A Time-Bound Framework for Negotiation and Decision Making of Virtual Manufacturing Enterprise

Kyoung Jun Lee
Graduate School of Public Administration
Seoul National University

Yong Sik Chang
Graduate School of Management
Korea Advanced Institute of Science and Technology

Hyung Rim Choi, Hyun Soo Kim, Young Jae Park, and Byung Joo Park
Department of MIS
Dong–A University

Virtual manufacturing has 2 characteristics as an agent-based electronic commerce environment: dynamic nature of resource status and variety of agents’ decision-making (i.e., scheduling) model. To reflect the characteristics, a relevant negotiation protocol should be designed and an appropriate decision-making model should be developed. In this article, from the perspective of a sales agent that is a middle man between customers and manufacturers in a virtual manufacturing environment, we provide a case study that suggests a time-bound framework for external negotiation between sales agents and customer agents, and internal cooperation between sales agents and manufacturing agents. We assume a job shop as the production model of a virtual manufacturing enterprise and formulate the optimal order selection problem with mixed integer programming, but its computation time is not acceptable for real-world problems. For this time-constrained decision making, we develop a genetic algorithm as an problem-solving method for the scheduling of the production model, which shows a reasonable computation time for real-world cases and good incremental problem-solving capability.
1. INTRODUCTION

A virtual enterprise is an organization composed of several business partners sharing costs and resources for the purpose of producing a product or service [1]. From the perspective of intelligent agents, a virtual manufacturing enterprise (VME) consists of (a) a sales agent negotiating with customer agents, and (b) manufacturing agents (internal or external) producing and supplying products and services. Sales agents receive requests for bids from customers and communicate with manufacturing agents. The manufacturing agents reply with the information on the manufacturing feasibility, process planning, expected delivery dates, resource availability, and so on. Then the sales agents compose an optimal order set and send bids to customer agents. The customer agents award orders to the sales agent or to another VME. The VME usually has internal manufacturing facilities and a tight or loose relationship with external manufacturers. When a VME cannot satisfy all orders from customers due to technical or resource capability, it might begin to negotiate with external manufacturers by sending a request for bids (RFB) to them.

The advancement of global digital networks such as the Internet and its open system architecture will continuously reduce transaction costs. Therefore, it has been commonly expected that the relation between sales function and manufacturing function would be changed from a long-term strategic and cooperative relation to a short-term contractual relation for the effectiveness and the efficiency of both businesses. Although there are some debates on this claim [2–4], this article deals with a VME that forms temporary and flexible networks with external suppliers for its objective. One of the important issues of such a VME is how to negotiate with its customers and manufacturers in real time and an automated manner. Usually, both the customer’s RFB and the manufacturer’s production proposal have a time limit for the sales agent to respond within. This time-constrained environment is common in real-world manufacturing. In a human-to-human negotiation environment, the time constraint is implicitly assumed and is not a serious consideration in designing negotiation protocol. However, in agent-to-agent negotiation for a VME in which a sales agent must respond to many RFBs from customer agents and also simultaneously process many production proposals from manufacturing agents, the time constraint becomes serious both in negotiation and decision making. In this situation, each business entity will need to automate the negotiation process using its agents and they should be able to make their decisions within given time bounds. We expect that most automated negotiation cases in most application areas should be able to consider time constraints for involved agents to make a decision. With this premise, Sections 2 and 3 propose a negotiation and coordination protocol for agent-based automated negotiation between sales agent, customer agents (software or human), and manufacturing agents in a time-bound situation. Based on this protocol, we develop a time-bound decision-making strategy for a VME.

For profit maximization of a VME, it should solve an optimization problem, selecting optimal customers’ orders under the scheduling constraints such as manufacturing feasibility, resource constraints, and customer-given due dates. In this article, we deal with the VME that has a job shop, which is the most generic model of manufacturing, and assume that the VME is informed of the external or internal manufacturers’ resource availability via communication or information sharing.
For the exact optimization of profits in such a situation, we formulate the order selection problem with mixed integer programming (MIP) in Section 4, but the computation time of the MIP is not acceptable for real-world problems.

As such, the dynamic nature of the real-world VME raises another important issue; that is, time-bound decision making. Even though a VME makes a decision on the optimal order sets and sends bids to its customers, there is a probability that a customer will not award the order. In addition, in the meantime, the VME might receive new RFBs from other prospective customers. This dynamic situation requires an incremental decision-making algorithm that both solves a new problem based on the previous solution and shows the advantage in computation time and quality. For the incremental heuristic problem-solving method, we develop a genetic algorithm (GA) in Section 5, which shows a reasonable computation time for real-world cases and a good incremental nature. The accumulated experimental data are summarized and employed as strategic problem-solving knowledge for sales agents. In Section 6, we show the system architecture and Extensible Markup Language (XML) messages of the agent-based negotiation and scheduling system for the dynamic and time-bound VME.

2. CHARACTERISTICS OF VIRTUAL MANUFACTURING AS AN AGENT-BASED ELECTRONIC COMMERCE ENVIRONMENT

This section begins with a typical scenario of the negotiation in virtual manufacturing:

1. Customer agents send RFBs to multiple sales agents representing their corresponding VMEs, asking if the VMEs can deliver a set of products satisfying specified criteria such as price, quality, delivery date, and so on.
2. With given orders included in the RFBs, each sales agent gets information on the manufacturability of products in the RFBs with its own process planning and scheduling system or by communicating with its manufacturing agents.
3. Manufacturing agents check or schedule their facilities and send the information on the feasible order set to the sales agent.
4. With the given proposals from manufacturing agents, the sales agent selects an optimal order set and sends the corresponding bids to customers.

In this scenario, there are several questions, as follows:

1. How frequently does a sales agent process the RFBs?
2. What kind of temporal strategy should manufacturing agents use in responding to sales agents?
3. How does a sales agent optimize its objective function by selecting a set of orders that maximizes profit?
4. What kind of temporal strategy should a sales agent employ in negotiating with customer agents and manufacturing agents?
5. How does a sales agent cope with changes of situation such as new arrivals of RFBs, award of orders, possible cancellation (i.e., decommitment) of orders, and so on?
These issues are interrelated with each other, but they are summarized as the two characteristics of virtual manufacturing as an electronic commerce negotiation environment. The first characteristic is the dynamic nature of the resource status. The resource status of each agent in virtual manufacturing is dynamically changing during the negotiation cycle because manufacturing is inherently a dynamic resource environment. The dynamic nature of virtual manufacturing demands that the negotiation protocol reflect the fact that the status of each agent might be changed during the negotiation cycle. To consider the dynamic nature of negotiation, many negotiation protocols have been extended from the simple contract net protocol [5–9]. In this article, we adopt a time-bound negotiation framework (TBNF)[10], in which agents negotiate through messages with commitment duration (denoted by $T$). When a message has zero-length commitment duration ($T = 0$), it is interpreted that the agent has no commitment in its message. When a message has infinite-length commitment duration ($T = \infty$), the message is interpreted to have an eternal commitment. When a message has a finite-length commitment duration ($T = \alpha, 0 < \alpha < \infty$), the message is interpreted to be valid for the specified duration. With the TBNF, we can address the second and the fourth questions in the preceding list. If a customer agent gives commitment duration $\alpha$ to its message sent to a sales agent, the sales agent should reply within $\alpha$. With the $\alpha$, the response deadline, the sales agent will demand the manufacturing agents reply within a specified time $\beta$ ($\beta < \alpha$). The difference ($\alpha - \beta$) will be the maximum time available for the sales agent deliberating on the optimal order selection and other decisions (Figure 1).

Such time-constrained decision making requires a kind of anytime algorithmic approach. For example, the MAGNET project presents an anytime algorithm for a customer agent to select bids submitted by supplier agents in response to a call for bids in an automated contracting environment [11, 12]. Whereas the MAGNET project deals with bid selection by customers among the multiple bids without a scheduling model, our research deals with order selection by sales agent rather than customer agent and the selection is based on the job shop scheduling model. As such, the two approaches differ in the decision-making entity and its deci-

![Figure 1. Generic decision-making model of virtual manufacturing agents.](image-url)
sion-making model. Another important feature of the VME as an electronic commerce negotiation environment is its variety of agent’s decision-making models (i.e., scheduling models). The scheduling models are related to Questions 1, 3, and 5 in the preceding list. Decision-making models of an agent vary according to the complexity and the features of the manufacturing environment. We might classify the sales agent’s scheduling models as follows:

Model 1. No scheduling but communication with manufacturers.
Model 2. Exact capacity scheduling but no communication with manufacturers.
Model 3. Exact capacity scheduling and communication with manufacturers.
Model 4. Rough capacity scheduling and communication with manufacturers.

The selection of scheduling models reflects the organizational structure of manufacturing enterprises. A pure VME, or an enterprise without any internal manufacturing facilities, can use Models 1 and 4. A purely physical manufacturing enterprise with internal manufacturing facilities will use Model 2. In this situation, agent formulation of the enterprise is unrealistic and a centralized scheduling approach is more desirable. On the other hand, a hybrid VME with both internal manufacturing facilities and relations with external manufacturers (Figure 1) will use Models 3 and 4. The MAGNET project corresponds to Case 1 and our research corresponds to Case 3. The MASCOT architecture [13] corresponds to Case 4, where the high-level coordination agent has the rough capacity scheduling model and communicates with lower level agents that perform integrated process planning and production scheduling.

3. EXTERNAL NEGOTIATION AND INTERNAL COOPERATION IN VIRTUAL MANUFACTURING

A negotiation protocol depends on the transaction environment and situation. In the VME, there are two kinds of coordination mechanisms, as depicted in Figure 2. One is the external negotiation protocol between customer agents and sales agents of VME, and the other is the internal cooperation protocol between sales agents of VME and manufacturing agents of VME.

3.1 Negotiation Between Sales Agents and Customer Agents

Customer agents try to maximize their utility (e.g., minimize the purchasing cost) by negotiating with multiple VMEs that compete with each other. VMEs try to maximize their profit by selecting an optimal order set considering the production resource capacity and the profits of the products. To coordinate the negotiation between the self-interested agents, time-bound commitments are used in the messages such as RFBs, bidding, and awards. As discussed by Lee, Chang, and Lee [10], each agent can generate a message with a commitment duration in its self-interest. In our VME, the customer agent can use committed RFBs \( T = \alpha \) to contact the sales agents sequentially and select one of them. The customer agent can demand a committed bid submission \( T = \gamma \) to sales agents to make a safe choice among the committed al-
alternatives, and sales agents can use the committed bid submission \((T = \gamma)\) for increasing the probability of getting the award from the customer agent by providing a safe choice or maintaining internal consistency for itself and escaping responsibility for possible future rejection of the awards from the customer agent.

### 3.2 Cooperation Between Sales Agents and Manufacturing Agents

Sales agents and manufacturing agents in VME usually maintain a cooperative relationship to efficiently respond to customers’ orders. For a sales agent to respond to a customer agent within the given time limit \(\alpha\), the sales agent will send a request for a process plan to manufacturing agents and request a response time \(\beta\) \((\beta < \alpha)\). Then, the manufacturing agents examine the manufacturing feasibility, generate a process plan, and send it to the sales agent within \(\beta\). The manufacturing agents might attach a commitment duration \(\sigma\) for the efficient utilization of their resources. As a result, the decision-making time available to the sales agent is limited by both customer agent and manufacturing agents. By simple calculation, the sales agent has less time for its deliberation than the minimum \((\alpha - \beta, \sigma)\), and the committed time \(\gamma\) in the bid submission should be shorter than \(\sigma\) \((\gamma < \sigma)\). In this time-constrained situation, the sales agent should choose the mode of problem solving: batch mode or incremental mode. The choice of problem-solving mode depends on the length of the response time bound \(\alpha\) given by customers, the expected \(\sigma\) given by manufacturing agents, the expected time for decision making (i.e., selecting optimal order sets), and the degree of dynamics in order status and resource status of the VME. The next sections deal with how the sales agent solves the optimal order selection problem, which is one of its important decision-making problems.

### 4. OPTIMIZATION MODEL FOR SALES AGENT DECISION MAKING

In this article, we assume that the VME has the job shop, which is the most generic model of manufacturing, and that it knows resource availability of manufacturers
by communicating with them. The decision making of the VME sales agent for selecting optimal order set is formulated based on Manne's MIP formulation [14]. In this formulation, the objective function maximizes the profit of selected orders and constraints are imposed to satisfy due date requirements.

\[
\max \sum_i p_i O_i ,
\]

subject to

\[
\sum_k r_{ijk}(T_{ijk} + p_{ijk}) \leq \sum_k r_{i,j+1,k}T_{i,j+1,k} + (1 - O_i)M . \tag{1}
\]

\[
\sum_k r_{imk}(T_{imk} + p_{imk}) \leq d_i . \tag{2}
\]

\[
(1 - O)M + (1 - O_i)M + M \times Y_{i'j'k} + (T_{ijk} - T_{i'j'k}) \geq p_{i'j'k} . \tag{3}
\]

\[
(1 - O)M + (1 - O_i)M + M \times 1 - Y_{i'j'k} + (T_{i'j'k} - T_{ijk}) \geq p_{i'j'k} . \tag{4}
\]

The notations are as follows: \( p_i \) = profit generated when order \( i \) is produced; \( p_{ijk} \) = the processing time of the \( j \)th operation of order \( i \) on machine \( k \); \( r_{ijk} = 1 \) if the \( j \)th operation of order \( i \) requires machine \( k \), 0 if otherwise; \( d_i \) = due date of order \( i \); \( T_{ijk} \geq 0 \), the starting time of the \( j \)th operation of order \( i \) on machine \( k \); \( M \) = an arbitrary large integer; \( O_i = 1 \) if order \( i \) is selected, 0 if not; \( Y_{i'j'k} = 1 \) if the \( j \)th operation of order \( i \) precedes the \( j' \)th operation of order \( i' \) on machine \( k \), 0 otherwise; \( m \) = the final operation of each order.

This model has three types of constraints. Constraint 1 represents the precedence relations between the operations of a same order \( i \). Constraint 2 represents the requirement that the last operation of each order must be completed before the due date. Constraints 3 and 4 are artificially generated constraints for representing the model's decision on the precedence of operations on a specific machine. To test this optimization model, we used Muth and Thompson's MT6Í6 [15], which is well recognized as a benchmark problem in the job shop scheduling domain. The result of our experiment was successful and satisfactory. However, the ILOG package for solving the MIP was good only up to the size of an 8 jobs \( \times \) 8 machines problem and requires enormous computing time to find a solution in the larger problems. Therefore we need a different approach to solve real-world problems in a reasonable amount of time.

5. GA FOR SALES AGENT DECISION MAKING

5.1 GA for Job Shop Scheduling

Because most negotiations are time-bound in the real world, each negotiation entity cannot spend much time on its deliberation. Especially in the VME, the negotiation entities such as sales agents or manufacturing agents should solve a scheduling problem for their decision making. Because most scheduling problems are NP-hard, the exact optimization method usually needs a much longer computation time than the time limit given by the negotiation partner. Therefore exact optimiza-
tion is not suitable when a quick response is required (i.e., in time-bound situations). Heuristic search techniques can be used to solve larger problems and often forego guarantees of an optimal solution for gains in speed. Thus we suggest a heuristic algorithm. Numerous approaches have been used to solve job shop scheduling problems in the literature. The most popular of these techniques are the branch and bound method [16], shifting bottleneck procedure [17], GAs [18–20], tabu search [21], simulated annealing [22], fuzzy logic [23], and neural networks [24]. GAs are search techniques based on an abstract model of natural evolution and simulated annealing is based on principles of physical science. Tabu search is a simple technique that intelligently guides a search process away from solutions that appear to duplicate or resemble previously achieved solutions. Neural networks are capable of adapting to new environments with little human intervention and can mimic thought processes in solving job shop scheduling problems. However, current neural network approaches to job shop scheduling are only suitable for small-scale problems and require excessive computing time. Fuzzy theory has been applied to solve optimization problems including uncertain elements. Most fuzzy logic manufacturing scheduling procedures fuzzify processing time, due date, maximal delay, and so on.

Among various search methods for job shop scheduling problems, GA has been recognized as a general search strategy and optimization method useful in attacking combinational problems. In contrast to other search methods (e.g., simulated annealing and tabu search) that are based on handling one feasible solution, GA utilizes a population of solutions in its search, giving it more resistance to premature convergence on local minima. This characteristic leads us to use a GA as an anytime algorithm approach in this article. In addition, we selected GA because our scheduling problem does not need to fuzzify variables, but requires very short time to get feasible solutions for the large-scale instance. We briefly introduce representation, initialization, crossover, mutation, and replacement of the GA that we use.

For the representation, we adopt an operation-based representation [25] that is capable of coping with additional constraints of the job shop scheduling problem. For an n-job m-machine problem, it uses an unpartitioned permutation with m repetition of job numbers. Here, each job number occurs m times in the permutation. The kth occurrence of a job number refers to the kth operation in the technological order of this job. In this way we avoid scheduling operations with technological predecessors that have not yet been scheduled. For example, consider the chromosome of three jobs and three machines represented in Figure 3. An index number refers to the kth occurrence of a particular job number. It is used to point to the corresponding operations in crossover. A permutation with repetition of job numbers merely expresses the order in which operations of jobs are scheduled. A chromosome is made by the Giffler and Thomson algorithm [26] to generate active schedules.

For the seed selection, makespan is used as an objective function to evaluate a chromosome [27]. A male parent is randomly chosen in seed size involving superior individuals from ranking population when random value is smaller than a probability value (i.e., 0.9); otherwise one individual is randomly chosen from the population. A female parent is randomly chosen from the population. First, we choose two individuals, then choose the fittest one if a random value is smaller than a probability value. Otherwise we choose the other one. Selected individuals
are returned to the population and can be chosen again as a parent. We use a cross-
over operator modified from the techniques of GOX [28], which assembles one off-
spring from two parental chromosomes. As we can see in Figure 4, a substring is 
chosen randomly from Parent 1. This crossover implants the substring into Parent 
2 at the position where the first gene of the substring is located. Then all genes of 
the substring are deleted with respect to their index of occurrence in the receiving 
chromosome. The process of crossover follows the same procedure after exchang-
ing Parents 1 and 2 with the same interval. Then two offsprings are evaluated and 
the best one is chosen. Changing Parents 1 and 2 with the same interval generates 
Offspring 2.

For the mutation, we adopt neighborhood search-based mutation [29]. The 
neighborhood for a given chromosome can be considered as the set of chromo-
somes transformable from a given chromosome by exchanging the positions of 
three genes (randomly selected nonidentical genes). The permutations of the genes 
together with the remaining genes of the chromosome form the neighbor chromo-
somes shown in Figure 5. Subsequently, we evaluate all neighbor chromosomes 
and the best one is used as the offspring of mutation.

In Figure 5, the mutation comparing all six cases (Cases 1–6) is called six-case mu-
tation; the mutation comparing only five cases (Cases 2–6) is called five-case muta-
tion. The former is used right after crossover, whereas the latter is used as the general 
mutation operator. The next generation replaces the current generation only after 
the new population is completely created. We use elitism in replacement. In incorpo-
rating heuristics into initialization that generates a well-adapted initial population, 
elitism can guarantee to do no worse than conventional heuristics does.

5.2 Incremental Problem-Solving Capability of GA

Through the experiments detailed in [30], we could confirm that our simplified GA 
is more suitable as a scheduling module for a sales agent. However, in addition to 
the reasonable computation time for a real-world-scale problem, the GA should 
show another functionality: incremental problem-solving capability. Especially for 
a dynamic scheduling situation in which new orders dynamically arrive and some 
orders are canceled after solving a scheduling problem, the GA should solve the 
new problem based on the information of the previous solution. In a dynamic envi-
ronment with a new job arriving during production, rescheduling should keep the 
previous schedule while minimizing the completion time of new jobs because the 
due date of the previous jobs is already fixed with the consent of customers. In this 
study, to keep the previous schedule, the incremental scheduling regards the start-
ing time of a new job as the finishing time of the previous schedule. To indicate 
whether an algorithm satisfies the condition of incremental problem-solving capa-
bility, we test two minimal conditions of an algorithm. Given an algorithm, the first 
condition says that given the same computation time, the solution quality of the al-
gorithm in incremental mode is better than that in batch use. The second condition 
says if the solution quality of the algorithm in incremental mode and that in batch
mode are the same, the required time in incremental use of the algorithm for achieving solution quality is shorter than that in its batch use. The two conditions and the notations are as follows:

\[ t_i \): Processing time of algorithm in incremental mode.
\[ t_b \): Processing time of algorithm in batch mode.
\[ q_i(t) \): Solution quality of algorithm in incremental mode running for time \( t \).
\[ q_b(t) \): Solution quality of algorithm in batch mode running for time \( t \).

Condition 1: If \( t_i = t_b \) then \( q_i(t_i) > q_b(t_b) \).

Condition 2: If \( q_i(t_i) = q_b(t_b) \) then \( t_i < t_b \).

To test Condition 1, we ran the GA in batch and incremental mode for solving a 40-job shop scheduling problem. The incremental GA assumes that half of the jobs are scheduled first and the other half are added to the previous solution. As shown in Figure 6, given the same computation time, the incremental mode outperforms the batch mode in solution quality (profit).

To test Condition 2, we ran the GA in batch and incremental mode for solving a job shop scheduling problem varying the number of jobs: 10, 20, 30, 40, 50, and 60. At each run, we observed the stopping time of the algorithm, the time when the GA
converges to a solution and does not improve the solution quality any more. Because both batch and incremental GAs finally converge to similar solution quality, we use the stopping time as the time required to get a same-quality solution. Figure 7 shows that the incremental use of the GA comes to the desired solution quality much earlier than its batch use.

The two figures show that our suggested GA has the incremental problem-solving capability. Figure 7 provide especially valuable data for the sales agent. Such accumulated experimental data will be summarized and employed as strategic problem-solving knowledge for the sales agent. The knowledge corresponds to the concept of the expected performance profile in Zilberstein [31]. As seen in Figure 7, for a 60-job shop scheduling, the sales agent can expect that it will take about 6 to 12 sec to get a good solution. If some experiments in other environments show that a batch algorithm produces the better solution quality although it takes longer, the sales agent can choose either batch use or incremental use, depending on the time available for decision making, which is a function of commitment duration and number of jobs. From Figure 6, we interpret that incremental mode is much more preferable than batch mode when the available time is short (i.e., the commitment duration is short). From Figure 7, we see that incremental mode is more preferable than batch mode when there are many jobs.

6. IMPLEMENTATION OF VIRTUAL MANUFACTURING AGENTS

6.1 Architecture of Agents

The architecture of customer agent, sales agent, and manufacturing agent is depicted in Figure 8.

Each agent has a communication controller that helps agents communicate with other agents. The order manager in the customer agent announces RFBs to potential VMEs. The VME has two kinds of agents: sales agents and manufacturing agents. The bid manager in each sales agent receives RFBs from customer agents and sends

Figure 6. Processing time versus profit of selected orders (number of jobs = 40).

Figure 7. Comparison of stopping times in incremental GA and batch GA.
requests for process plans to manufacturing agents. The manufacturing manager in each manufacturing agent replies with a process plan, which the process planner builds after checking its resource availability and manufacturability. Then the order selector extracts parameter values from the process plan to solve the order selection model using MIP or GA and selects order sets, and the bid manager sends bids to customers. The bid selector in the customer agent selects the best one of several bids from the sales agent. Table 1 describes the main modules of each agent.

### 6.2 Message Structure for Communication Between Agents

We use messages in XML form, which is commonly used in business-to-business electronic commerce. In Figure 9, we give an example of a process plan message used for the scheduling and the optimal order selection, sent by a manufacturing
agent to a sales agent. The manufacturing agent decides manufacturability of each order and generates its process plan for each order. A process plan includes commitment duration as well as parameter values necessary for the selection model.

The sales agent receives this message from the manufacturing agent and the order selector module interprets messages and extracts the parameter values necessary for solving the order selection model. For example, the value of the \(<\text{COMMITMENT_DURATI}O_N>\) tag in Line 8 means \(\gamma\) and \(<\text{ORDER_ITEM }\text{no=\"1\"}>\) tag in Line 10, \(<\text{OPERATION }\text{no=\"1\"}>\) tag in Line 16, and \(<\text{MACHINE }\text{no=\"1\"}>\) in Line 17 means \(i = 1, j = 1,\) and \(k = 1,\) respectively. The value of the \(<\text{DELIVERY_DATE_WANTED}>\) tag in Line 13 means \(d\). Table 2 describes these interpreted values from Line 8 to Line 25.

7. CONCLUSIONS

Virtual manufacturing presents interesting issues for agent-based electronic commerce research and applications. To reflect its dynamic nature and the complexity

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>(p)</td>
<td>(p = 50,\ldots)</td>
</tr>
<tr>
<td>(p_{ijk})</td>
<td>(p_{111} = 4, p_{112} = 3,\ldots)</td>
</tr>
<tr>
<td>(r_{ijk})</td>
<td>(r_{111} = 1, r_{112} = 1,\ldots)</td>
</tr>
<tr>
<td>(d_{1})</td>
<td>(d_{1} = 05/01/2001,\ldots)</td>
</tr>
<tr>
<td>(T_{ijk})</td>
<td>(T = 04/15/2001\ 12:00, T = 04/15/2001\ 16:00,\ldots)</td>
</tr>
</tbody>
</table>

Figure 9. An XML message for a process plan.

Table 2

Values Extracted From a Process Plan Message for Order Selection Model
of the decision-making model (e.g., scheduling model), a relevant negotiation protocol should be designed and an appropriate decision-making model should be developed. This article deals with such issues from the perspective of a sales agent, a middle man between customers and manufactures in a VME. Research from a customer perspective or manufacturer perspective will also provide good research opportunities. This article provides a case study that suggests a negotiation protocol (i.e., TBNF), assuming a job shop as the production model of virtual manufacturing, and develops a GA as an anytime problem-solving method for the scheduling of the production model. There should be various approaches other than the approach used here: different negotiation protocols, different production models, and different scheduling methods, depending on application areas and problem-solving methodologies. Our future research topics include the formalization and the refinement of the dynamic decision-making procedures of sales agents, finding the optimal strategy for sales agents or manufacturing agents in choosing scheduling mode and assignment of time limits, and real-world implementation of the methodology.

REFERENCES


