J Syst Sci Syst Eng (Jun 2010) 19(2): 131-149
DOI: 10.1007/s11518-010-5132-z
ISSN: 1004-3756 (Paper) 1861-9576 (Online)
CN11-2983/N

APPLYING PROCESS MINING APPROACH TO SUPPORT THE VERIFICATION OF A MULTI-AGENT SYSTEM*

C. OU-YANG[1] **Yeh-Chun JUAN[2]**

[1]*Depart. of Industrial Management, National Taiwan University of Science and Technology, Taipei, Taiwan, China*
ouyang@mail.ntust.edu.tw (✉)

[2]*Depart. of Industrial Engineering and Management, Ming Chi University of Technology, Taipei, Taiwan, China*
ycjuan@mail.mcut.edu.tw

Abstract

Using agent development tools to construct an agent-based system is a well applied approach. However, the development tools usually do not have the function to check the feasibility about the workflow of the agent system during it implementation stage. Therefore, to develop an evaluation approach to analyze the feasibility of a developing agent system such that the improper workflow of an agent system can be found in the early design stage is a necessary task to reduce the risk of implementation.

In this research, a Petri Net (PN) based three-stage evaluation approach was developed. In the conceptual stage, the pitfall of the current agent system developing process was examined and an improvement analysis process was specified. Then, in the system design stage, an evaluation approach which extracted the process log file from a developing agent system into a PN model in terms of a process mining approach-α algorithm was proposed. This model was simulated in a PN simulation package. The agent system performance was evaluated in terms of analyzing the deadlock phenomena of the PN model. Finally, in the implementation stage, the proposed concept was implemented by using an agent developing tool JADE and a PN simulation tool CPN. An agent-based robotic assembly system was used to examine the possible deadlock of the agent system.

Keywords: Agent-based systems, workflow feasibility, process mining, Petri Nets

1. Introduction

Recently, multi-agent systems have been applied in many fields such as manufacturing control, supply chain collaboration and e-commerce. It can be found that the agents required many business functions in order to interact properly among one another. For instance, in the supply chain collaboration, it might require the agents with planning, forecasting, production and inventory control

*The paper was financially supported by National Science Council, Taiwan, through project No. NSC-95-2221-E-011-015

capabilities to cooperate among each other. It also required frequent communications among these agents to achieve an acceptable performance. Therefore, as the application domain became more complicated, the interactions schema among the agents also would be more sophisticated.

On the other hands, there are many approaches developed to design and develop a multi-agent system. Basically, the system development can be categorized as analyzing stage, and system developing stage. In the system analyzing stage, most of the developed modeling approaches, such as Multiagent Systems Engineering (MaSE) and Agent-based Reflective Processes (WARP), were based on UML. That is, the designer can apply various Unified Modeling Language (UML) constructs such as use case diagram, sequence diagram and class diagram to model the behaviors of an agent system. In the system developing stage, several tools have been developed to assist the construction of an agent system. For instance, ZEUS is a graph based tool, users can generate agents and built rules in terms of several build-in constructs. Another popular agent development tools is Java Agent DEvelopment Framework (JADE). It is a JAVA based environment contained many components can be used to construct an agent system.

Once an agent system was constructed, it is necessary to simulate the system behaviors to assure the system performance. Currently, most of the development tools still lack of the capability of system verification. That is once an agent system was constructed, the related system simulation model such as Petri Net (PN) was modeled according to the "understanding" of the agent system behaviors rather than the directly extracted behavior information. Therefore, it would be difficult to assure the compatibility between the agent model and the simulation model. That is, there is no assurance that the data generated from the simulation model and the data shown in the log file of the agent system are in common. In other words, the behavior performance of the PN might not reflect the behaviors of the respective agent system. Therefore, in this research, an evaluation approach was developed, which extracted the process log file from a developing agent system and then transferred into a PN model in terms of α algorithm. The agent system feasibilities were evaluated in terms of analyzing the deadlock of the PN model.

2. Background

Cooperation among various organization units through internet has been addressed heavily in the recent period. Applications such as collaborative design/production activities in terms of Product Development Management (PDM) tools, or Collaborative Planning Forecasting and Replenishment (CPFR) through a supply chain are the few examples. Among the various collaborative application methodologies, agent based approaches had been focused in many areas. In the shop floor manufacturing, agent systems had been used to support the collaboration among the controllers of various manufacturing facilities. For instance, an agent-based discrete shop floor control system was developed to coordinate and control a range of shop floor facilities such as CNC and conveyor (Schoop 2001). Another work addressed on developing an agent-based

production control framework for elevator industry. The agents in this framework could cooperate with each other to achieve lower production cost (Lu 2005). Also, a multi-agent control framework had been developed to support the concept of networked manufacturing. This framework tended to apply agent to integrate various industrial application tools to improve the production efficiency. The proposed concept also implemented in an area in China (Fan 2005). As to the works about applying agent in collaborative product design, a self-adaptive collaborative design method has been proposed to support the design of aircraft tooling. In this work, an integrated multi-agent and PDM technologies has been developed to support the implementation of this model (Li 2009). Also, an e-Engineering framework has been proposed to support the resource planning in a product development process. This work developed various types agent to integrate with several related tools used in product development to assist the arrangement required product design resource (Kuk 2008). A review about the application of agent-based systems in the intelligent manufacturing can be found from Shen's work (Shen 2006).

Several works also addressed on the agent-based supply chain collaboration. For instance, an artificial neural network approach had been used in an agent-based supply chain network to enable the planning tasks. The experimental results showed that the performance about order fulfillment and resource utilization had been improved significantly (Chiu 2004). Another approach had linked the intelligent agent with a CPFR process for trading partner negotiation. Two kinds of agent models (advanced model and learning model) had been proposed. The experimental results showed that these models achieved advantages in terms of inventory-level, stock out level (Cardid 2005). In addition, an agent based negotiation model was proposed to support the collaboration of individual enterprises during the project configuration stage in a supply chain (Lau 2005). A contract-net protocol based MAS were proposed to assist multiple contractors to carry out their distributed scheduling in the supply chain (Lau 2006). In addition, a multi-behavior agent model was developed to support the collaborative planning in the supply chain of lumber industry (Forget 2008). Because of the highly dynamic situation in this industry, the agent model using different planning strategies to support the decision making various stages of the supply chain. Two review works about the agent based collaborative planning can be found in Zhang's paper and Stadtler's works respectively (Zhang 2007, Stadtler 2005).

In addition to the area of manufacturing industries, e-commerce was another field focused on agent-based collaboration. An on-line bargaining systems was proposed by Lin et al. This multi-agent system including three separate agents could be used in a dynamic price issuing and matching environment (Lin 2001). Another multi-agent system InterMarket was developed to assist the user to find the potential business opportunities in an on-line marketplace. In terms of two types of agents, mobile agent and intelligent agent, this system could find the potential trading opportunities for the intended sellers or buyers and improve the success probabilities of the trading events (Kowalczky 2002).

In addition to the related research, several agent-based analyze methods and implementation tools had been developed. DeLoach had proposed MaSE approach for the tasks of agent system design. This approach modified from the famous Object Modeling Technique (OMT) and UML, such that it could easily decompose the problem domain and describes the coordination and communication activities among multiple-agents (DeLoach 1999). Another analyzing approach Gaia dealing with both macro and micro level aspects of a design and has developed its own modeling constructs and language (Wooldridge 2000). As for the graphic-based implementation tools, ZEUS seems a convenient tool. It is a JAVA based toolkit, which provides a visualized design environment. Also, an agent library is provided such that the designer can easily extend the functions of related agents ((Nwana 1999).

Once an agent-based system has been developed, its performance should be verified. Various methods have been used. For instance, a simulation approach has been used to Most of the research works in system verification were based on simulation. For instance, a simulation approach has been used to verify that the strategies made by a cooperative agent system could reach a convergence value for a manufacturing environment (Deen 2006). Another verification rule schema has been developed for a cooperative traffic agents system to validate the situation such as the worst case behaviors of a traffic system (Damm 2006).

In addition to the works on performance verification, very few researches were addressed on the agent system feasibilities verification. The system feasibilities indicated that an agent system could be carried out smoothly in most of the conditions. The methods for feasibilities verification can be found in few modeling tools such as PN. For instance, the deadlock of a PN can be analyzed by calculating the state transformation matrix of the net. The system bounded condition can verified in terms of the place invariant properties of the net.

As for the agent system feasibility verification, the UML based modeling tools still cannot verify certain feasibility situation such as deadlock or infinite-loop of an agent system. However, these situations could occur and difficult to observe for a complicated multi-agent system during its developing stage. For instance, for a multi-agent system with three agents and each agent contained many rules. If there were three rules: *IF A THEN B; IF B THEN C; IF C THEN A,* in each of the three agents respectively. Then, an infinite loop might occur in the agent system if these rules were triggered one after another. However, it would be very difficult to observe these situations if the system designer only inspected the rules in each agent separately.

The idea of applying process mining in the context of workflow management system was first introduced in Agrawal's work (Agrawal 1998). The main object of process mining was to extract sequential relationship among the process events from the process log and to generate the respective process model. Most of the developed mining approach can generated PN oriented process model. Various mining algorithms have been developed recently, such as α miner (van der Aalst 2004) , heuristic miner (Weijters 2003), α++ miner (Wen 2006), and

genetic miner (van der Aalst 2006). Among those, α miner is the most straightforward approach. Basically, this approach intends to find the sequential relationships among the activities existed in the log, and then to construct the respective PN model according to the discovered activity order. Certain constraints cannot be solved by α miner such as looped process has been re-solved in terms of α++ miner. Heuristic miner tends to find the probability of each branched loop in a process and generic miner can solve the situation when there are noises, such as lost of data, existed in the log. A comparison about these miners based on four indexes: fitness, precision, generalization and structure, can be found from (Rozinat 2008).

In this research, an evaluation approach which extracted the process log files from an agent system and transferred into a PN model in terms of α algorithm would be proposed. This transferred model was imported into a PN simulation package CPN. The agent system feasibilities were evaluated in terms of verifying the deadlock of the PN model.

3. The Proposed Evaluation Framework

A three-stage approach has been proposed to support the feasibility analysis of an agent system (Figure 1). In the conceptual stage, the concept of using the behavior information for verifying an agent system was discussed. Then, in the design stage, an approach of extracting the behavior information from the testing agent system and then transforming into a PN system in terms of α algorithm was proposed. The feasibility of the agent system would be examined based on the simulation results of the respective PN. Finally, in the implementation stage, the proposed approach would be implemented in terms of the agent development tool, JADE and a PN tool Colored Petri Net (CPN). An agent-based robotic assembly cell would be used to examine the proposed concepts.

3.1 The Conceptual Stage

In most of the multi-agent system development processes (Figure 2-(a)), a system analysis would be performed first by using UML related tools. Then, a verification model would be built in terms of certain simulation tool such as PN. The PN model would be simulated and analyzed to verify the system performance. If the verification results could be accepted, an agent system would be built and implement. The main issue of this approach was that most of the PN based simulation models were built according to the examination of the behaviors of the agent model rather than the direct integration of the two models. In other words, it is necessary to build a bridge to link the two models.

This research applied another system development processes (Figure 2-(b)). An agent system would be constructed based on UML analysis. The constructed system would be tested and the triggering sequence of the system would be recorded in a log file. Then, the contents of the log file would be transferred to a PN model through α algorithm. The transferred PN model would be simulated and certain infeasible conditions such as deadlock could be verified through the simulation data.

3.2 The Design Stage

The main objective of this section was to develop an approach to verify the feasibility of an agent system. The basic idea was to extract the interaction information from a multi-agent system and then to transfer the behaviors of the extracted data into a PN. The main issue in this stage was how to transfer the information about the agent interactive logic into a proper PN model. In this work, α algorithm was applied to fulfill this goal.

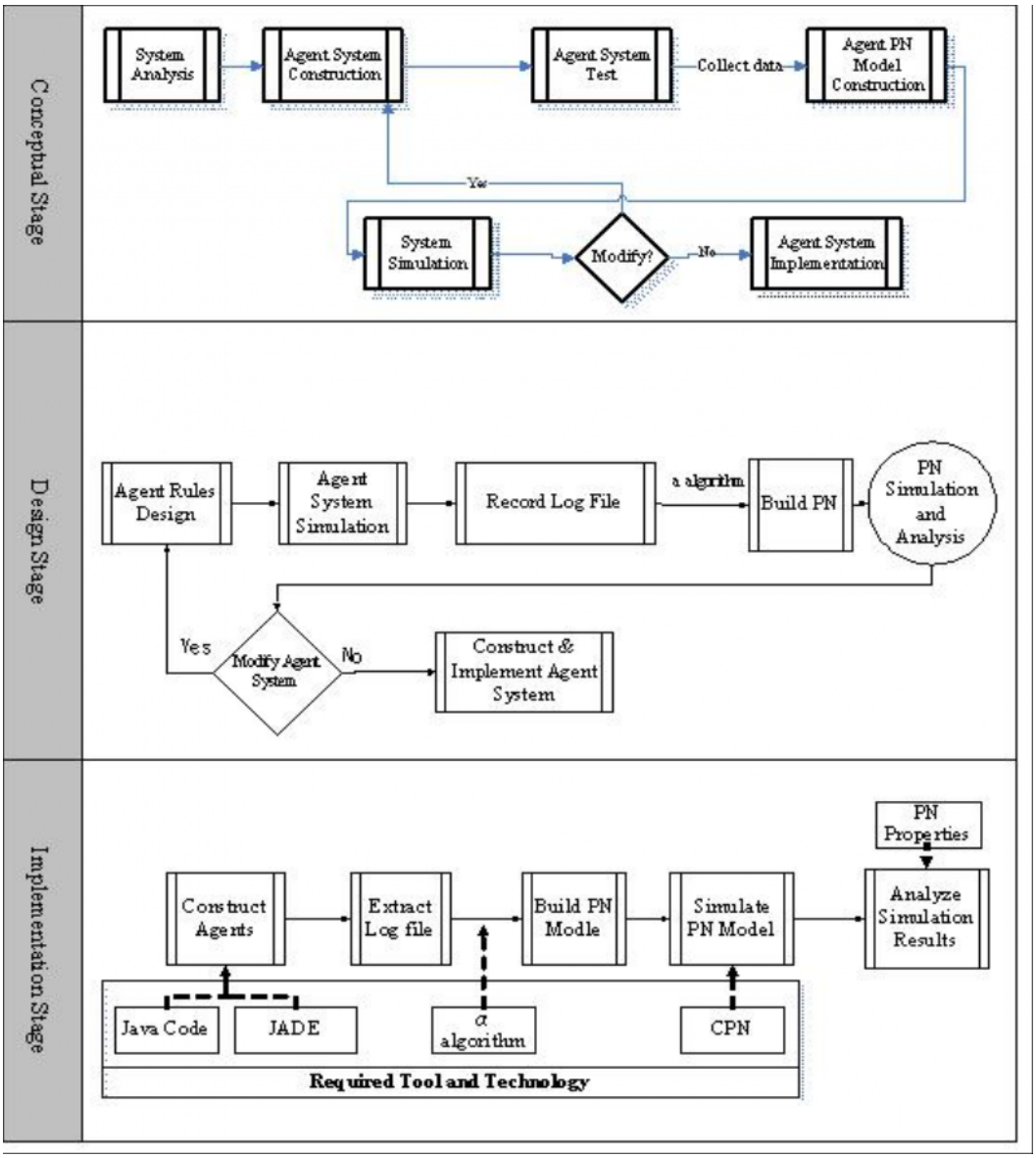
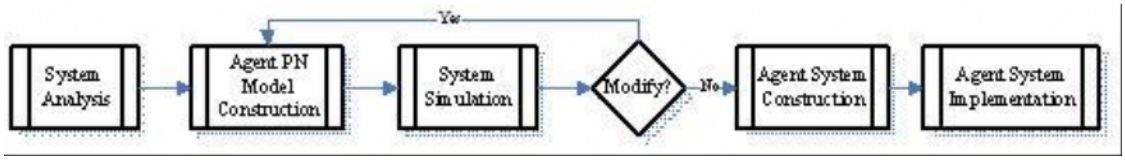
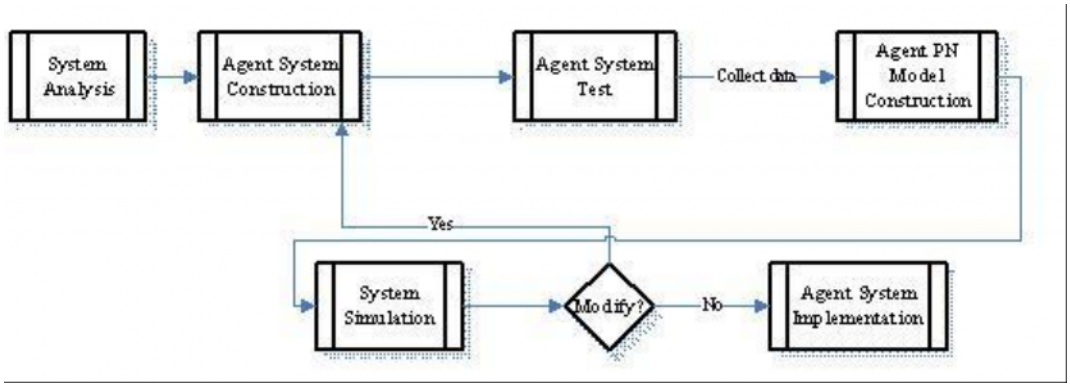


Figure 1 Proposed framework



(a) The original agent development process



(b) The proposed agent development process

Figure 2 Agent development process

Table 1 showed the hypothetical data extracted from an agent system. Basically, each "case" represented a process carried out by the agent system and each "task" indicated a rule fired by a specific agent. Therefore, in this test data, there are five processes carried out by the agent system in a common period of time, and five rules were fired by various agents during that time period.

Based on Table 1 , the recorded data can be represented as: case 1: {a,b,c,d}, case 2: {a,c,b,d}, case 3:{a,b,c,d}, case 4:{a,c,b,d} and case 5:{e,f}. It means that for process "case 1", four rules were fired as the sequence of a, b, c, d.

According to the α algorithm (van der Aalst et al. 2004), the related PN can then be transferred based on the following procedure:

1. Find the transition set: T_W={a,b,c,d,e,f}.
2. Find the set of initial transitions: T_I={a,e}.
3. Find the set of final transitions: T_O={d, f}.
4. Find the set of transitions which has only one directional precedence: X_W={(a,b), (a,c), (b,d), (c,d), (e,f)}.
5. Remove the duplicated transition sets. Y_W={(a,b), (a,c), (b,d), (c,d), (e,f)}.
6. Develop the place set: 1. Add a place construct to each of the transition set; 2. Add the initial and final place constructs. Pw={i_W, $p_{(a, b)}$, $p_{(a, c)}$, $p_{(b, d)}$, $p_{(c, d)}$, $p_{(e, f)}$, o_W}.
7. Complete the final PN: α(w)=(P_W, T_W, F_W).

The transferred PN was shown in Figure 3. It can be found that α algorithm is suitable for transforming one process log file into respective PN model. However, in the multi-agent system, each agent might have its own log file. Therefore, α algorithm should be modified to reflect this situation.

Table 1 The hypothetical data extracted from an agent system

Process	Operation
case 1	task a
case 2	task a
case 3	task a
case 3	task b
case 1	task b
case 1	task c
case 2	task c
case 4	task a
case 2	task b
case 2	task d
case 5	task e
case 4	task c
case 1	task d
case 3	task c
case 3	task d
case 4	task b
case 5	task f
case 4	task d

3.2.1 Multi-Agent System Application and Modified α Algorithm

This research would be addressed on a multi-agent system where a pair of agents would be interacted with one another each time. That is, although there were several agents in a system; however, only two agents would be communicated with one another each time.

According to Aalst et al. (2004) work, if a task was directly followed task b in a log file, then task b is "direct succession" of task a. This situation could occur in a multi-agent system. If one agent sent a request signal to another agent and wait until the reply signal was received, then the tasks of sending and receiving signals would form a "direct succession" relation. However, in a multi-agent system with this situation, this information might be duplicated in the log files of these two agents. For instance, in Figure 4-(a), Agent I would send a request for service information (Task b) to agent II. Then, Agent II would send a reply signal back to Agent I (Task c) (Figure 4-(b)). Task b and Task c formed a direct succession relation. However, these tasks would appear in the log files of both agents. Moreover, from the PN model points of view, since Agent I played an active role at the predecessor task and Agent II played a passive role at the successor task in this directive succession relations, there should be two places which could directly enable the transitions with respective to Task b and Task c respectively. Therefore, α algorithm should be modified to reflect this multi-agent interactive situation.

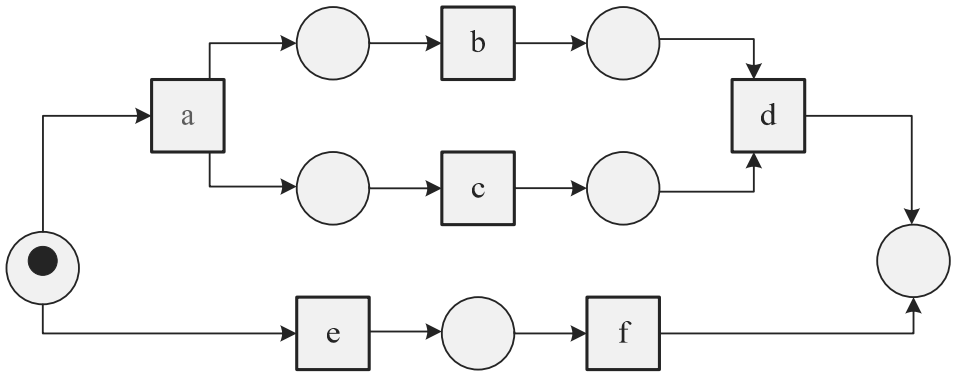


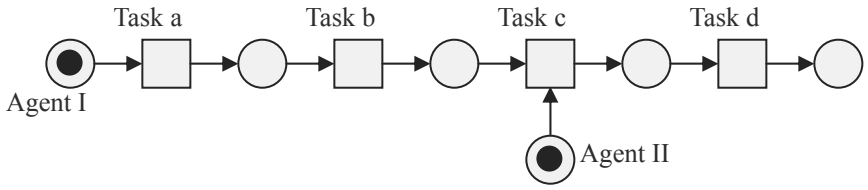
Figure 3 The transferred PN (reference)

Process	Operation	Notation
Case 1	Task a	Beginning Task
Case 1	Task b(Request)	Agent I send a request to Agent II
Case 1	Task c(Agree)	Agent II agree with Agent I
Case 1	Task d	Ending Task

(a) Log file for agent I

Process	Operation	Notation
Case 2	Task b(Request)	Agent I send a request to Agent II
Case 2	Task c(Agree)	Agent II agree with Agent I

(b) Log file for agent II



(c) The PN for agent I and agent II

Figure 4 Information for scenario 1

Basically, the main modification was to find the succession task from the set of duplicated tasks in two agents' log files. Based on the log files shown in Figure 4-(a) and Figure 4-(b), the related PN can be translated according to the following procedure:

1. Find the transition set: T_{W1}={a,b,c,d}, T_{W2}={b,c}.
2. Find the initial transition set: T_I={a,b}.

Find the final transition of each set: T_O={d,c}.

3.3 The Implementation Stage

In this stage, the multi-agent system would be implemented in terms of JADE (Java Agent Development Framework). The experimental agents and their related rules would be built in. During the execution period, the "sniffer agent" in JADE would be used to trace the interaction messages among the agents. The traced information would be written into log files. Then, α algorithm transformation module would be used to generate a related PN model. This model would be put into PN software CPN for simulation. The generated report could be used to verify the feasibility of the multi-agent system.

4. Empirical Verification

The proposed approach has been implemented and verified in terms of a robotic assembly cell example from Zhou's book. (Zhou 1993). In this book, Zhou modeled the assembly behaviors as a PN model and then proved that possible deadlock might occur in this system.

Figure 5 showed the assembly cell. There are

two robots and two assembly stations. Each station required both of the robots to carry out an assembly job. If workstation 1 required performing the task, it would acquire robot 1 first. Once it got the authorization of robot 1, it would acquire robot 2. For workstation 2, the sequence of acquiring robots would in a reverse way. Therefore, it could be found that if both of the workstation 1 and 2 acquired robots at the same time, then, neither of them could get another robot. The assembly cell would be in a deadlock situation.

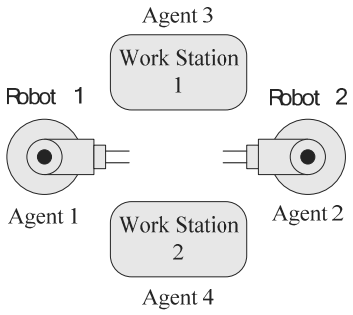


Figure 5 The robotic assembly cell

The empirical assembly cell has been implemented as a multi-agent system in terms of JADE. Figure 6 showed the implementation framework. In general, four agents would be developed to represent the control behaviors of two robots and two workstations respectively. In the JADE distributed agent platform, each agent should register and get a unique ID from Agent Management System (AMS). During the run time, two robot agents should register in the Directory Facilitator (DF) first. Once an assembly workstation agent required a robot service, it could require the DF to search for the available robot agent. These agents would be communicated among one another through the Agent Communication Channel (ACC). Therefore, the ACC would store the communication data. A sniffer agent would be developed to extracted the data and generate the log files for each agent.

4.1 Log Files Generation

Figure 7 showed the window of the sniffer agent and its retracted information. In the left side of the window, it showed that the four developed agents (R1: Robot 1, R2: Robot 2, WS1: Workstation 1, and WS2: Workstation 2) as well as the AMD and DF. The right side of the window showed the sequence diagram for the process of WS1 requesting R1 agent and R2 agent. The communication messages among these agents were shown in the bottom of the diagram. The information about the log files was extracted from the message files.

The sequence diagram showed in the sniffer agent window indicated the sequence for WS1 requesting R1 and R2 robots. Mainly, the WS1 would send a “request” to DF for searching required robot. Then, DF would acknowledge WS1’s request through “inform” message. In the third stage, W1 would “request” R1 for service, and R1 would “agree” the service in the fourth stage. Then W1 would “request” R2 in the fifth stage. The information and sequence about these tasks can be extracted from the message file of sniffer agent and converted into the log file for W1 agent as shown in Table 2.

The information about the log files for W2, R1 and R2 also can be extracted from the same message file and were shown in Table 3, Table 4 and Table 5. It can be found that W2 would be refused by R2 since it already assigned to W1. As for the log file for R2, it showed that it agreed the request from W1 and refused R2.

4.2 Log Files Conversion to PN Model

Once a log file has been created, the next stage was to convert the log file into a PN model. For the WS1 log files as shown in Table 1, its log file can be shown as {REQUEST, INFORM, REQUEST, AGREE, REQUEST, AGREE, CANCEL, CANCEL} and { REQUEST, INFORM, REQUEST, AGREE, REQUEST, AGREE, CANCEL, CANCEL}. Table 6 showed the messages in the log file and their mapped symbols. It can be found that although the messages for G and H are CANCEL, G was the message sent from WS1 to R1 and H was sent from WS1 to R2.

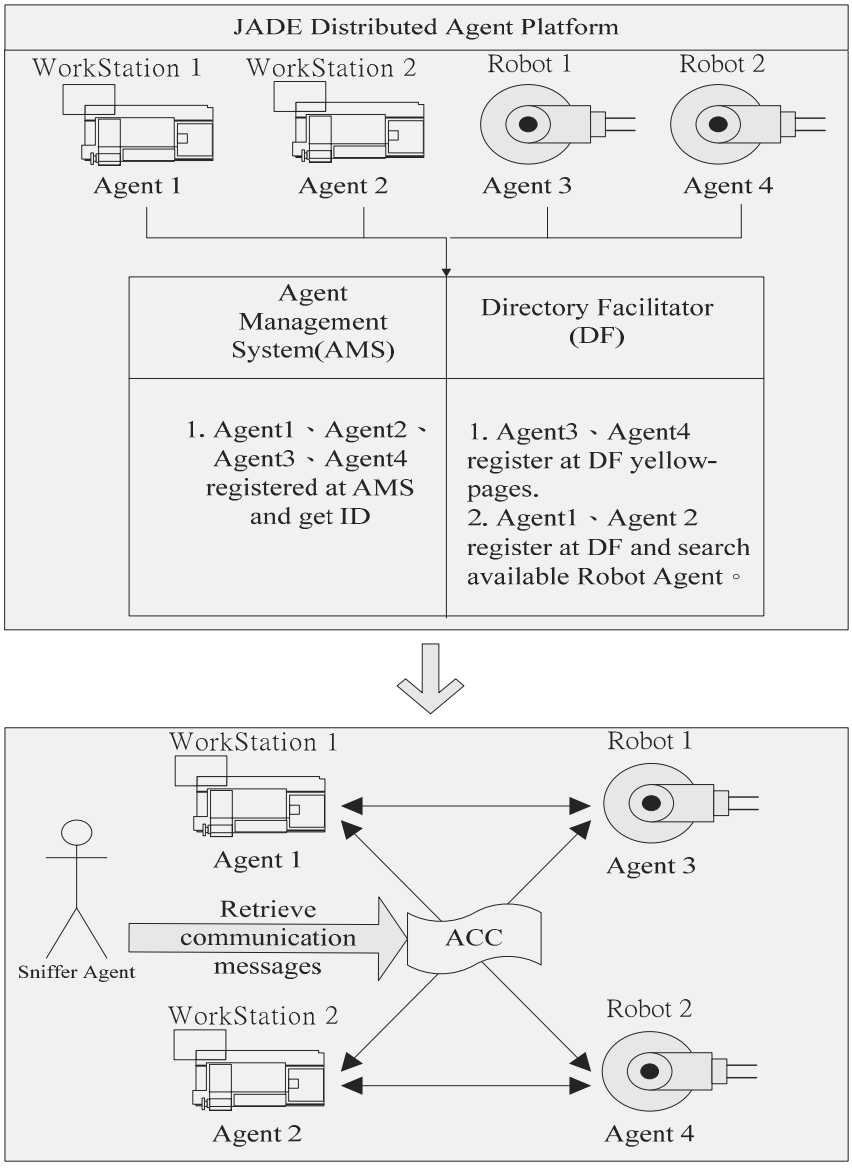


Figure 6 Robotic assembly cell in JADE

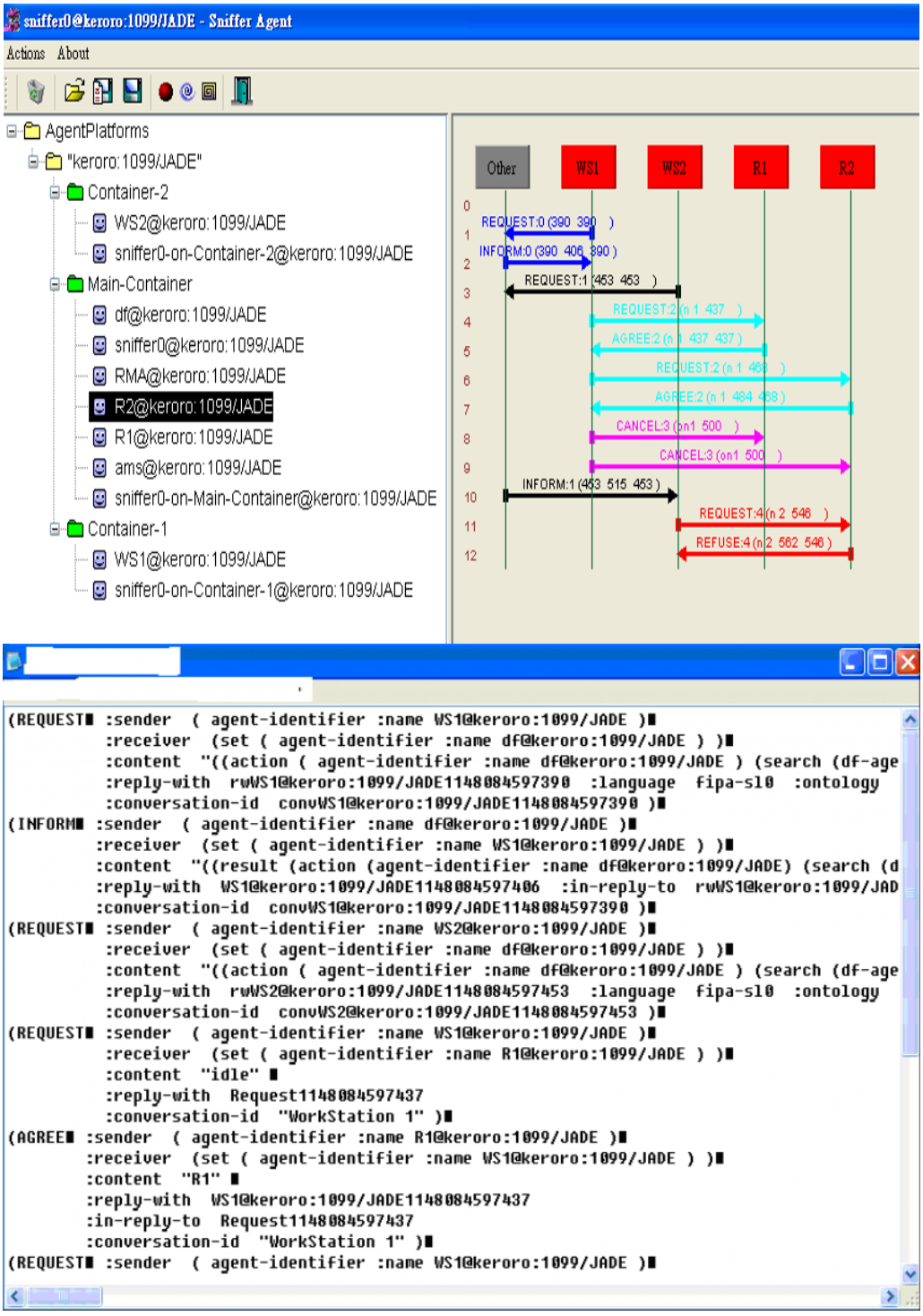


Figure 7 JADE window and the generated communication messages

Table 2 The log file for W1 (workstation1) agent

Stage	Messages	sender/receiver
1	REQUEST	WS1/DF
2	INFORM	DF/WS1
3	REQUEST	WS1/R1
4	AGREE	R1/WS1
5	REQUEST	WS1/R2
6	AGREE	R2/WS1
7	CANCEL	WS1/R1
8	CANCEL	WS1/R2

Table 3 The log file for W2 (workstation2) agent

Stage	Messages	sender/receiver
1	REQUEST	WS2/DF
2	INFORM	DF/WS2
3	REQUEST	WS2/R2
4	REFUSE	R2/WS2

Table 4 The log file for R1 (Robot1) agent

Stage	Messages	sender/receiver
1	REQUEST	WS1/R1
2	AGREE	R1/WS1
3	CANCEL	WS1/R1

Table 5 The log file for R2 (Robot2) agent

Stage	Messages	sender/receiver
1	REQUEST	WS1/R2
2	AGREE	R2/WS1
3	CANCEL	WS1/R2
4	REQUEST	WS2/R2
5	REFUSE	R2/WS2

According the log file, the respective PN can be transferred as the following steps:

1. Find the transition set: T_W= {A, B, C, D, E, F, G, H}.
2. Find the initial transition set: T_I= {A}.
3. Find the Final transition set: T_O= {G, H}.
4. Find the set of transitions which has only one directional precedence: X_W= {(A, B), (B,C), (C,D), (D,E), (E,F), (F,G), (F,H)}.
5. Remove the duplicated transition set. There was no duplicated set in this log file. So, Y_W= X_W = {(A, B), (B,C), (C,D), (D,E), (E,F), (F,G), (F,H)}.
6. Develop the place set: P_W={i_W, $p_{(A,B)}$, $p_{(B,C)}$, $p_{(C,D)}$, $p_{(D,E)}$, $p_{(E,F)}$, $p_{(F,G)}$, $p_{(F,H)}$, o_W}.
7. Develop the arrow set to link the places and transitions: F_W={(i_W, A), (A, $p_{(A,B)}$), ($p_{(A,B)}$, B), (B, $p_{(B,C)}$), ($p_{(B,C)}$, C), (C, $p_{(C,D)}$), ($p_{(C,D)}$, D), (D, $p_{(D,E)}$), ($p_{(D,E)}$, E), (E, $p_{(E,F)}$), ($p_{(E,F)}$, F), (F, $p_{(F,G)}$), (F, $p_{(F,H)}$), ($p_{(F,G)}$, G), ($p_{(F,H)}$, H), (G, o_W), (H, o_W)}.
8. Complete the final PN: $\alpha(w)=(P_W, T_W, F_W)$.

The next task was to transfer the log files for R1 and R2 into the model. From Figure 8, it can be found that the messages for the three stages of the R1 agent were the same as the stage 3, 4 and 7 of W1 agent (link 1 and 2 in Figure 9). In addition, W1 played an active role and R1 played a passive role in this communication. According to the modified algorithm, a place can be added to the first "AGREE" transition in the PN model. The similar situation also can be found in the log files for W1 and R2 agents (link 3 and 4 in Figure 8). Therefore, another place can be added to the second "AGREE" transition in the PN model. The PN model from W1 agent point of view was shown in Figure 9-(a), and the definition of places and transitions were shown

in Table 7.

The third task was to build the PN model based on the log file of W2 agent (Table 3) and to add the information about R2 agent (link 5 in Figure 8) into the model. The PN outcome was shown in Figure 9-(b).

The constructed PN model was simulated through CPN tool. As mentioned before, the empirical assembly cell would encounter a deadlock if W1 issued a request to R1 and W2 sent a request to R2 at the same time. This situation can be simulated through the PN model by issuing two tokens at place "WS1" and WS2" respectively. The CPN report was shown in Table 8. It can be found that two tokens were halted in place P5 and P12 respectively. From the PN model, P5 indicated that W1 was waiting the response from R2, and P12 specified that W2 was waiting the response from R1. Since both of the assembly workstation occupied a robot and waited another robot, a deadlock would happen.

Table 6 The messages and its mapped symbols for the log file of WS1

Messages	REQUEST	INFORM	REQUEST	AGREE
Symbol	A	B	C	D
sender/receiver	WS1/DF	DF/WS1	WS1/R1	R1/WS1
Messages	REQUEST	AGREE	CANCEL	CANCEL
Symbol	E	F	G	H
sender/receiver	WS1/R2	R2/WS1	WS1/R1	WS1/R2

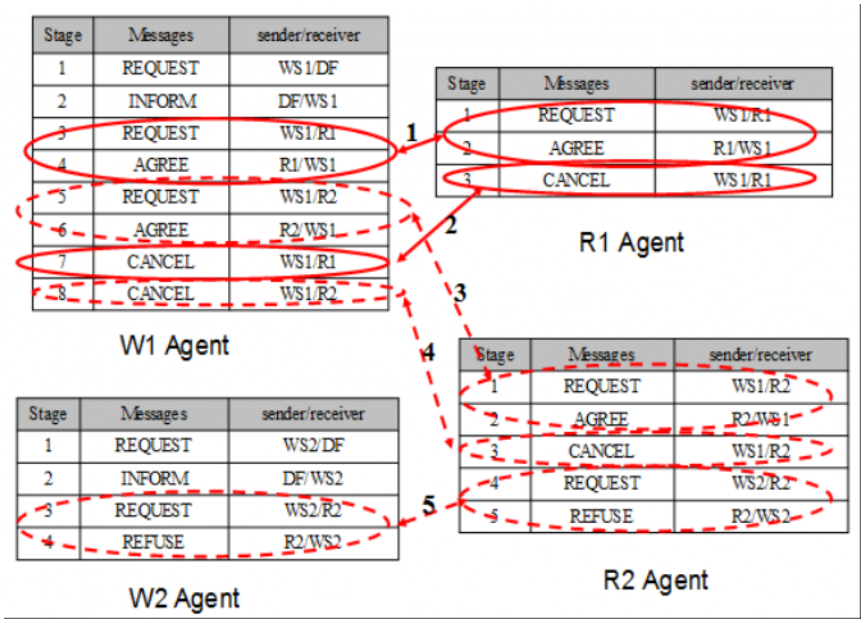
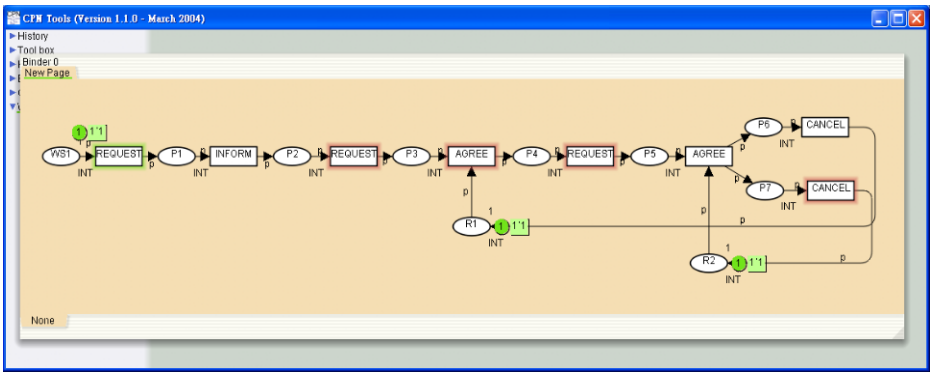
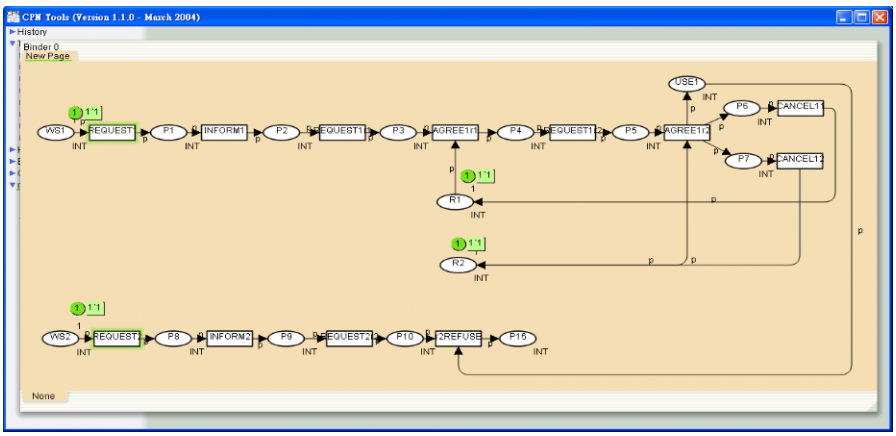


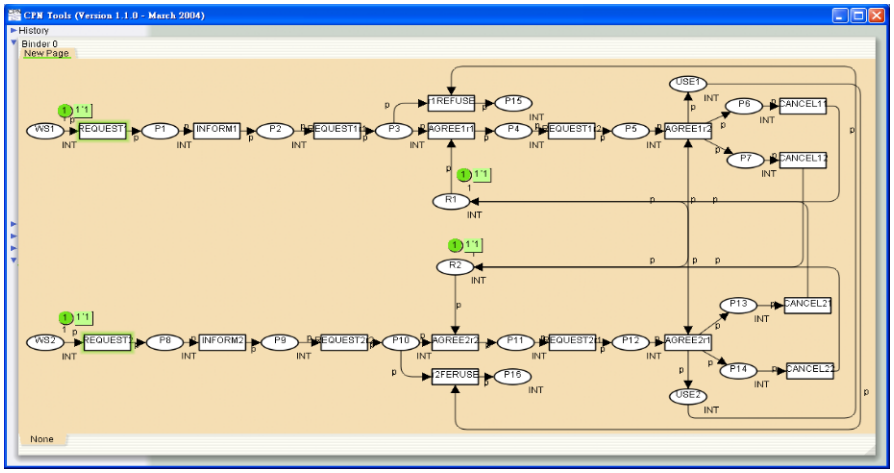
Figure 8 The interactions among the four agents



(a) W1 agent oriented PN model



(b) W2 agent oriented PN model



(c) The final PN model

Figure 9

5. Discussion and Conclusions

This research tried to bridge the gap between the development of a multi-agent system and the related PN model, such that the developed PN model can be used to simulate and to assist the analyzing the feasibility of the multi-agent system. The major contributions of this work were:

Table 7 Definitions of places and transitions in Figure 9-(a)

Place	Definition
WS1	WorkStation Agent 1
P1	Wait for DF response
P2	Get DF response and prepare send message to R1
P3	Wait for R1 response
P4	Get R1 response and prepare send message to R2
P5	Wait for R2 response
P6	Using R1 to assemble
P7	Using R2 to assemble
P15	Set R1 and R2 as "idle"
Transition	**Definition**
REQUEST	Request DF to search the idle robot agent
INFORM	DF inform the search result
REQUEST	Request R1 service
AGREE	R1 agree
REQUEST	Request R2 service
AGREE	R2 agree
CANCEL	R1 finish assembly
CANCEL	R2 finish assembly

1. The concept of verifying the feasibility of a multi-agent system during its developing stage can be accomplished.
2. An approach was developed to extract the log files from a multi-agent system and then to build its respective PN model.

Several possible issues were also worth to be considered. One was that in this work, α algorithm was modified such that the log files for two interactive agents can be transformed to a PN model. The main assumption of the modification was based that two agents should have "direct succession" relation. That was two agents would send a "request" and "response" messages between each other in the beginning of the interaction. However, in the real environment, the phenomena for agent interaction might be more complicated. For instance, if two agents both have reactivity property, they might have interaction due to the trigger from outside environment. One possible example was that if the inventory level was lower than the safety level, an inventory control agent might send a replenishment order to the supplier agent without sending the "request" message first. Since these agent might not have a common "request", "response" messages in their log files, the modified α algorithm could not link these agents in a PN model.

Another possible future work was to use the transferred PN model to assist the design of a multi-agent system. This research was only addressed on transferring the log files into a PN model for the purpose of feasibility analysis. However, the transferred PN could be used to assist the re-design of the agent system. For instance, the PN properties such as place invariant and transition invariant can be used to control certain behaviors of an agent-based manufacturing system. Therefore, it is possible to add certain place and transition constructs into the converted PN model such that the related agent system can fulfill certain phenomena. In order to achieve this, an inverse-α algorithm might also need to be conducted.

Table 8 CPN simulation results

Boundedness Properties		
Best Integers Bounds	Upper	Lower
New'P10	0	0
New'P1	0	0
New'P11	0	0
New'P12	1	1
New'P13	0	0
New'P14	0	0
New'P2	0	0
New'P3	0	0
New'P4	0	0
New'P5	1	1
New'P6	0	0
New'P7	0	0
New'P8	0	0
New'P9	0	0
New'R1	0	0
New'R1	0	0
New'WS1	0	0
New'WS2	0	0
Liveness Properties		
Dead Markings: All		
Dead Transitions Instances: All		
Live Transitions Instances: None		

References

[1] van der Aalst, W.M.P., Weijters, A.J.M.M. & Maruster, L. (2004). Workflow mining: discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering, 16 (9): 1128-1142

[2] van der Aalst, W.M.P., Rubin, V., van Dongen, B.F., Kindler, E. & Gunther, C.W. (2006). Process mining: a two-step approach using transition systems and regions. BPM Center Report BPM-06-30. Available via DIALOG. http://wwwis.win.tue.nl/~wvdaalst/BPMcenter/reports.htm

[3] Agrawal, R., Gunopulos, D. & Leymann, F. (1998). Mining process from workflow logs. In: Sixth International Conference on Extending Database Technology, 469-483

[4] Cardid, M., Cigolini, R. & Marco, D.D. (2005). Improving supply chain collaboration by linking intelligent agent to CPFR. International Journal of Production Research, 43 (20): 4191-4218

[5] Chiu, M. & Lin, G. (2004). Collaborative supply chain planning using the artificial neural network approach. Journal of Manufacturing Technology Management, 15 (8): 787-796

[6] Damm, W., Hungar, H. & Olderog, E.R. (2006). Verification of cooperating traffic agents. International Journal of Control, 79 (4): 395-421

[7] Deen, S.M. & Jayousi, R. (2006). A preference processing model for cooperative agents. Journal of Intelligence Information System, 126: 115-147

[8] DeLoach, S.A. (1999). Multiagent systems engineering: A methodology and language for designing agent systems. In: Proceedings of Agent-Oriented Information Systems (AOIS), 45-57

[9] Fan, Y., Huang, C., Wang, Y. & Zhang, L. (2005). Architecture and operational mechanism of networked manufacturing integrated platform. International Journal of Production Research, 43 (12): 2615-2629

[10] Forget, P., D'Amours, S. & Frayret, J.-M. (2008). Multi-behavior agent model for planning in supply chains: an application to the lumber industry. Robotics and Computer-Integrated Manufacturing, 24 (5): 664-679

[11] Kowalczyk, R., Franczyk, B., Speck, A., Braun, P., Eismann, J. & Rossak, W. (2002). Intermarket-towards intelligent mobile agent e-marketplaces. In: Proceedings of 9th Annual IEEE International Conference and Workshop on Engineering of Computer-Based Systems, 268-275

[12] Kuk, S.H., Kim, H.S., Lee, J.K., Han, S. & Park, S.W. (2008). An e-engineering framework based on service-oriented architecture and agent technologies. Computers in Industry, 59 (9): 923-935

[13] Lau, J.S.K., Huang, G.Q., Mak, K.L. & Liang, L. (2005). Distributed project scheduling with information sharing in supply chains, Part I: an agent-based negotiation model. International Journal of Production Research, 43 (22): 4813-4838

[14] Lau, J.S.K., Huang, G.Q., Mak, K.L. & Liang, L. (2006). Agent-based modeling of supply chains for distributed scheduling. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 136 (5): 847-861

[15] Li, Y., Jian, J., Yan, R. & Liao, W. (2009). Aircraft tooling collaborative design based on multi-agent and PDM. Concurrent Engineering, 17 (2): 139-146

[16] Lin, F.R. & Chang, K.Y. (2001). A multi-agent framework for automated online bargaining. IEEE Intelligent Systems, 16 (4): 41-47

[17] Lu, T.P., Chang, T.M. & Yih, Y. (2005). Production control framework for supply chain management-an application in the elevator manufacturing industry. International Journal of Production Research, 43 (20): 4219-4233

[18] Nwana, H., Ndumu, D., Lee, L. & Collis, J. (1999). ZEUS: a tool-kit for building distributed multi-agent systems. Journal of Applied Artificial Intelligence, 13: 187-208

[19] Rozinat, A., De Medeiros, A.K.A., Gunther, C.W., Weijters, A.J.M.M. & van der Aalst, W.M.P. (2008). The need for a process mining evaluation framework in research and practice. Lecture Notes in Computer Science, 4928: 84-89

[20] Shen, W., Hao, Q., Yoon, H.J. & Norrie, D.H. (2006). Applications of agent-based systems in intelligent manufacturing: an updated review. Advanced Engineering Informatics, 20: 415-431

[21] Schoop, R., Neubert, R. & Colombo, A.W. (2001). A multiagent-based distributed control platform for industrial flexible

production systems. In: IECON'01, The 27th Annual Conference of the IEEE Industrial Electronics Society, 1: 279-284

[22] Stadtler, H. (2005). Supply chain management and advanced planning-basics, overview and challenges. European Journal of Operational Research, 163 (3): 575-588

[23] Wen, L., Wang, J. & Sun, J.G. (2006). Detecting implicit dependencies between tasks from event logs. Lecture Notes in Computer Science, 3841: 591-603

[24] Weijters, A.J.M.M. & van der Aalst, W.M.P. (2003). Rediscovering workflow models from event-based data. Integrated Computer-Aided Engineering, 10 (2): 151-162

[25] Wooldridge, M., Jennings, R. & Kinny, D. (2000). The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems, 3: 285-312

[26] Zhang, W.J. & Xie, S.Q. (2007). Agent technology for collaborative process planning: a review. International Journal of Advanced Manufacturing Technology, 32: 315-325

[27] Zhou, M. (1993). Petri Net Synthesis for Discrete Event Control of Manufacturing Systems. Kluwer Academic Publishers, Massachusetts

C. Ou-Yang is a professor in the Department of Industrial Management, National Taiwan University of Science and Technology (NTUST), Taiwan, China. He received his PhD degree from Dept. of Industrial and Systems Engineering, The Ohio State University. Currently, he serves as the associate dean for the school of management, and director of the graduate institute of management in NTUST. His main research interests include business process management, concurrent engineering, and collaborative engineering.

Yeh-Chun Juan currently is an associate professor of Department of Industrial Engineering and Management at Ming Chi University of Technology, Taiwan, China. He received his Ph.D. degree in Industrial Management at National Taiwan University of Science and Technology, Taiwan, China in 2003. His current research and teaching interests are in the general area of Business Process Management, Design/Supply Chain Management, Collaboration Commerce and Business Intelligence. He is a member of Electronic Business Management Society (EBMS).