots,n$.

Step 2: Eliminating all the processes with degree(p_i) $=1$. This is because the influence of the process with degree(p_i) $=1$ is local and it only has influence on the single process related to it.

Step 3: Generating a new process net. The reduced process net appears after eliminating all the processes with degree(p_i) $=1$. The nodes in this new process net are labeled as generalized core processes of the upper process net. The logic relationships among these core processes can be identified, and they can decide the logic relationships between their affiliated processes. Suppose p_i and p_j are two core processes, and ${}^{f}p_i^n$ and p_j are their affiliated processes, respectively,

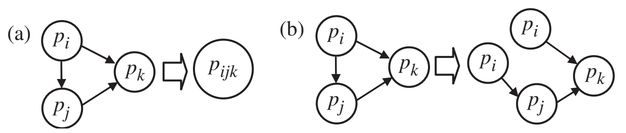

Figure 3. The disposal methods of process loop. (a) Equivalent (b) Breaking.

the relationships among them can be described as follows

$$\text{If } p_i\langle r\rangle p_j, \quad \text{then } {}^{f}p_i^n\langle r\rangle {}^{f}p_j^m$$

Step 4: Repeating the same steps above. The iterations can be executed until the process net has one or two processes. It can be demonstrated that this process is consistent and always ends with one or two processes.

We can get the generalized core processes after each iteration of simplification. The core processes are labeled as the first core process, the second core process, and so on.

The generalized core processes have very important influence on the performance of the whole process net, and their influence degree is much bigger than that of the other processes. When the development process is programmed, the core processes should be considered firstly and should be given higher priority than other processes. When the development process is modified or reorganized, the core processes should also be considered firstly, because the cost of improving the core processes is much smaller than that of improving all the processes under the precondition getting the same benefits. Moreover, it is impossible to improve or reorganize all the nodes in the process net. Only improving the core processes can reduce the complexity degree of process programming to a great extent.

The flow chart of process simplification is shown in Figure 4.

5. Case Study

Take the design process of a worm drive as an example to analyze the logic and dependency relations programming approach based on graph theory. The whole process can be divided into nine sub-processes, i.e., requirement analysis (p_1), concept design (p_2), scenario design (p_3), parameter design (p_4), parameter optimization (p_5), cost analysis (p_6), meshing characteristic analysis (p_7), tooth-surface-profile analysis (p_8), and undercut checking (p_9). The binary logic relations between any two processes can be developed by the analysis of processes, i.e., $f\{p_1,p_2\}$, $m\{p_3,p_1\}$, $o\{p_3,p_6\}$, $o\{p_6,p_1\}$, $f\{p_1,p_4\}$, $b\{p_3,p_4\}$, $b\{p_4,p_5\}$, $e\{p_4,p_7\}$, $e\{p_4,p_8\}$, $e\{p_4,p_9\}$. The adjacent matrix A and the process net can be developed as follows. The process net is described in Figure 5.

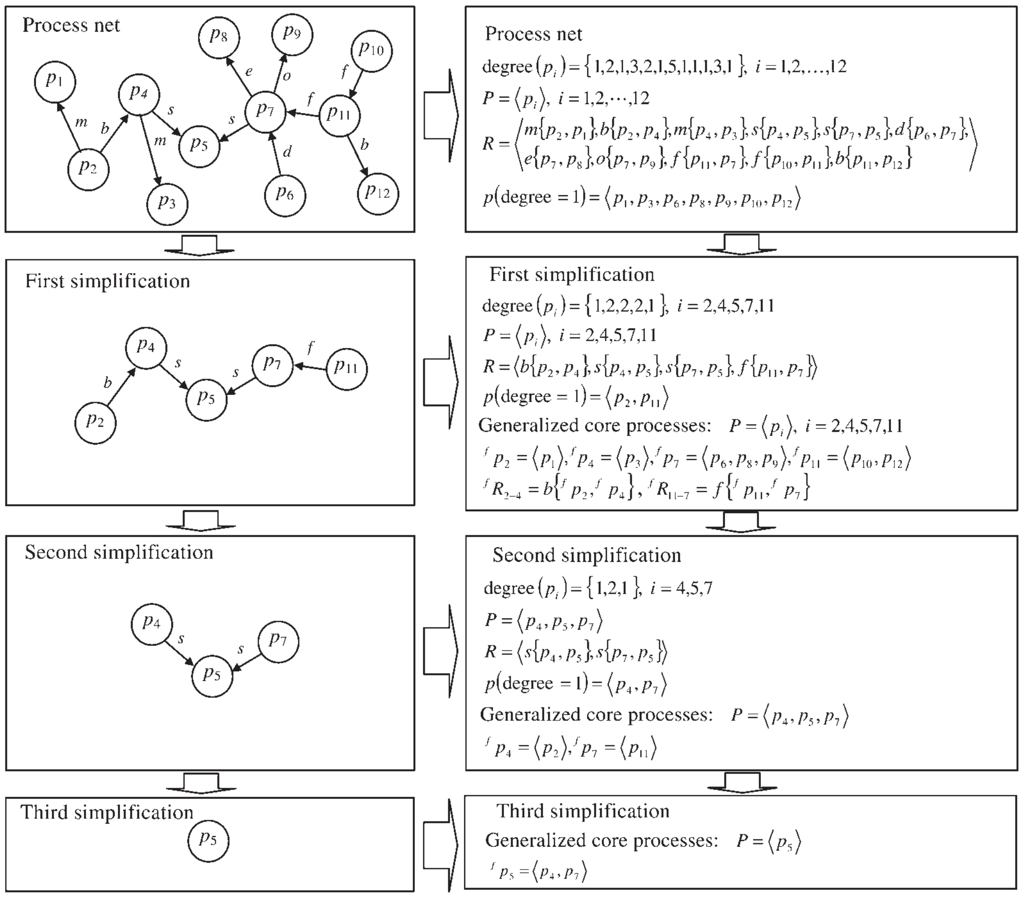
Process net
p_1 p_2 p_3 p_4 p_5 p_6 p_7 p_8 p_9 p_{10} p_{11} p_{12}
m b m s s e o f f d b
Process net
degree$(p_i)=\{1,2,1,3,2,1,5,1,1,1,3,1\}$, $i=1,2,\ldots,12$
$P=\langle p_i\rangle$, $i=1,2,\cdots,12$
$R=\left\langle \begin{matrix} m\{p_2,p_1\}, b\{p_2,p_4\}, m\{p_4,p_3\}, s\{p_4,p_5\}, s\{p_7,p_5\}, d\{p_6,p_7\}, \\ e\{p_7,p_8\}, o\{p_7,p_9\}, f\{p_{11},p_7\}, f\{p_{10},p_{11}\}, b\{p_{11},p_{12}\} \end{matrix} \right\rangle$
$p(\text{degree}=1)=\langle p_1,p_3,p_6,p_8,p_9,p_{10},p_{12}\rangle$
First simplification
p_4 p_2 p_5 p_7 p_{11}
b s s f
First simplification
degree$(p_i)=\{1,2,2,2,1\}$, $i=2,4,5,7,11$
$P=\langle p_i\rangle$, $i=2,4,5,7,11$
$R=\langle b\{p_2,p_4\}, s\{p_4,p_5\}, s\{p_7,p_5\}, f\{p_{11},p_7\}\rangle$
$p(\text{degree}=1)=\langle p_2,p_{11}\rangle$
Generalized core processes: $P=\langle p_i\rangle$, $i=2,4,5,7,11$
${}^f p_2=\langle p_1\rangle, {}^f p_4=\langle p_3\rangle, {}^f p_7=\langle p_6,p_8,p_9\rangle, {}^f p_{11}=\langle p_{10},p_{12}\rangle$
${}^f R_{2-4}=b\{{}^f p_2, {}^f p_4\}$, ${}^f R_{11-7}=f\{{}^f p_{11}, {}^f p_7\}$
Second simplification
p_4 p_5 p_7
s s
Second simplification
degree$(p_i)=\{1,2,1\}$, $i=4,5,7$
$P=\langle p_4,p_5,p_7\rangle$
$R=\langle s\{p_4,p_5\}, s\{p_7,p_5\}\rangle$
$p(\text{degree}=1)=\langle p_4,p_7\rangle$
Generalized core processes: $P=\langle p_4,p_5,p_7\rangle$
${}^f p_4=\langle p_2\rangle, {}^f p_7=\langle p_{11}\rangle$
Third simplification
p_5
Third simplification
Generalized core processes: $P=\langle p_5\rangle$
${}^f p_5=\langle p_4,p_7\rangle$

Figure 4. The flow chart of process simplification.

From Figure 5 we see that the processes p_1 and p_4 have higher relation degree than other processes. They are the core processes of the net and have important influence on the whole design process of the worm drive. Modification of these two processes can guarantee the success of design and reduce the cost of process improvement.

$$A=\begin{bmatrix}1&1&1&1&0&1&0&0&0\\1&1&0&0&0&0&0&0&0\\1&0&1&1&0&1&0&0&0\\1&0&1&1&1&0&1&1&1\\0&0&0&1&1&0&0&0&0\\1&0&1&0&0&1&0&0&0\\0&0&0&1&0&0&1&0&0\\0&0&0&1&0&0&0&1&0\\0&0&0&1&0&0&0&0&1\end{bmatrix}$$

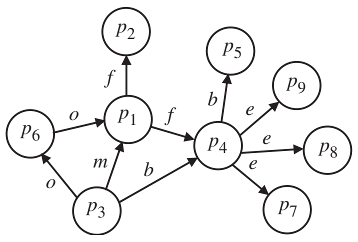

Figure 5. The process net of the worm drive design.

6. Conclusions

The characteristics of product development processes, such as evolvement, dependency and irreversibility, determine that there exist stronger logic relations among processes. In order to achieve the goal of product development, such as developing the product fast and optimally, the graph theory is used as a tool to develop a process net. The proposed method can provide a theoretic foundation for process programming and improving, and can also provide a method for realizing automation of product development at the same time.

Acknowledgments

This research is partially supported by the Specialized Research Fund for the Doctoral Program of Higher Education of China under the contract number 20060614016, and the National Excellent Doctoral Dissertation Special Foundation of China under the contract number 200232.

References

1. Yu, J. (2002). *China Mechanical Design Canon:* Vol. 1, Modern Method of Mechanical Design, *Nanchang: Nanchang Press of Science and Technology*.
2. Gu, Y.K., Huang, H.Z. and Wu, W.D. (2005). The Fuzzy-Logic-Based Reasoning Mechanism for Product Development Process, *Lecture Notes in Artificial Intelligence*, **3614**: 897–906.
3. Allen, J.F. (1984). Towards a General Theory of Action and Time, *Artificial Intelligence*, **23**(2): 123–154.
4. Raposo, A.B., Magalhaes, L.P., Ricarte, I.L.M. and Fuks, H. (2001). Coordination of Collaborative Activities: A Framework for the Definition of Tasks Interdependencies, In: *Proceedings of Seventh International Workshop on Groupware*, pp. 170–179.
5. Da Cruz, A.J.A., Raposo, A.B. and Magalhaes, L.P. (2002). Coordination in Collaborative Environments-A Global Approach, In: *Proceedings of the 7th International Conference on Computer Supported Cooperative Work in Design*, pp. 25–30.
6. Li, S.P., Liu, N.R. and Guo, M. (2002). *Data Standard of Product and PDM*, Beijing: Press of Tsinghua University.
7. Gu, Y.K., Huang, H.Z. and Wu, W.D. (2003). Product Development Microcosmic Process Model Based on Constraint Nets, *Journal of Tsinghua University*, **43**(11): 1448–1451.
8. Alocilja, E.C. (1990). Process Network Theory. *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pp. 66–71.
9. Kusiak, A. (1995). Dependency Analysis in Constraint Negotiation. *IEEE Transaction on System, Man, and Cybernetics*, **25**(9): 1301–1313.
10. Durai Prabhakaran, R.T., Babu, B.J.C. and Agrawal V.P. (2006). Design for 'X'-Abilities of RTM Products – A Graph Theoretic Approach, *Concurrent Engineering: Research and Applications*, **14**(2): 151–161.

Dr Hong-Zhong Huang

Dr Hong-Zhong Huang is a full professor and dean of the School of Mechanical, Electronic, and Industrial Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China. He has held visiting appointments at several universities in the USA, Canada, and Asia. He received his Ph D degree in Reliability Engineering from Shanghai Jiaotong University, China. He has published 120 journal papers and five books in fields of reliability engineering, optimization design, fuzzy sets theory, and product development. He is an Editorial Board Member of International Journal of Reliability, Quality and Safety Engineering, International Journal of Reliability and Applications, International Journal of Performability Engineering, Advances in Fuzzy Sets and Systems, and The Open Mechanical Engineering Journal. He received the Golomski Award from the Institute of Industrial Engineers in 2006. His current research interests include system reliability analysis, warranty, maintenance planning and optimization, computational intelligence in product design.

Dr Ying-Kui Gu

Dr Ying-Kui Gu is currently an associate professor in School of Mechanical & Electronical Engineering at Jiangxi University of Science and Technology, Ganzhou, Jiangxi, 341000, China. He received his Ph D degree in Mechanical Engineering from Dalian University of Technology, China. Dr Gu has published 20 journal papers. His research interests include product development, design optimization, production and engineering management.