Advances in Systems Science and Applications (2014) Vol.14 No.1 36-49

An Agent-based Distributed Process Planning and Scheduling System

Y.F. Wang, Y.F. Zhang and J.Y.H. Fuh
Department of Mechanical Engineering, National University of Singapore,

Abstract
This paper reports an agent-based approach to develop a distributed process planning and scheduling system to allow geographically dispersed entities to work cooperatively towards the global goals. The system comprises several intelligent functional agents, each performing a specific task while accomplishing the global tasks through the corporation with other agents. For a schedule with an unsatisfactory performance measure, a facilitator agent is able to automatically generate instructions, which is fed back to the corresponding process planning agent (PPA) for updating the solution space. The PPA then employs an optimization algorithm to find a new optimal plan. This new plan, together with the unchanged plans, will come out a new schedule in the scheduling agent. The coordination process continues until a satisfactory process plans/schedule solution is derived. In this way, the schedule is improved while the process plans still possess high quality in terms of the process planning objective. To ensure effective coordination and communication for the involved agents, the system implementation employs the architecture based on a multi-tier application model and an agent development framework.
Keywords Process planning, scheduling, agent.

1 Introduction

Effective process planning and scheduling is increasingly becoming a core component of the manufacturing system. Process planning is the task of transforming a product design into a set of instructions in sequence while scheduling is the task of allocating resources over time to perform a collection of activities. Traditionally, they are treated as two separate tasks, between which the generated gap will cause several problems. Firstly, since the process planning is usually done before the process plan is executed, the process plan may not be applicable when the job is dispatched in an overall schedule due to change of resource availability. Secondly, in process planning, it is often not possible to use the knowledge of the actual situation in the shop floor condition, thus resulting in the lower overall resource utilization and poor on-time delivery performance. Moreover, the dynamic shop floor is full of different kinds of disruptions, which will easily disturb the process plans and schedule being executed infeasible. Thus, in order to maintain the optimum and feasibility of the process plans and schedule, the collaboration between the process planning and scheduling becomes more and more necessary. On the other hand, industrial growth is increasingly dependent on situations where large businesses are distributed. To account for these requirements, it is possible to be realized by the agent technology.

Agent technology derived from distributed artificial intelligence is increasingly being recognized as a promising paradigm for collaborative manufacturing systems. The term "agen" has been defined as"a computer system situated in some environment and capable of autonomous action in this environment, in order to meet its design objectives"[1],with the properties of autonomy, social ability, reactivity, and pro-activeness [2]. A group of such agents form a multi-agent system (MAS), where the participated agents can communicate, cooperate, and interact with each other to achieve their individual objectives as well as global goals of the system [2]. Other advantages of the agent-based system include scalability, modularity and re-configurability.

In this paper, we present an agent-based distributed process planning and scheduling system (ADPPS), which consists of several functional agents. Each agent is able to conquer its individual local problems while accomplishing the global tasks through the corporation with other agents. The overall aim is to improve the performance of the schedule while minimum process planning cost is also desirable. With the help of an optimization algorithm, the process planning agent is able to generate a high quality of process plan for the designed part according to the predefined objective functions. The scheduling agent is capable of generating a schedule based on the involved process plans and the available resources in the shop floor. The facilitator agent plays a coordination role among the different agents to iteratively achieve the global objectives. Such a system is capable of balancing the machine utilization and reducing the tardy jobs for the production of prismatic parts [3,4]. In addition, it also has the capability to deal with machine breakdown scenario that commonly occurs in the production [5].

2 Related work

Since the early 1990s, agent technology has been applied to develop the intelligent manufacturing system by negotiation, coordination, and cooperation. Due to the page limitation, only some typical examples in distributed collaborative systems are investigated. SHADE [6] proposed the information sharing and decision coordination approach for integrated collaborative manufacturing. CIIMPLEX [7] presented an agent-based system for intelligent enterprise integration for manufacturing planning and execution. MetaMorph II [8] developed a hybrid agent-based mediator-centric architecture to integrate process planning and scheduling through the higher level coordination within a supply chain network. Jia et al. [9] demonstrated an agent based system for coordinated product development and manufacture. Wong et al. [10] implemented a hybrid-based multi-agent system for integrating process planning with scheduling in job shops, where each of the jobs and machines is modeled as a local agent. One limitation for this approach is that with the increasing number of resources in reality, the amount of agents will also increase, leading to the overhead communication. The functional decomposition approach adopted in our work can overcome this problem, where each manufacturing activity is encapsulated as an intelligent and autonomous agent. Such an agent-based encapsulation approach can greatly improve the integration of heterogeneous software components, which also facilitates the coordination and cooperation among geographically dispersed departments.

3 A proposed agent-based distributed process planning and scheduling system

3.1 System overview

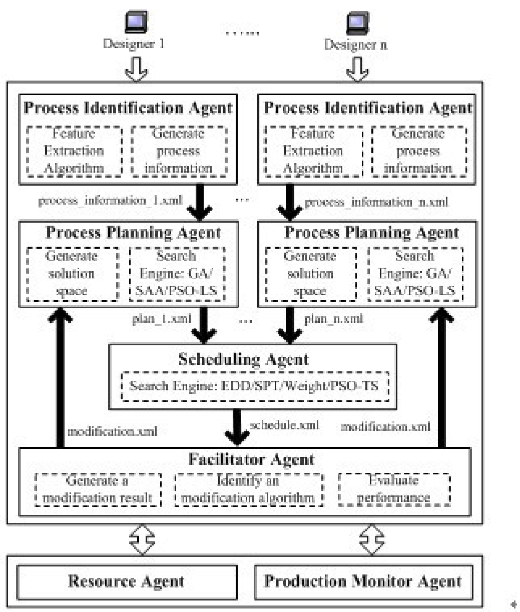

Fig.1 LPM system layout.

Fig.1 presents the framework for the agent-based distributed process planning and scheduling system, which consists of six kinds of agents: resource agent (RA), production monitor agent (PMA), process identification agent (PIA), process planning agent (PPA), scheduling agent (SA), and facilitator agent (FA). These functional agents can be configured in different computers located in geographically different places. All the agents cooperate with each other to achieve a common goal through the internet. In recent years, extensible markup language

(XML) has become an increasingly accepted language for internet-based applications, which provides the benefit of neutral data representation. In compliance with this trend, XML technology is used to encapsulate, extract, and transfer the knowledge among the participated agents. The functionality of each agent is described in the following sections.

3.1.1 Resource agent (RA)

The RA plays a fundamental role in the distributed manufacturing system, which is able to retrieve the resource data for the other agents according to their demands. It is also responsible for managing the resource information in the shop floor, including the specification of machines and tools such as maximum size, achievable accuracy, and surface finish, etc., as well as their manufacturing capabilities. In order to ensure the data consistency, the resource information is dynamically updated. When a machine breaks down, the RA updates its status to be unavailable. Similarly, when it gets recovered, its status turns to be available again. By connecting to each resource entity in the production, RA sends the working status of the current resources to the production monitor agent so as to react the unexpected disruptions.

3.1.2 Process Identification Agent (PIA)

The PIA is used to identify the process information for an input part and the stock used. It follows two steps. Firstly, a feature extraction algorithm [11] is used to partition the delta volume (difference between the stock and part) into a set of volumetric features (VFs), each of which can be removed by a set of operations (e.g. a cylinder can be removed by central drilling + drilling) along one or several tool approach directions (TADs). Secondly, after VF extraction, the process information for all VFs is identified based on the available machining resources managed by the RA. For each VF, all the operation types (OPTs) are determined based on the feature type and the technological requirements (tolerance and surface finish). For instance, to remove a cylinder with certain accuracy requirement, either "central drilling + drilling + boring" or "drilling + milling" can be selected. Therefore, each VF could lead to several sets of OPTs and each may have different number of OPTs. To make every OPT set for a VF having the same number of OPTs, the concept of dummy OPT is introduced, which incurs no manufacturing cost and imposes no precedence relationships with other OPTs. Up to this point, each VF corresponds to a number of OPT sets, each having the same number of OPTs. For each OPT, all the feasible operation methods (OPMs) are formed by the combination of specific machine (M), cutter (T), and tool approach direction (TAD). Furthermore, the precedence relationship between different OPTs is also generated based on fixture constraints, datum dependence, and knowledge of good manufacturing practice. Fig.2 presents the hierarchical representation of the generated process information. In order to transfer this process information using XML file, we define the following XML format according to the presented hierarchical representation:

```
<?xml version=‘‘1.0’’?>
<part part_id=‘‘Part_32’’>
  <volumetric_feature feature_id=‘‘VF_7’’>
     <operation_type opt_id=‘‘OPT_7’’>
        <machines>M1,M2,M3,M4,M5</machines>
        <tools>T14</tools>
        <tads>+z,−z</tads>
        <predecessors></predecessors>
     </operation_type>
     <operation_type opt_id=”OPT_8”>
        <machines>M1,M2,M3,M4,M5</machines>
        <tools>T11</tools>
        <tads>+z,−z</tads>
        <predecessors>OPT_7</predecessors>
     </operation_type>
  </volumetric_feature>

  <volumetric_feature>

  </volumetric_feature>
</part>
```

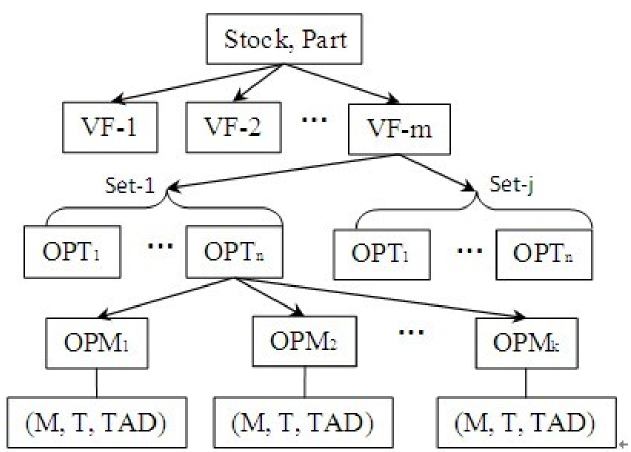

Fig.2 The hierarchical representation of process information based on an input part and stock used.

3.1.3 Production monitor agent (PMA)

Due to the dynamic and stochastic characteristic in nature, a job shop production always faces different kinds of uncertainties (e.g. machine breakdown and rush order), leading to a predictive schedule not applicable. Thus, in order to minimize the impact of the disruption on the production, we need to quickly react to these disturbances and revise the schedule in a cost-effective manner. The task of PMA is to monitor any disruption occurred in the production. Once a disruption is detected, a pre-processing procedure is triggered to pre-process the disruption so that the rescheduling procedure can further handle it. Generally, with respect to the different disruptions, the pre-processing procedure will perform in different ways. For instance, when a rush order comes, it will inform the PPA to generate a feasible process plan for the input part based on the current available machines. When a machine breaks down, it will identify the affected jobs, which are to be processed on this breakdown machine, and then inform the PPA to generate new process plans for them. Note that the stability of the schedule will be maintained and measured in terms of the sequence deviation when revising the disturbed schedule.

3.1.4 Process planning agent (PPA)

The PPA is used to obtain a high quality of process plan for a part, with its input process information from the PIA. The objective of PPA is to select a set of OPMs for each VF and place them into an ordered sequence such that the sequence satisfies the precedence constraints and process plan achieves the predefined objective function. Obviously, process planning problem is a combinatorial optimization problem.

Typically, the criteria for process plan evaluation include minimum number of setups, shortest processing time, minimum manufacturing cost, etc. From the economic point of view, the minimum manufacturing cost is taken as the objective function in this work, which can be considered from the following five cost aspects.

(1) Machine cost (MC)

$$MC = \sum_{i=1}^{n} MCI_i \tag{1}$$

where n is the total number of OPTs and MCI_i is the cost index for the machine used to perform OPTi, which is a constant for a particular machine.

(2) Tool cost (TC)

$$TC = \sum_{i=1}^{n} TCI_i \tag{2}$$

where TCIi is the tool cost index for the tool used to perform OPTi, which is a constant for a particular tool.

(3) Machine change cost (MCC): a machine change cost is required when two adjacent OPTs are performed on different machines.

$$MCC = MCCI \sum_{i=1}^{n-1} (1 - \Omega(M_{i+1}, M_i)) \tag{3}$$

$$\Omega(x, y) = \begin{cases} 1 & if\ x = y \\ 0 & otherwise \end{cases} \tag{4}$$

where MCCI is the machine change cost index and Mi is the identity of the machine used to perform OPTi.
(4) Set-up change cost (SCC): a setup change cost is required when two adjacent OPTs performed on the same machine have different TADs.

$$SCC = SCCI \sum_{i=1}^{n-1} \Omega(M_{i+1}, M_i)(1 - \Omega(TAD_{i+1}, TAD_i)) \tag{5}$$

where SCCI is the setup change cost index.
(5) Tool change cost (TCC): a tool change cost is required when two adjacent OPTs performed on the same machine use different tools

$$TCC = TCCI \sum_{i=1}^{n-1} \Omega(M_{i+1}, M_i)(1 - \Omega(T_{i+1}, T_i)) \tag{6}$$

where TCCI is the tool change cost index.
The above five cost items can be taken either individually or collectively as a cost compound based on the actual requirement and data availability of the job shop. In this paper, all these five items are considered in the objective function, i.e., the total manufacturing cost (TMC):

$$TCM = MC + TC + MCC + SCC + TCC \tag{7}$$

Two steps are followed to find an optimal process plan. The first step is to generate the process plan solution space of the input part, formed by all feasible OPMs that can be used for the fabrication of the part subject to the precedence relationship. The second step is to find the best solution based on a given optimization objective, i.e., minimum manufacture cost, as formulated in Eq.(7). Due to its vast solution space, it is difficult to find the optimal solution for this combination problem in a reasonable amount of time. Two optimization methods based on genetic algorithm (GA) and simulated annealing algorithm (SAA), respectively, have been developed to resolve this intractable problem and promising results were reported [12-13]. Recently, we have developed an alternative algorithm

based on particle swarm optimization [14].
Particle swarm optimization (PSO), being one of evolutionary computation approaches, was firstly proposed by Kennedy and Eberhart [15]. It is a class of population-based optimization algorithm that imitates the social swarm behaviours. Members in the population interact with one another by learning from their own experience and gradually individuals move into better regions of the problem space. The attractive features of PSO include inexpensive computation, individual improvement, and the ability of effective exploration and exploitation search. Due to the characteristic of discrete process planning solution space and the continuous nature of the original PSO, a novel solution representation scheme is introduced for the application of PSO in solving the process planning problem. Moreover, a local search algorithm is incorporated and interweaved with PSO evolution to improve the best solution in each generation. The numerical experiments and analysis have demonstrated that the proposed algorithm (PSO-LS) is capable of gaining a good quality of solution in an efficient way. One example optimal process plan in terms of XML format is shown below:

```
<?xml version=‘‘1.0’’?>
<part part_id=‘‘Part_32’’>
   <operation_type opt_id=”OPT_18”>
      <sequence>1</sequence>
      <opm>M1,T1,+x</opm>
   </operation_type>
   <operation_type opt_id=”OPT_1”>
      <sequence>2</sequence>
      <opm>M1,T1,+x</opm>
   </operation_type>

   <operation_type>

   </operation_type>
</part>
```

3.1.5 Scheduling agent (SA)

The SA is used to generate a high quality of schedule based on the involved process plans from the different PPAs, each specified with the information of weight (1-10), due date, and batch size. A set of dispatching rules [16], including earliest due date (EDD), shortest processing time (SPT), are developed for the generation of schedule. Since the scheduling problem is a typical non-polynomial (NP) hard problem, the quality of solution obtained by dispatching rules may not be good. We therefore developed an approximation algorithm by incorporating a tabu search in the PSO for minimizing the tardiness in a flexible job shop scheduling. By this integration, PSO provides a diversity of initial solutions for the tabu search while tabu search performs an exploitation search so as to affect the particles swarm search behavior. This hybrid procedure, taking the advantage of the difference between these two algorithms, proves to be an effective algorithm. The proposed algorithm (PSO-TS) has been tested on various experiments and the results have demonstrated its robustness, efficiency and efficacy in all sets of experiments.

3.1.6 Facilitator Agent (FA)

The objective of the FA is to improve the performance measure of the schedule through the coordination with the other agents. The FA evaluates the schedule obtained from SA, followed by invoking a heuristic rule, which is able to automatically issue a modification suggestion by identifying a particular operation of one job from the scheduled jobs and the resource to be modified. Although the rule can choose several jobs for process plan modification, the strategy employed here is for fine-tuning, thus only one job is chosen in each round of iteration. The generated suggestion will be fed back to the corresponding PPA that the job belongs to and impose the constraints on its solution space. Based on the update solution space, the affected PPA will then re-generate a process plan using an optimization algorithm. This generated optimal process plan, together with unchanged process plans, forms a new schedule in the SA. The coordination process continues until a satisfactory result is achieved. It is noted that the heuristic rules are in association with the optimization scenario and thus performs in different ways. Currently, the system is capable of balancing the machine utilization and minimizing the number of tardy jobs and their tardiness, as well as accommodating the disruption of machine breakdown. The carried out simulation results have proven its effectiveness to achieve a satisfactory plans/schedule solution in a reasonable amount of time [3-5].

3.2 Agent Structure

In a multi-agent system, each agent is essentially an autonomous cognitive entity, which is able to communicate, reason, and react to the events from the external environment. To start, the agent receives a message and stores it in the receiving message queue. By decoding a message from the message list, an agent resolves the problem using its domain knowledge. Once the problem is resolved, the agent will encode the solutions in a well-defined message and return it to the sender agent. The internal structure of an agent typically includes the following components:

(1) Network interface: It couples the agent to the network.

(2) Communication interface: It enables the agent to exchange and understand the messages.

(3) Interaction interface: It allows the agent to have conversation with other agents.
(4) Domain knowledge: It provides enterprise expertise knowledge required to perform the specific task.
(5) Problem solving module: Owning the reasoning and decision making capability, it enables the agent to resolve the problem according to the received message and domain knowledge.

3.3 System architecture

Fig.3 presents the system architecture for the agent-based distributed process planning and scheduling system, which is based on the multi-tier application model, consisting of presentation tier, business tier, and EIS (Enterprise information system) tier. Moreover, in order to ensure the effective coordination and communication among the involved agents, each agent sits on a platform based on JADE (Java Agent DEvelopment Framework), which is a software framework to develop distributed agent-based applications (http://jade.tilab.com).
In the presentation tier, a user can operate one or more agents to accomplish the required tasks in the local machine through the graphic user interfaces (GUI). Depending on the role of the user, one may have different privileges to access different kinds of agents. As such, these agents configured at geographically dispersed locations forms a peer-to-peer network, where the loosely coupled agents can communicate and cooperate with each other to fulfill the task. Being located at the JADE platform, each agent has a lifecycle. Some agents are shut down when its behavior is completed while some are always active to repetitively execute the specific task. To manage these agents, a directory facilitator agent (DFA) is used to search and modify the description of registered agents. When an agent starts up (shuts down), it will be registered (deregistered) to the DFA. The business-tier located on the server side provides a set of functionality for business logic processing. It comprises two parts. The first part is to create an instance of runtime container for all active agents based on JADE platform. The agent in the container is taken as an independent and autonomous process with a unique identity. In order to fulfil the task, it requires communicating with other agents. Such a communication is realized through asynchronous message passing with an Agent Communication Language (ACL) in terms of a well-defined semantics. To implement it, some kernel packages are utilized, including JADE Core, JADE ACL, JADE Content, JADE Domain, JADE MTP (Message Transport Protocol), and JADE Protocol. However, the modeled agents cannot perform the specific task, since they are not endowed with capabilities except those of communication and interaction. We therefore develop different kinds of algorithms placed in the business tier, which are eligible for each agent to invoke in order to accomplish the assigned task. Note that the feature extraction algorithm is written in C++

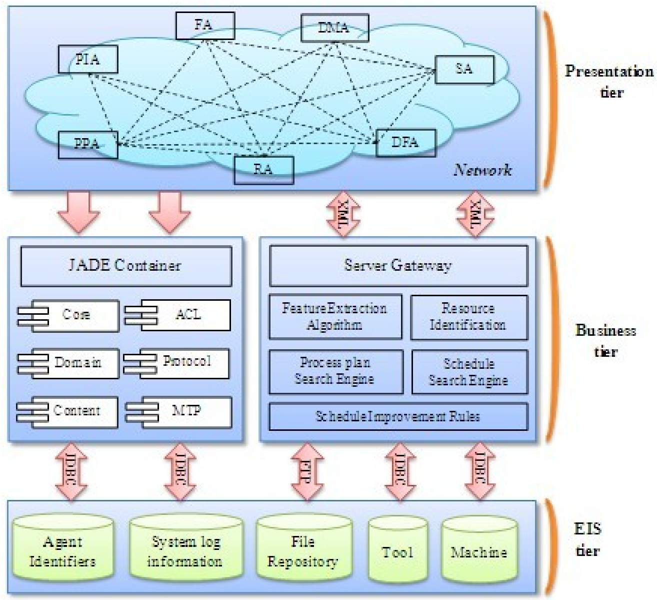

Fig.3 System architecture

language. Since the framework is developed based on the Java Virtual Machines (JVM) and Remote Method Invocation (RMI), agents cannot directly invoke this algorithm. We therefore utilize the Java Native Interface (JNI) technology, which allows a java application running in the JVM to operate with other applications or libraries written in other languages. The EIS-tier plays a critical role in providing information infrastructure to the business process of an enterprise, which may include data integration, the existing application system integration, and legacy system integration. In this work, we only concern about the data integration, focusing on integrating the existing data with the developed system. In this way, the business process and data can be easily shared. The connection to the data stored in a relational database is realized by the JDBC technology. Based on this architecture, the agents in the client tier communicate and exchange the data with the EIS-tier through business tier while the business-tier functions manipulate the data from EIS-tier and client-tier. By logically separating the presentation tier, the business tier, and the EIS tier, the presented architecture brings the following benefits:

(1) It only needs to define the business logic once within the business tier, which can be shared by any agent in the presentation tier.

(2) It is possible to change the contents of any tier without having to make corresponding changes in the others.

(3) It enables parallel development of the different tiers for a complex application.

4 System implementation

A prototype system has been developed to realize the integration of distributed process planning and scheduling based on the proposed architecture, using JAVA language. MySQL database is used to manage the production entities so as to ensure timely information sharing and maintain the data consistency among different functionalities. Moreover, database connection pool is designed to improve the performance of the data retrieval and manipulation. In this way, users in geographically dispersed departments are able to cooperate with each other in a distributed, transactional, and portable environment. Fig.4 illustrates the GUIs of the PPA, SA, and FA.

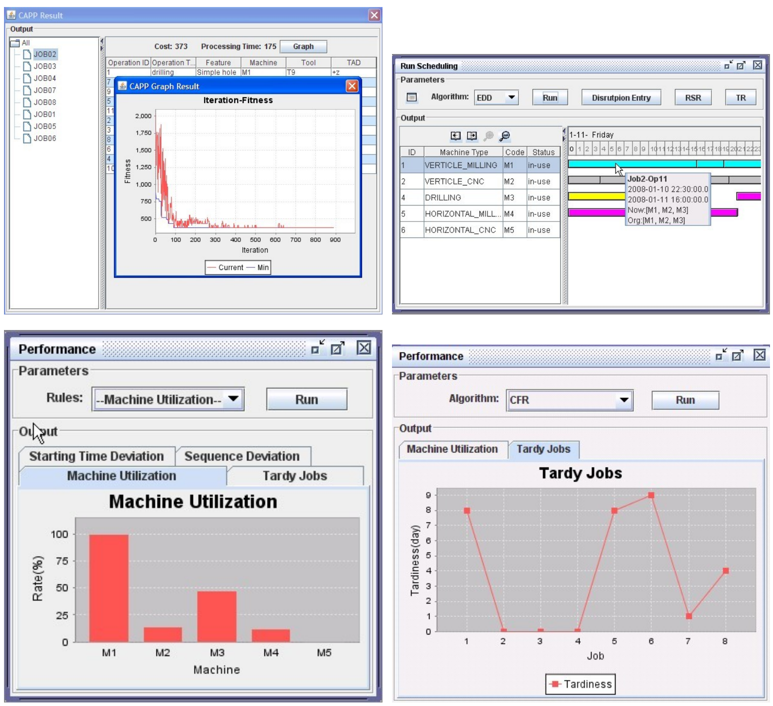

Fig.4 System interfaces of the process planning agent, scheduling agent, and facilitator agent.

5 Conclusion

In this paper, an interoperable intelligent multi-agent system is implemented to realize the integration of process planning and scheduling in a distributed manner. Since both process planning and scheduling are known to be NP-hard in nature, the combined solution space therefore makes it more difficult to find a good solution. We therefore developed an intuitive approach based on the agent technology. Through the iterative coordination among the modelled functional agents, the user delivery requirement will be finally satisfied while the costs of process plans are maintained as low as possible. To develop the prototype system, multi-tier system architecture and the JADE platform, taking the advantage of its flexibility, scalability, reusability, and interoperability, were employed. Although the prototype system based on the agent technology has gained certain success to find a satisfactory solution, more efforts still need to realize its applicability to the actual manufacturing environment. Firstly, the functionalities in a manufacturing system usually interact with each other to give a better product design and achieve a higher production efficiency. Likewise, the design process and shop floor control, which are respectively performed before the process planning and after the scheduling, have a significant impact on the quality of the obtained process plans and schedule. Thus, to achieve a more robust solution, it would be better to incorporate the design agent and the shop floor agent. Secondly, due to various unexpected events in the actual manufacturing, future work is expected to extend its capability to handle more disruptions. Moreover, to meet different users requirement, more performance measures in the scheduling agent should be included.

References

[1] [1] Jennings, N. R., and Wooldridge, M. (1998), "Applications of intelligent agents", in: Jennings, N.R., and Wooldridge, M.J., *Agent Technology: Foundations, Applications, and Markets*, Springer, pp.3-28, .

[2] Wooldridge, M., and Jennings, N.R. (1995), "Intelligent agents: theory and practice", *The Knowledge Engineering Review*, Vol.10, No.2, pp.115-152,

[3] Zhang, Y.F., Saravanan, A.N. and Fuh, J.Y.H. (2003), "Integration of process planning and scheduling by exploring the flexibility of process planning", *Int. J. Prod. Res.*, Vol.41, No.3, pp.611-628, .

[4] Wang Y.F., Zhang Y.F., Fuh J.Y.H., Zhou Z. D., Xue L.G. and Lou P. (2008), "A web-based integrated process planning and scheduling system", *IEEE Int. Conf. on Automation Sci. and Eng.*, Aug. pp.23-26, .

[5] Wang Y.F., Zhang Y.F., Fuh J.Y.H., Zhou Z.D., Lou P., and Xue L.G. (2008), "An integrated approach to reactive scheduling subject to the machine breakdown", *IEEE Int. Conf. on Automation and Logistics*, Sep. pp.1-3, .

[6] McGuire, J.G., Kuokka, D.R., Weber, J.C., Tenenbaum, J.M., Gruber, T.R., and Olsen, G. R. (1993), "SHADE: technology for knowledge-based collab-

orative engineering", *Concurrent Engineering: Research and Application*, Vol.1, No.3, pp.137-146, .

[7] Peng, Y., Finin, T., Labrou, Y., Chu, B., Long, J., Tolone,W.J., and Boughannam, A. (1998), "A multi-agent system for enterprise integration", *Proc. Of PAAM 98*, London, UK, pp.155-169.

[8] Shen, W., Maturana, F., Norrie, D.H. (2000), "MetaMorph II: an agent-based architecture for distributed intelligent design and manufacturing", *Journal of intelligent manufacturing*, Vol.11, No.4, pp.237-251.

[9] Jia, H.Z., Ong, S.K., Fuh, J.Y.H., Zhang, Y.F., and Nee, A.Y.C. (2004), "An adaptive and upgradable agent-based system for coordinated product development and manufacture", *Robotic and Computer-Integrated Manufacturing*, Vol.20, No.2, pp.79-90.

[10] Wong, T.N., Leung, C.W., Mak, K.L. and Fung, R.Y.K. (2006), "Integrated process planning and scheduling/rescheduling-an agent-based approach", *Int. J. Prod. Res.*, Vol.44, No.18-19, pp.3627-3655.

[11] Ahmadi, H. (2008), "Automated volumetric feature extraction from the machining perspective", *Master of Engineering Thesis*, National University of Singapore.

[12] Zhang, F., Zhang, Y.F., and Nee, A.Y.C. (1997), "Using genetic algorithms in process planning for job shop machining", *IEEE transactions on evolutionary computation*, Vol.1, No.4, pp.278-289.

[13] Ma, G.H., Zhang, Y.F., and Nee, A.Y.C. (2000), "A simulated annealing-based optimization algorithm for process planning", *International Journal of Production Research*, Vol.38, No.1), pp.2671-2678.

[14] Wang Y.F., Zhang Y.F., and Fuh J.Y.H. (2009), "Using hybrid particle swarm optimization for process planning problem", *IEEE Int. Conf. on Computational Science and Optimization*, April pp.24-26.

[15] Kennedy, J. and Eberhart, R. (1995), "Particle swarm optimization", *Proceedings of IEEE Int. Conf. on Neural Networks*, Piscataway, NJ, IEEE Press, pp.1942-1948.

[16] Baker, K.R. (1974), "Introduction to Sequencing and Scheduling", New York: Wiley Publications.

Corresponding author
Author can be contacted at: mpefuhyh@nus.edu.sg