THE UNIVERSIRT OF CALGARY

Development of an Intelligent Production Scheduling System

by

Jun Sun

A THESIS

SUBMITED TO THE FACTULTY OF GRADUATE STUDIES

IN PARTIAL FULFULLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF MECHANICAL AND MANUFACTURING ENGINEERING

CALGARY, ALBERTA

November, 1999

0-612-49690-2

ABSTRACT

This research is devoted to the development of an intelligent production scheduling system.

The constraints in design and manufacturing aspects are considered in production scheduling. Design constraints are modeled using a feature-based product representation scheme. Manufacturing constraints are defined as available resources, including facilities and persons. Manufacturing requirements, including tasks and precedence constraints for accomplishing these tasks, are described as part of product feature descriptions.

The two intelligent scheduling functions — predictive scheduling and reactive scheduling are implemented in a multi-agent environment. In predictive scheduling, the optimal production task sequences, timing parameters of these tasks, and resource allocation for accomplishing these tasks are identified based upon heuristic search and agent-based negotiation. In reactive scheduling, the rescheduling and agent-based negotiation approaches are used to modify the original schedules for responding to production demand changes and resource changes.

This system has been implemented using Visualworks and tested for production scheduling at Gienow Building Products Ltd.

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my deep gratitude to my supervisor Dr. D. Xue. As a student and as a researcher, I have benefited from his guidance and support throughout my M.Sc program.

I sincerely thank Dr. D. H. Norrie for his guidance and support in the project for Gienow Building Products Ltd.

I would like to thank the Faculty of Graduate Studies, the Department of Mechanical and Manufacturing Engineering, and the Gienow Building Product Ltd. for their generous financial support during my studies at The University of Calgary. I also thank P. Dean, Vice President at Gienow, for his support during the development of the intelligent production scheduling system.

Special thanks are due to Dr. P. Gu for his advice and encouragement throughout my stay at the University of Calgary. Dr. D. H. Norrie and Dr. R. Kremer are gratefully acknowledged for their critical reviews of this thesis as my examining committee members.

All the support staff members of the Department of Mechanical and Manufacturing Engineering are appreciated for their help. Among those I would like to mention are Lynn Banach, Nareeza Khan, Khee Teck Wong, and Daniel Forre.

My deep appreciation goes to my parents, my sisters and brother, for their encouragement, support and love at all time.

TABLE OF CONTENTS

Chapter 1

Introduction

This introductory chapter is organized as follows: Section 1.1 'sketches the research background. Section 1.2 formulates the problems in production scheduling that motivated this research. Section 1.3 presents an overview of this research. Section 1.4 gives the outline of the thesis.

1.1 Background

With growing competition in the global market, the manufacturing industry must utilize the advanced technologies in product development to provide new products with shorter lead-time and better overall performance.

Development of a product undergoes a sequence of processes including conceptual design, embodiment design, detailed design, process planning, production scheduling, manufacturing, inspection, assembly, and so on. With the advances in computer technologies, especially in artificial intelligence (AI) techniques, the traditional product development activities, including design and manufacturing, have been automated by utilizing the advanced design and manufacturing technologies, such as, computer-aided design, automated process planning and production scheduling, CNC machining, flexible manufacturing, quality control, and so on [Kusiak 92; Singh 96].

Among all these product development activities, *production scheduling* is a process to allocate appropriate resources for the required manufacturing tasks and to identify the sequence and timing parameters to accomplish these tasks. Limited resources include facilities, personnel, materials, and so on. A good schedule can reduce the efforts in manufacturing, thus improving the competitiveness of the products.

Production scheduling has gathered much interest from both academia and manufacturing industries over the last three decades. The early work on production scheduling focused on developing methods and algorithms for generating the sequence to complete required tasks considering either only one processor (machine) or multiple processors (machines) [Baker 74; Bedworth 87; Rodammer 89]. In each of those scheduling methods, the optimal schedule is generated to achieve a certain desired goal such as to minimize the total make-span to complete all the selected tasks, or to minimize the mean flow of these selected tasks.

The early researches highly simplified the production scheduling problems encountered in practice. Since the last decade, the research community has realized that existing scheduling methods were not able to solve the actual industrial problems, because the real-life complexity of industrial plants was not considered sufficiently [Graves 81]. First, the industrial scheduling problems are dynamic in nature, i.e., new orders are received continuously during the production process. Second, during the production process, the created schedule may be changed to reflect the changes of product demand and manufacturing conditions. Production demand changes include insertion of an order to be completed within a short period of time that is required by a customer and removal of an order that is cancelled by a customer. Manufacturing condition changes include disturbance events of resources such as machine breakdowns and worker absence.

With the advances in computer technologies, it becomes possible to develop effective scheduling systems to address the real industrial scheduling problems. To this end, solving the reactive scheduling problems has become a major issue in current scheduling researches [Bean 91; Zweben 94; Szelk 94]. Contrary to *predictive* *scheduling*, which generates schedule based on available information before execution of the required production orders, *reactive scheduling* aims at updating the existing schedule to adapt changes in production environment during the production process [Zweben 94; Kott 98]. An effective production scheduling system should be able to track, react to, and compensate for changes in the shop floor environment. The research presented in this thesis focuses on developing an intelligent production scheduling system using both predictive scheduling approach and reactive scheduling approach.

According to the survey on existing scheduling researches, intelligence scheduling has emerged as a promising technique to be used by industry. Intelligent scheduling methods solve scheduling problems using artificial intelligence (AI) techniques. Intelligent scheduling was initiated by Fox in 1983 for identifying the optimal schedule by incorporating real world constraints [Fox 83]. In Fox's research, constraints were used for guiding the direction of search to identify the feasible and the optimal schedules. Since then, many researches on constraint-based scheduling and rescheduling have been carried out [Zweben 94].

The intelligent approaches have been proved effective for conducting both predictive scheduling and reactive scheduling [Zweben 94]. For predictive scheduling, generally intelligent scheduling approaches aim at identifying the optimal schedule through iterative search process. To improve the search efficiency, many constraint propagation methods, which deduce more constraints from the current available constraints, have been introduced to reduce the search space [Zweben 94]. For reactive scheduling, most researches attempt to revise only partial created schedules for responding to the production environment changes without rescheduling all the required tasks [Bean 91; Zweben 94; Szelke 94].

Both conventional predictive scheduling and reactive scheduling methods were primarily developed based on centralized control architecture, in which all the knowledge base and database were described in the same place [Parunak 96]. This control architecture has difficulty in handling complex manufacturing systems that require knowledge and data in different places. Due to the complexity of real world scheduling problems, improvement in scheduling efficiency, while maintaining the scheduling quality, still needs to be conducted. The recent research in agent-based distributed systems provides new approaches to solve the problems caused by centralized scheduling [Parunak 96]. The goal of this research is to develop an intelligent production scheduling system based on the intelligent scheduling approach and the agent-based distributed modeling approach.

1.2 Objectives

Since the last decade, a number of intelligent production scheduling systems have been developed [Zweben 94]. Despite of the progress, these systems still fall short of being complete scheduling systems. Problems of the conventional production scheduling systems are summarized into the following aspects, which motivated this research.

1. *A method to model manufacturing requirements and associate these requirements with the design descriptions is needed.*

   In the conventional production scheduling systems, manufacturing requirements are given as tasks. Each task specifies its requirements for resources including facilities and persons. These tasks are not associated with the design descriptions/constraints. Therefore, the computer-based design and scheduling systems were developed as separated systems. Since design descriptions also influence the requirements of manufacturing tasks, an integrated scheme for representing product design description and manufacturing requirements should be developed.

2. *A distributed method to model manufacturing resources is required.*

   Manufacturing constraints are usually modeled as resources, such as facilities and persons, in scheduling systems. Conventionally, the scheduling systems were developed using a centralized approach, by which the resource databases in a system were described at the same place and accessed by a centralized control system. This database representation method is effective when the complexity of the manufacturing system is not high. With the increase of the manufacturing system complexity, manufacturing activities often take place at different locations, thus requiring different knowledge bases and databases at different places. Therefore, a distributed resource modeling approach is needed. The recent advances in developing agent-based systems provide new approaches to model the different knowledge bases and databases at different locations, and associate these knowledge bases and databases in problem solving [Parunak 96].

3. *An efficient predictive scheduling method to identify the optimal schedule should be developed.*

   The objective of production scheduling is to identify the optimal schedule, based upon the desired goal such as the minimum production time or the minimum production cost. In scheduling, all the design and manufacturing constraints should be considered. The traditional heuristic search is effective to identify the optimal solution considering the constraints. However, due to the large search space, improvement in search efficiency, while maintaining the search quality, has to be conducted. The recent advances in multi-agent systems provide distributed methods to improve the search efficiency and quality [Parunak 96].

4. *Efficient reactive scheduling approaches for responding to the changes in production environment should be provided.*

   A complete production scheduling system should incorporate two functions: predictive scheduling and reactive scheduling. When a production order is received, the system conducts predictive scheduling to identify the optimal production schedule considering design and manufacturing constraints. When changes of production environment, such as machine breakdowns and person absence, occur during production process, the reactive scheduling approaches are then used to revise the original schedule for responding to these changes. Most of present researches in reactive scheduling attempt to revise only partial original schedules for responding to the production environment changes without rescheduling all the required tasks. To this end, the reactive scheduling approach should minimize the schedule changes, while satisfying the design and manufacturing constraints [Bean 91]. The recent advances in agent-based technology provide a distributed approach for improving reactive scheduling efficiency [Parunak 96; Tharumarajah 97].

1.3 Overview of This Research

The goal of this research is to develop an intelligent production scheduling system to address the motivations listed in Section 1.2. Production scheduling using the developed system is shown in Figure 1.1.

The intelligent production scheduling system has the following three modules: (1) product modeling module, (2) resource management module, and (3) scheduling module. The manufacturing requirements, including production tasks and precedence constraints for accomplishing these tasks, are described using the product modeling module. Manufacturing resources, including facilities and persons, are modeled in the resource management module using multi-agent system approach. When a production order is received, the system conducts predictive scheduling to identify the optimal schedule for producing this ordered product. Reactive scheduling is required for responding to

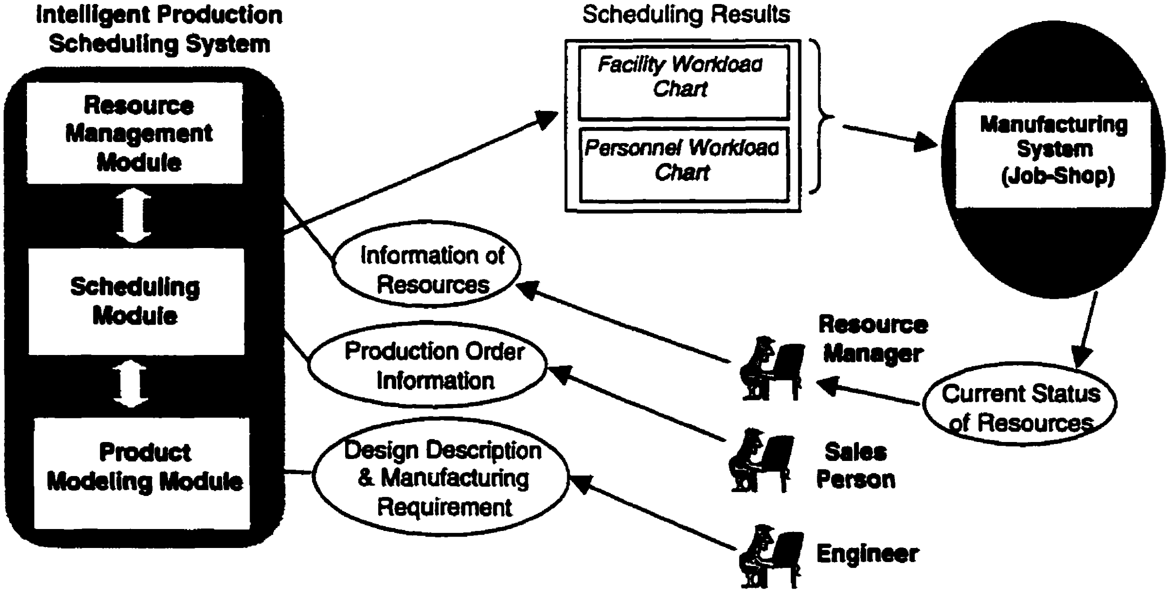

Intelligent Production Scheduling System
Resource Management Module
Scheduling Module
Product Modeling Module
Scheduling Results
Facility Workload Chart
Personnel Workload Chart
Manufacturing System (Job-Shop)
Information of Resources
Production Order Information
Design Description & Manufacturing Requirement
Resource Manager
Current Status of Resources
Sales Person
Engineer

Figure 1.1: The developed intelligent production scheduling system

canceling orders, inserting orders that cannot be scheduled using predictive scheduling due to the urgent due date requirements, facility breakdowns, and personnel absence.

1. *Product modeling module*

   The manufacturing requirements to produce ordered products are described using the product modeling module. In this module, the product primitives are modeled as features [Xue 93]. Features are described at two different levels, class level and instance level, corresponding to generic product libraries and special product data respectively. Instance features are generated using class features as their templates. Manufacturing requirements to produce the products, including tasks and precedence constraints for accomplishing these tasks, are modeled as part of the feature descriptions. Design constraints to be considered in scheduling, such as relations among composing features of products, are also described in this module.

2. *Resource management module*

   The manufacturing resource module is used to organize manufacturing resources, including facilities and persons. The manufacturing resources are described using an agent-based distributed system, in which each resource is modeled as a resource agent. Facility mediator and personnel mediator are used to coordinate the activities of resource agents during the scheduling process.

3. *Scheduling module*

   The scheduling module aims at conducting predictive scheduling and reactive scheduling.

   Predictive scheduling function: Upon receiving requirements from customers, the system generates a production order agent. The order agent negotiates with the resource agents using the corresponding design constraints and manufacturing requirements to identify the optimal production schedule. Heuristic search and agent-based negotiation are employed for the optimal schedule identification.

   Reactive scheduling function: The scheduling module conducts reactive scheduling by changing partial original schedule for responding to the changes such as canceling orders, inserting orders that cannot be scheduled using predictive scheduling due to the urgent due date requirement, facility breakdowns, and personnel absence.

1.4 Organization of the Thesis

This thesis consists of seven chapters and is organized as follows:

Chapter Two describes the background of this research in detail. Section 2.1 introduces the basic concepts in production scheduling. Section 2.2 surveys the literature on conventional production scheduling, new approaches for production scheduling including intelligent scheduling and distributed approach, and constraint consideration in production scheduling. Several technologies used in this research, including feature-based product modeling, agent-based distributed modeling, heuristic search, and an object-oriented programming language: Visualworks/Smalltalk, are introduced in Section 2.3.

Chapter Three enumerates the functional requirements for the intelligent production scheduling system and gives the architecture of this system.

Chapter Four focuses on the predictive scheduling aspect of the intelligent production scheduling system. In Section 4.2, manufacturing requirements, including tasks and precedence constraints for accomplishing these tasks, are described using the feature-based product representation scheme. Section 4.3 introduces the distributed resource management module, in which manufacturing resources including facilities and persons are defined as agents that are coordinated by two mediators: facility mediator and personnel mediator. An intelligent predictive scheduling mechanism is proposed in Section 4.4. The optimal production process sequences and their timing parameters, and resource allocation for producing the products ordered by customers are identified based upon heuristic search and agent-based negotiation.

Chapter Five introduces the reactive scheduling aspect of the intelligent production scheduling system. Two intelligent reactive scheduling mechanisms are used in this system to revise original schedules for responding to production demand changes and resource changes respectively. The intelligent reactive scheduling is conducted based on a match-up rescheduling approach and agent-based negotiation in a multi-agent environment. Section 5.2 introduces the reactive scheduling mechanism for adapting production demand changes, including canceling old orders and inserting urgent orders. Section 5.3 introduces the reactive scheduling mechanism for adapting resource changes, including facility breakdowns and personnel absence.

Chapter Six discusses the issues for implementing the intelligent production scheduling system. Also, this chapter presents some simulation results to evaluate the performance of the intelligent production scheduling system.

Chapter Seven gives conclusions of this work. The future directions of this research are also discussed in this chapter.

Chapter 2

Research Background

This chapter describes the background of this research in detail. Section 2.1 introduces the basic concepts in production scheduling. Section 2.2 surveys the literature on conventional production scheduling, new approaches for production scheduling including intelligent scheduling and distributed scheduling approaches, and constraint considerations in production scheduling. Several technologies used in this research, including feature-based product modeling, agent-based distributed modeling, heuristic search, and an object-oriented programming language: Visualworks/Smalltalk, are introduced in Section 2.3.

2.1 Definition of Production Scheduling

Production scheduling is a process to allocate appropriate resources for the required manufacturing tasks and to identify the sequence and timing parameters to accomplish these tasks. Limited resources include facilities, personnel, materials, tools, and so on.

Scheduling is an optimization process for reaching one or more objectives (also called performance criteria). Solving a scheduling problem is to obtain a feasible schedule that optimizes the given performance criteria [Dauzere-Peres 94]. Typical scheduling performance criteria in manufacturing context are:

- *Throughout (Q)*: the number of orders that the manufacturing system finishes per time unit;

- *Work-in-process inventory (WIP)*: the number of orders in the system which are not finished yet;
- *Order flow time or lead time (F)*: the difference between the order finish time and the order start time;
- *Order tardiness (T)*: the tardiness is the difference between the order finish time and the due date, if this difference is positive. Otherwise, the tardiness is 0.

The production scheduling process is usually subject to constraints. The relationships, limitations, and dependencies among tasks and resources are known as constraints [Beck 98], such as, limited capacity of a resource, and precedence relations among tasks. Due to these constraints, not all the resource allocation decisions are feasible in practical situations. The scheduling process should consider the constraints during exploring the possible solutions.

The different perspectives on the scheduling problems led to different approaches. The *predictive scheduling* approach is to generate schedule based on available information before the execution of the required production orders. The *reactive scheduling* aims at updating the existing schedule for responding to the changes in production environment and demand during the production process [Zweben 94; Kott 98].

The solution of a scheduling problem (the schedule) is often represented as a *Gantt chart* [Gantt 19]. A Gantt chart is a two-dimensional chart, showing time along the horizontal axis and the resources along the vertical axis. Each rectangle on the chart represents a manufacturing task that is allocated to a certain resource in a certain time slot. An example of Gantt chart is shown in Figure 2.1. In this example, The chart represents the generated schedules for three orders *A*, *B*, and *C*. The tasks for these orders are described as *A1*→*A2*→*A3*, *B1*→*B2*→*B3*, *C1*→*C2*, respectively.

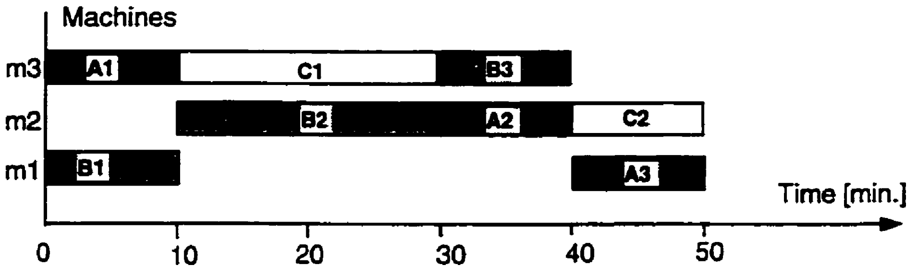


Figure 2.1: Example of a Gantt chart

2.2 Literature Survey

Production scheduling has gathered much interest from both academia and manufacturing industries over the last three decades. The traditional production scheduling researches mainly focused on the predictive scheduling without considering complex real world constraints. With the advance in computer technologies, it has become possible to develop effective scheduling systems to address the real industrial scheduling problems. According to the survey on existing scheduling researches, intelligent scheduling approach and agent-based distributed approach have emerged as two promising techniques to be used for production scheduling [Zweben 94; Parunak 96].

2.2.1 Early Research on Production Scheduling

Traditionally, scheduling research focused on developing methods and algorithms for generating the sequence to complete required tasks considering either only one processor (machine) or multiple processors (machines) [Baker 74; Bedworth 87; Rodammer 89].

In each of these early scheduling approaches, an optimal schedule was generally generated to achieve a certain desired goal such as to minimize the total make-span to complete all the selected tasks, or to minimize the mean flow time of these selected tasks. The sequence and timing parameters of each task on each machine were determined in term of a variety of sequencing rules, such as the SPT (Shortest Processing Time) rule to minimize mean flow time and the EDD (Earliest Due-Date) rule for minimizing max lateness [Bedworth 87].

The early research highly simplified scheduling problems encountered in practice.

- Problems dealt with by the early scheduling approaches were often restricted to a small number of tasks and machines. In the real world, these numbers are significantly larger, and it needs considerable computation efforts to achieve the optimal result. Due to such NP-hard nature of real scheduling problems, it becomes prohibitively time consuming.
- The traditional scheduling research only tried to solve the predictive scheduling or off-line scheduling problems. Those predictive scheduling methods were not able to solve the actual industrial problems, because the real-life complexity of industrial plants was not considered sufficiently [Graves 81]. First, the industrial scheduling problems are dynamic in nature, i.e., new orders are received continuously during the production process. Second, during the production process, the created schedule should be changed to reflect the changes of production demands and manufacturing conditions. Production demand changes include insertion of an order that is required by a customer within a short period of time and removal of an order that is cancelled by a customer. Manufacturing condition changes include disturbance events of resources such as machine breakdowns and worker absence.
- The complex real world constraints were not considered in early production scheduling problems. In real scheduling problems, both design constraints and manufacturing constraints should be considered.

2.2.2 Intelligent Scheduling Approaches

With the advances in computer technologies, especially in artificial intelligence, many advanced intelligent scheduling methods and algorithms have been introduced [Clark 64; Baker 74; Graves 81; Camden 90; Zweben 94].

Intelligent scheduling approaches solve scheduling problems using artificial intelligence (AI) techniques. Typically intelligent scheduling methods depict the scheduling problem as the determination and satisfaction of large number and variety of constraints. Artificial intelligence is used to capture these constraints, to integrate constraints into a scheduling process, to relax constraints when a conflict occurs, and to diagnose poor solutions to the scheduling problem [Rodammer 88].

At the early stage of intelligent scheduling research, most of the work focused on solving the combinatorial problems such as the famous travelling salesman problem using AI-based state space search approaches [Bellmore 87; Winston 92]. At this stage, a scheduling problem was usually to identify the optimal sequence of the selected tasks without considering complex constraints.

The research on intelligent scheduling for solving production problems considering real world constraints was initiated by Fox *et al.* [Fox 83; Fox 89]. In their research, search technique was used to identify the feasible and the optimal schedules, and constraints were used for guiding the search direction. Since then, many researches on intelligent scheduling have been carried out [Zweben 94].

The methodologies of intelligent scheduling are classified into two categories: *constructive approach* and *repair approach*. The constructive approach achieves a complete schedule gradually from a partial schedule using constraints as guidance [Fox 83; Fox 89]. The repair approach, on the other hand, starts with a complete schedule and modifies it iteratively towards the optimal solution [Morris 94; Zweben 92]. Several AI techniques have been employed for constructive scheduling and repair scheduling, including search techniques (heuristic search, neighborhood search, tabu search, and simulated annealing search), genetic algorithm, and neural networks [Fox 89; Zwebean 94; Artiba 97; Jawahar 98].

For the reactive scheduling problem, most research attempts to revise only partial created schedules for responding to the production environment changes without rescheduling all the required tasks. This method is usually called *match-up approach*, which was proposed based on the assumption that enough idle time is presented in the original schedule. In the match-up rescheduling approach, when unexpected events, such as facility breakdown and personnel absence, disrupt the original schedule, the original schedule should be changed partially so that the revised schedule can maximally match up with the original schedule [Collinot 88; Bean 91; Smith 94; Szelke 94; Zweben 94].

The intelligent scheduling approaches have been proved effective for conducting both predictive scheduling and reactive scheduling considering constraints. However, due to the complexity of real world scheduling problem, improvement in scheduling efficiency, while maintaining the scheduling quality, still needs to be conducted. The recent advances in multi-agent systems provide distributed methods to improve the scheduling efficiency and quality [Parunak 96].

2.2.3 Distributed Approaches for Production Scheduling

To achieve higher levels of reconfigurability and adaptability, there is a growing trend toward adopting distributed approaches for organization and management of manufacturing systems [Parunak 96].

Research in distributed approaches focuses on modeling knowledge and actions in collaborative entities [Gasser 91]. Since the 80's, many distributed approaches for production scheduling have been proposed [Shaw 87; Sycara 87; Parunak 90; Butle 94; Saad 95; Maturana 96; Tharumarajah96; Parunak 97].

Conventionally, scheduling systems were developed using centralized control structure, as shown in Figure 2.2 (a), in which manufacturing resources were modeled at the same place, and scheduling was conducted using a central controller. This type of the

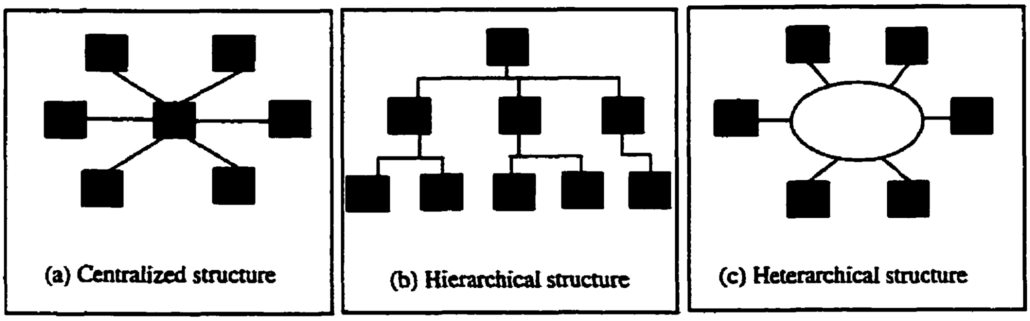

(a) Centralized structure
(b) Hierarchical structure
(c) Heterarchical structure

Figure 2.2: The centralized structure, the hierarchical structure, and the heterarchical structure

structure is effective when the complexity of the manufacturing system is not high. With the increase of the manufacturing system complexity, manufacturing activities often take place at different locations, thus requiring different knowledge bases and databases at different places. Therefore, the centralized approach has such limitations as it makes the system very complicated, and difficult to handle and realize dynamic decision-making if the number of interacting entities is large.

Two typical distributed control structures, hierarchical structure and heterarchical structure, are commonly adopted for production scheduling, as shown in Figure 2.2 (b) and (c) respectively. In the hierarchical structure, entities, such as factories, job-shops, work cells, and machines, are represented by a tree of nodes. Decision making is conducted from higher level nodes to lower level nodes [Parunak 87]. The heterarchical structure is implemented by a collection of independent entities without centralized or explicit hierarchical control in decision making [Duffie 88].

The comparison for the centralized approaches and distributed approaches is summarized in Table 2.1.

Table 2.1: Comparison of distributed approach and centralized approach [Shaw 87]

	Centralized Approaches	**Distributed Approaches**
Structure	centralized	decentralized
Execution of Scheduling	a master scheduler	a number of collaborative entities
Control Mechanism for Scheduling	master-slave control with unidirectional message-passing	coordination through exchanging messages
Vulnerability to Scheduler's Failure	entire system would stop	only the particular entity would be disrupted
Manufacturing Database	a global database	A distributed database
Maintaining Dynamic System Information	constant updating through communication messages	local updating without communication activities

Shaw firstly proposed a distributed scheduling method based on bidding mechanism for a computer-integrated manufacturing environment [Shaw 87]. In the bidding mechanism, among all the potential work cells, the task was assigned to the work cell that could provide the best offer, such as the lowest cost to complete the assigned task, through the communication among the work cells. Parunak developed the Yet Another Manufacturing System (YAMS) [Parunak 87]. In this system, factory, work cells, workstations, and machines were represented as nodes in a hierarchical control structure, in which production tasks were allocated from higher level agents to lower level agents. Duffie, *et al* and Saad *et al.* adopted the heterarchical control structures in manufacturing systems [Duffie 88; Saad 95].

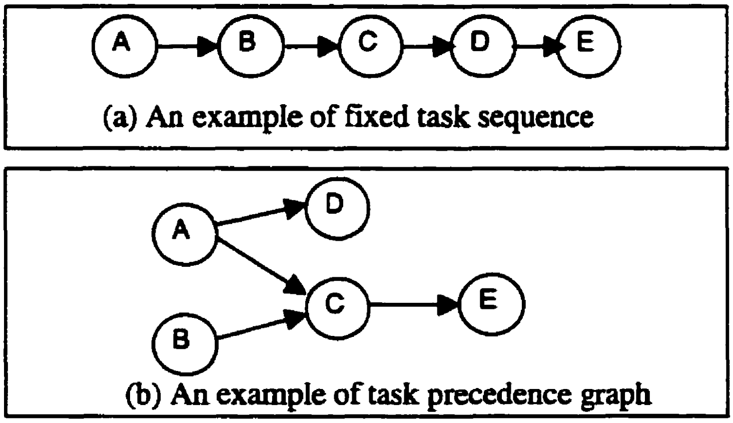
Based on the concept of distributed system, the conventional centralized intelligent scheduling approaches have begun to move towards distributed' intelligent scheduling approaches. Sycara *et al.* developed a job-shop scheduling system using distributed constrained heuristic search [Sycara 91]. This distributed job-shop model had a set of collaborative entities, called scheduling agents. Each scheduling agent was responsible for a set of orders and manages a set of local resources. The local scheduling decision was achieved by the individual scheduling agent using the constraint-directed heuristic search. Allocations of shared resources were conducted through collaboration of the scheduling agents that needed these resources. In the PRIAM (Polite Rescheduler for Intelligent Automated Manufacturing), the match-up rescheduling approach was implemented in a multi-agent environment [Tsukada 96]. In this system, a controller at a disrupted cell tried to respond to schedule disruptions, such as machine breakdowns, in a way which is likely to be least disruptive to other cells, through negotiation with controllers at other cells, thus avoiding wide propagation of the disruption through the rest of the system.

2.2.4 Constraint Consideration in Production Scheduling

With the advances of intelligent scheduling approaches, it is possible to develop scheduling systems to address real world scheduling problems considering complex constraints, such as precedence constraints, time constraints, and resource constraints [Fox 83; Fox 89; Zweben 94].

The task precedence constraints are usually defined in the period of the process planning. The *process planing* defines how products can be made, including the overall process plan defining the possible sequences of tasks or operations to manufacture a product, and the manufacturing requirements for these tasks or operations, including the parameters of the individual manufacturing task or operation. Traditionally, process planning and production scheduling were two separated functions in manufacturing systems. In a traditional manufacturing system, process planning only provided a fixed sequence of tasks or operations to be executed for each product. Figure 2.3 (a) shows an example of this kind representation of task precedence constraint. In this example, the tasks, *A*, *B*, *C*, *D*, and *E*, should be conducted by following a fixed sequence, $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$. Recently, process planning systems are requested to provide flexible task precedence constraints, based on the use of alternative routings and alternative resource allocations, which enable scheduling system to identify the best sequence of tasks depending on available resources and their workloads.

The task precedence graphs has been employed to represent flexible precedence constraints [French 82; Detand 93; Sprecher 94]. In this kind of graph, manufacturing requirements are given as tasks. Each task specifies its manufacturing requirements including required machines and persons, duration time, and so on. As shown in Figure 2.3 (b), the tasks, *A*, *B*, *C*, *D*, and *E*, are linked by arcs representing the precedence



(a) An example of fixed task sequence

(b) An example of task precedence graph

Figure 2.3: Representation of task precedence constraints

constraints. For instance, task *C* must be conducted before task *E*, and after task *A* and task *B*. The task sequence given in Figure 2.3 (a) is only one possible choice that satisfies the task precedence constraints represented in Figure 2.3 (b).

In another approach, Dong *et al.* suggested a feature-based integrated approach for process planning and scheduling [Dong 92]. The precedence constraints were represented using the manufacturing priorities of features, which were extracted from the part to be produced.

However, the task precedence constraints, which are represented by either the task precedence graph or the manufacturing priority tasks, are not associated with the design descriptions and constraints, because currently the computer-based design and scheduling systems were developed as separated systems. Since the design descriptions also influence the requirements of manufacturing tasks, an integrated scheme for representing product design descriptions and manufacturing tasks should be developed.

2.3 Relevant Technologies

In the proposed production scheduling system, the *feature-based modeling* approach is used to model the products and the manufacturing requirements for these products. The manufacturing resources are modeled using *agent-based distributed approach*, in which the agent-based negotiation is employed to facilitate the optimal schedule identification and effective schedule revision. *Heuristic search* is conducted in the distributed multi-agent environment to identify the optimal schedule with improved search efficiency. The system has been implemented using *Visualworks/Smalltalk*, an object-oriented programming language.

2.3.1 Feature-Based Product Modeling

The whole product design process consists of conceptual design, embodiment design, and detail design. The conventional CAD systems are primarily used for geometric modeling, which serves for documenting design results, including 2D and 3D product geometric information, during the last design stage. The geometric modeling method has the following deficiencies [Shah 95]:

- The geometric model only includes geometric information and fails to capture the design intention and generate manufacturing process.
- The geometric elements are organized only according to their topological relations without representation of the functions behind geometry, including design and manufacturing relations among geometric elements for particular purposes.

To solve the above problems, the concept of *feature*, which is described by a collection of relevant geometric elements for particular manufacturing purpose, has been introduced [Grayer 76].

Researchers have given various overlapping definitions of features and their classifications according to their. perspectives and implementation needs [Vickers 88; Shah 95; Xue 93]. Usually a mechanical design can be viewed from three different perspectives: design, manufacturing, and geometry. Therefore, description of design objects from each of the three distinctive perspectives is always given using a set of elementary functional or descriptive primitives, called design feature, manufacturing feature, or geometry feature [Xue 93]. *Design features* are mechanical components and mechanisms, such as gears and shafts, for modeling design candidates based upon the functional requirements. *Geometric features* are geometric primitives, such as blocks, cylinders, to construct the product geometry. *Manufacturing features* are partial product geometry, such as holes and slots, for planning manufacturing processes.

The feature concept can be illustrated using an example of designing a pair of gears, shown in Figure 2.4. The kinematics relations of the gear pair form the product descriptions from the design perspective. The required production process and cost characteristics of the two gears form the product descriptions from the manufacturing perspective. The geometric aspects of gear, such as type of gear, dimensions, and tolerances, form the product descriptions from the geometric perspective.

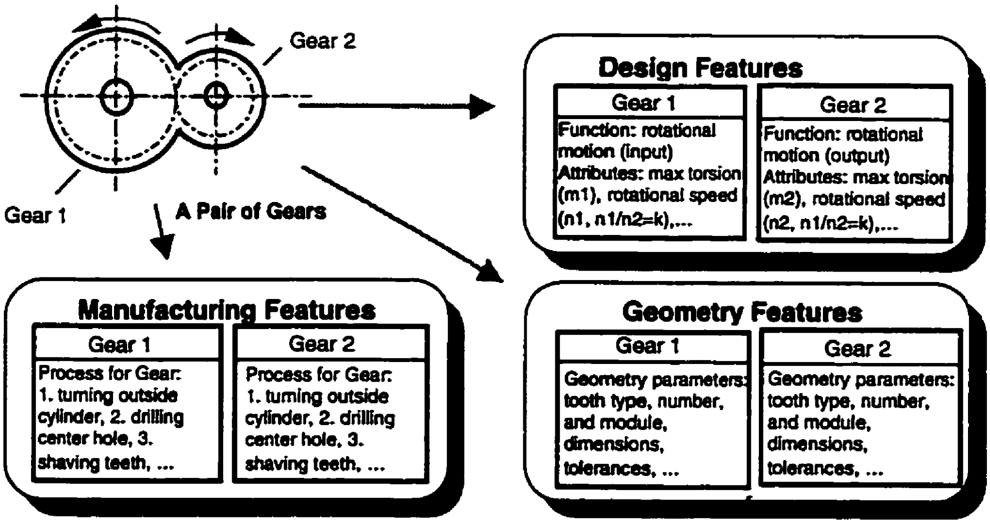

Gear 2
Gear 1
A Pair of Gears
Design Features
Gear 1
Function: rotational motion (input) Attributes: max torsion (m1), rotational speed (n1, n1/n2=k),...
Gear 2
Function: rotational motion (output) Attributes: max torsion (m2), rotational speed (n2, n1/n2=k),...
Manufacturing Features
Gear 1
Process for Gear: 1. turning outside cylinder, 2. drilling center hole, 3. shaving teeth, ...
Gear 2
Process for Gear: 1. turning outside cylinder, 2. drilling center hole, 3. shaving teeth, ...
Geometry Features
Gear 1
Geometry parameters: tooth type, number, and module, dimensions, tolerances, ...
Gear 2
Geometry parameters: tooth type, number, and module, dimensions, tolerances, ...

Figure 2.4: Feature definition from design, manufacturing, and geometry perspectives

The *feature-based modeling* is an approach to use features as the primitives for building up a product. In the feature-based modeling approach, features are first defined in a library, and then used as building blocks for modeling design candidates. The generic schematic of feature-based modeling approach is shown in Figure 2.5.

In this research, the features are defined as basic elements for building up a product from design perspective. The features are described at two different levels, class level and instance level, corresponding to generic product libraries and special product data respectively. Instance features are generated using class features as their templates.

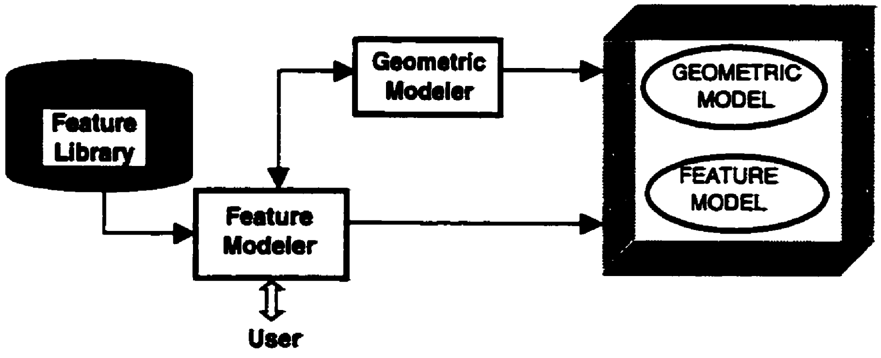


Figure 2.5: Feature-based product modeling [Shah 95]

2.3.2 Agent-Based Distributed Modeling Approach

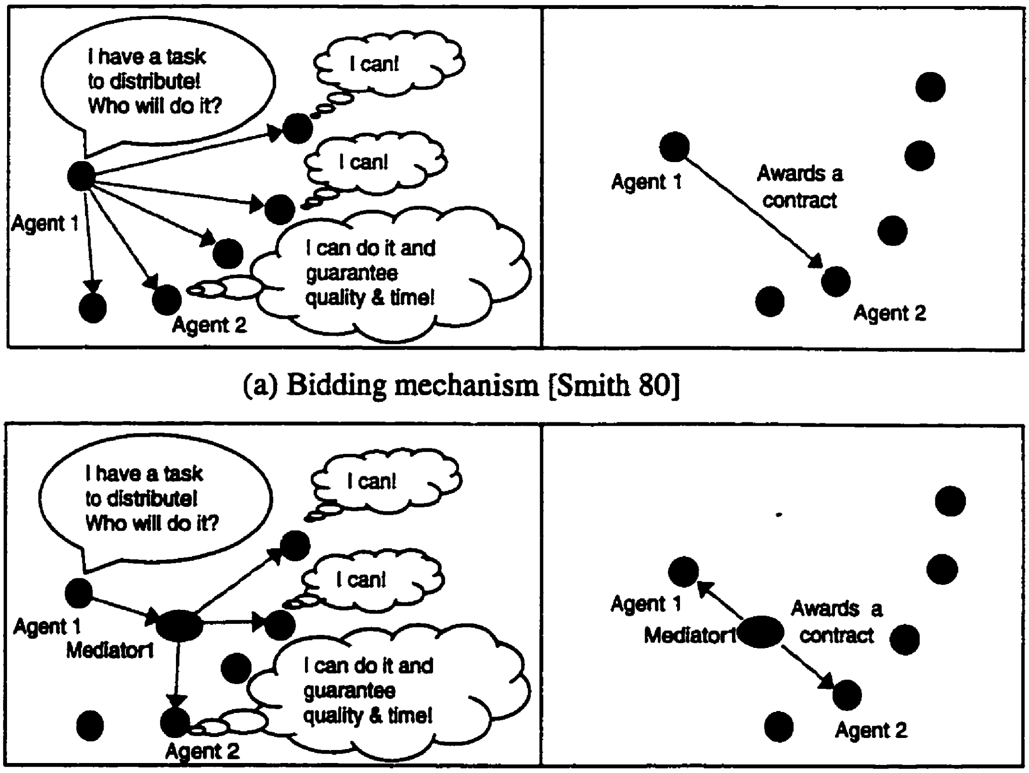
The *agent-based distributed modeling approach* focuses on organizing a distributed system using an aggregation of multiple and locally autonomous entities, called *agents*, thus allowing a cooperative approach to global decision-making [Moulin 96].

There are two major issues to be considered in attempting to achieve the collaboration among agents in distributed system [Shaw 87]: (1) an effective task assignment scheme among agents to ensure that all the resources can be efficiently utilized, and (2) a coordination scheme exercised among the agents carrying out manufacturing tasks cooperatively.

The bidding mechanism based on the contract net protocol can achieve the two above functions and thus became most commonly used negotiation mechanism among agents. In the *contract net protocol,* the agent that needs to solve the problem broadcasts a call-for-bids message to all the potential contractor agents, waits for replying messages for a period of time, and then awards the contract to the agent that can provide the best offer according to the contract selection criteria [Smith 80]. The contractor agent can also distribute tasks to other agents continuously if it is necessary. It forms a so-called contract-net structure, as shown in Figure 2.6 (a).

Mutual adjustment and direct supervision are two fundamental coordination schemes in a multi-agent system. In mutual adjustment scheme, agents share information and resources to achieve a common goal, adjusting their behavior according to the behavior of the other agents. In direct supervision scheme, one agent has some degree of control over others and can control information, resources, and behavior [O'Hare 96].

The mediation mechanism is used in a multi-agent system to coordinate the activities of agents [Maturana 96]. In mediation mechanism, a mediator facilitates or brokers mutual adjustment between agents and may also use direct supervision [O'Hare 96]. For example, if the mediation mechanism is combined with the bidding mechanism, shown in Figure 2.6 (b), the mediator can help the contract be signed quickly in the contract net



(a) Bidding mechanism [Smith 80]

(b) Mediation mechanism

Figure 2.6: Biding mechanism and mediation mechanism

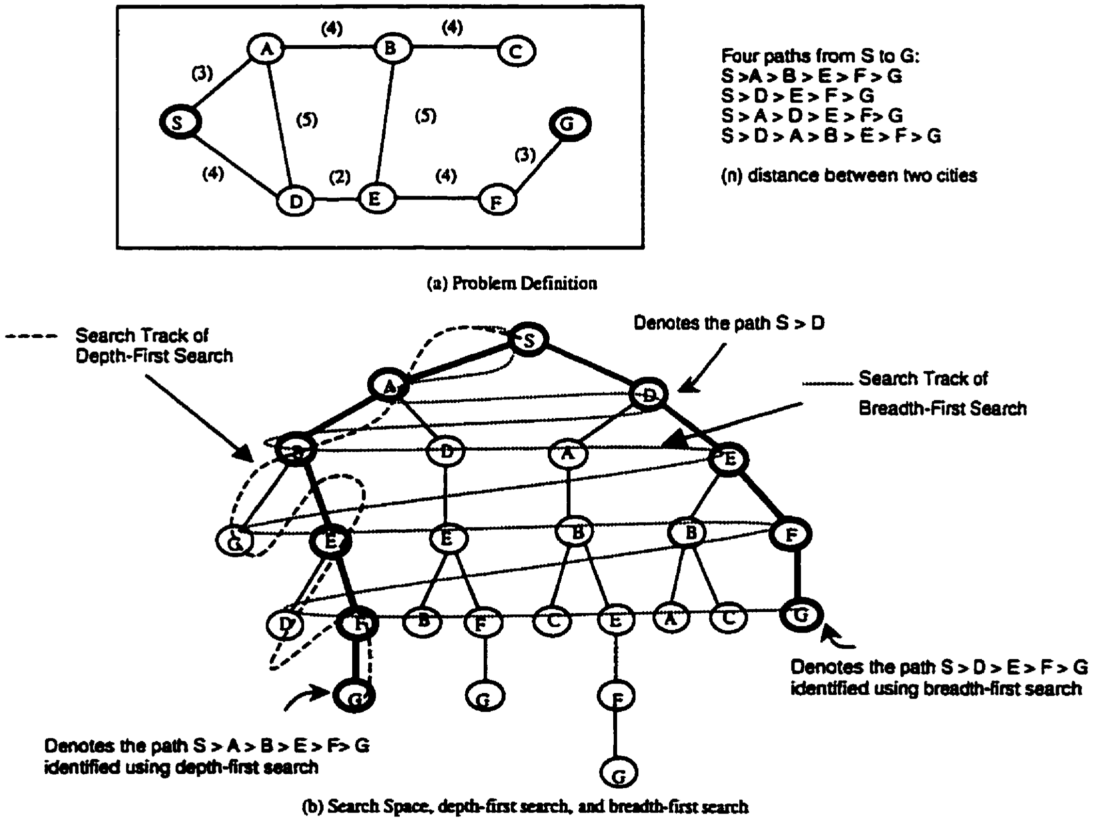
protocol, since the mediator can quickly find out the *relevant* agents without spending time to wait for the messages from the *unrelevant* agents.

An example of earliest work on the mediation mechanism is the MetaMorph I architecture developed by Maturana and Norrie. This architecture takes its name from its ability to change its activities to adapt dynamically to tasks that emerge in the manufacturing system. In this architecture, mediator agents facilitate the coordination of the resource agents that are associated with the physical manufacturing system [Maturana 96].

2.3.3 Heuristic Search Techniques

Artificial intelligence (AI) is the design of computer programs that behave intelligently [Dean 95]. A well-developed area of AI is the state space search.

The state space search is used to identify a solution from a start state through a sequence of intermediate states. Each state represents the status of the real world at a certain problem solving stage [Xue 99d]. In the example shown in Figure 2.7 (a), the search objective is to find the path from the start city *S* to the goal city *G*, thus the start state and the goal state represent locations of *S* and *G* respectively [Winston 92]. A search process is conducted by exploring the possible state change through developing a tree of nodes, in which each node represents a problem state and each arc represents a relationship between the two states represented by the nodes it connects. Figure 2.7 (b) shows a complete search space for solving the problem defined in Figure 2.7 (a). Four possible paths from *S* to *G* exist in the search tree. The path *S*→*D*→*E*→*F*→*G* is the solution with the minimum total traveling distance.


(4)
(4)
(3)
(5)
(5)
(3)
(4)
(2)
(4)
S
A
B
C
D
E
F
G
Four paths from S to G:
S>A>B>E>F>G
S>D>E>F>G
S>A>D>E>F>G
S>D>A>B>E>F>G
(n) distance between two cities
Search Track of Depth-First Search
Denotes the path S > D
Search Track of Breadth-First Search
Denotes the path S > D > E > F > G identified using breadth-first search
Denotes the path S > A > B > E > F> G identified using depth-first search

(a) Problem Definition

(b) Search Space, depth-first search, and breadth-first search

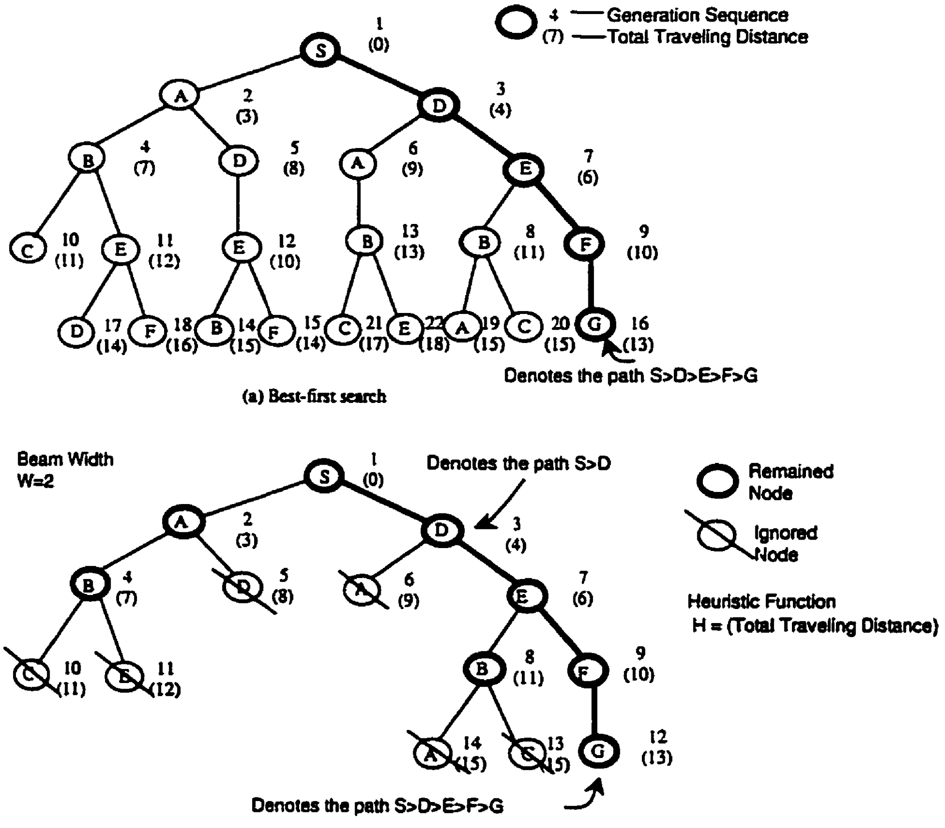
Figure 2.7: Search problem, search space, depth-first search, and breadth-first search

Depth-first search and breadth-first search are two commonly used search strategies for identifying feasible solutions without heuristic functions. The solutions obtained using these two search strategies for solving the problem defined in Figure 2.7 (a) are shown in Figure 2.7 (b). Depth-first search and breadth-first search can identify a feasible solution, but not the optimal solution.

Therefore, it is necessary to introduce heuristic search for improving search quality. *Heuristic function* is a rule or method that provides guidance for finding the optimal solution in a heuristic search process [Dean 95]. For instance, in the problem of finding the optimal path from S to G shown in Figure 2.7 (a), the heuristic function is the total travelling distance.

Best-first search and beam search are two kinds of heuristic search techniques. In ***best-first search*** algorithm, each time the best node is selected for generating its sub-nodes. This process is carried out continuously until the selected node is the goal node. The path found by best-first search is a global optimum, because best-first search always moves forward from the node that has the best measurement of heuristic function. The best-first search algorithm identifies the optimal solution in a way shown in Figure 2.8 (a).

Beam search is a modified best-first search to improve search efficiency. In this method, when the number of nodes at a certain level of the search tree exceeds a pre-defined number, the nodes with poor evaluation measures should be cut off from the search tree. Figure 2.8 (b) illustrates how beam search handles the search problem shown in Figure 2.7 (a). In this example, the beam width is selected as 2. The number of generated nodes has been reduced to 14 to achieve the result. The beam search risks falling into a local optimum caused by cutting off the nodes that lead to the global optimum. However, due to its efficiency and comparatively good quality, the beam search is often adopted for solving real life engineering problems [Yadav 99]. Therefore, in this research, the beam search is used to identify the optimal production schedule.


4 — Generation Sequence
(7) — Total Traveling Distance
Denotes the path S>D>E>F>G
Beam Width
W=2
Denotes the path S>D
Remained Node
Ignored Node
Heuristic Function
H = (Total Traveling Distance)
Denotes the path S>D>E>F>G
S
A
D
B
E
F
G
C

(a) Best-first search

(b) Beam search

Figure 2.8: Best-first search and beam search

2.3.4 An Object-Oriented Programming Language: Visualworks/Smalltalk

The *object-oriented programming* (OOP) is a method of software implementation in which program are organized as cooperative collections of objects, each of which represents an instance of some class, and classes are united via inheritance relationships [Booch 94].

The four fundamental concepts in object-oriented programming are discussed below:

- *Abstraction:*

  Abstraction has been defined as the essential characteristic of an object that distinguishes itself from other kinds of objects, thus providing crisply defined boundaries. Abstraction is implemented by introducing classes and instances in

OOP. For example, concept of gear can be represented by a class, while specific gears can be described by instances.

- *Encapsulation:*

  Encapsulation is the property that the inside (implementation) details of an object are hidden from the outside world. This is a very useful property in developing large systems due to its high modularity characteristic. If object A wants to access object B, object A has to send a message to B and ask object B to do so. This mechanism is called message passing.

- *Inheritance:*

  In the OOP paradigm, classes are organized in a hierarchical data structure. A sub-class is able to inherit all the characteristics of its super-class automatically. For instance, if class *Car* is a super-class for classes *Toyota* and *Taurus*, all the characteristics of *Car* are inherited by *Toyota* and *Taurus.*

- *Polymorphism:*

  Polymorphism is a mechanism to respond to messages with the same name using different functions. For instance, the function *display* can be defined in each of the classes of *Circle*, *Rectangle*, and *Triangle*. If the *display* method is sent to an object, a circle, a rectangle or a triangle is displayed depending on the class type of the object.

The *Smalltalk language and programming environment* were developed at Xerox Palo Alto Research Center (PARC) in 1970's and early 1980's. Its first commercial version, Smalltalk-80, is one of the pioneer OOP languages [Goldberg 80]. Many concepts of Smalltalk-80 have been adopted by subsequent OOP languages such as C++ and Java [Hopkins 95; Sharp 97].

Generally, Smalltalk language has the following major features:

(1) *Class* and *instance*: Classes are generic abstractions of the physical objects with similar characteristics, attributes and behaviors. An instance is created using a class as the template.

(2) *Class variables* and *instance variables*: A class has two kinds of variables: *class variables* and *instance variables*. The class variables are shared by all instances of the class, i.e., the value of a class variable is the same for all the instances of that class. The values of instance variables are specific to a particular instance.

(3) *Class method* and *instance method:* The functions of classes are defined as class methods and instance methods. Classes respond to class methods (messages), while instances respond to instance methods (messages). All the variables and methods defined in a class are inherited by its sub-classes.

In this research, the latest version of Smalltalk, called VisualWorks 2.5, has been used for implementing the intelligent production scheduling system. VisualWorks 2.5 provides a user-friendly interface environment and a large library of classes. Several hundreds of classes have been provided for modeling the components of this system. New classes can be defined in the same environment by declaring them as subclasses of the existing classes. Users can also modify the classes defined by the system.

It has been proved that Visualworks/Smalltalk is an excellent tool for developing research oriented software prototype systems, even though this language has weaknesses, such as slow computation speed, large memory requirement, and high price of software package [Xue 92].

Chapter 3

An Intelligent Production Scheduling System

In this chapter, the architecture of an intelligent production scheduling system is proposed to achieve the system functional requirements. In Section 3.1, the functional requirements are identified for the intelligent production scheduling system in an integrated design and manufacturing environment. Section 3.2 gives an overview of each module in this system, including product modeling module, resource management module, and scheduling module.

3.1 System Functional Requirements

In this section, the functional requirements are identified for the intelligent production scheduling in an integrated design and manufacturing environment.

3.1.1 A Research Project for Gienow Building Products Ltd.

Gienow Building Products Ltd. is a manufacturing company for producing building products of windows and doors. The computer-based production management system at this building product manufacturing company is primarily used to partially assist the various activities in the product realization process, including product ordering, design modeling, resource allocation, product cost estimation, and so on. To further improve the product development efficiency, a university-industry project was launched to develop an integrated design and manufacturing system in 1997. This project focuses on the following two issues: (1) to improve the efficiency of product modeling and design, and

(2) to reduce the production time and cost by introducing advanced production process planning and scheduling methods [Xue 99b; Xue 99c].

Therefore, the two different systems, a feature-based intelligent design system and an intelligent production scheduling system, are integrated, as shown in Figure 3.1. The feature-based intelligent design system has been developed before the work introduced in this thesis [Yadav 98; Xue 99b]. The feature-based intelligent design system consists of the three modules:

(1) A feature-based product modeling module is used to model the products in an efficient manner. In this module, the standard product libraries are described as class features. Actual products ordered by customers are modeled by instance features. Instance features are generated using the class features as their templates. The generated instance features can be modified to satisfy the requirements from the customers.

(2) A knowledge-based inference module is used to generate products, which are described by instance features, automatically using design knowledge. Design knowledge is modeled by collections of rules, called rule-bases. During the knowledge-based design, only partial rule-bases and instance features are considered to improve the inference efficiency.

(3) A product geometric modeling module is used to associate the symbolic product model, which is represented by instance features, with the 3D solid geometric model to show the design results. In this module, a geometry exchange module is used to keep the consistency of the symbolic model and the 3D solid geometric model. 3D Studio MAX [Michael 96], a CAD package, has been used for modeling the 3D geometry of the products.

The work of this thesis focuses on developing the intelligent production scheduling system that is associated with the feature-based intelligent design system introduced above.

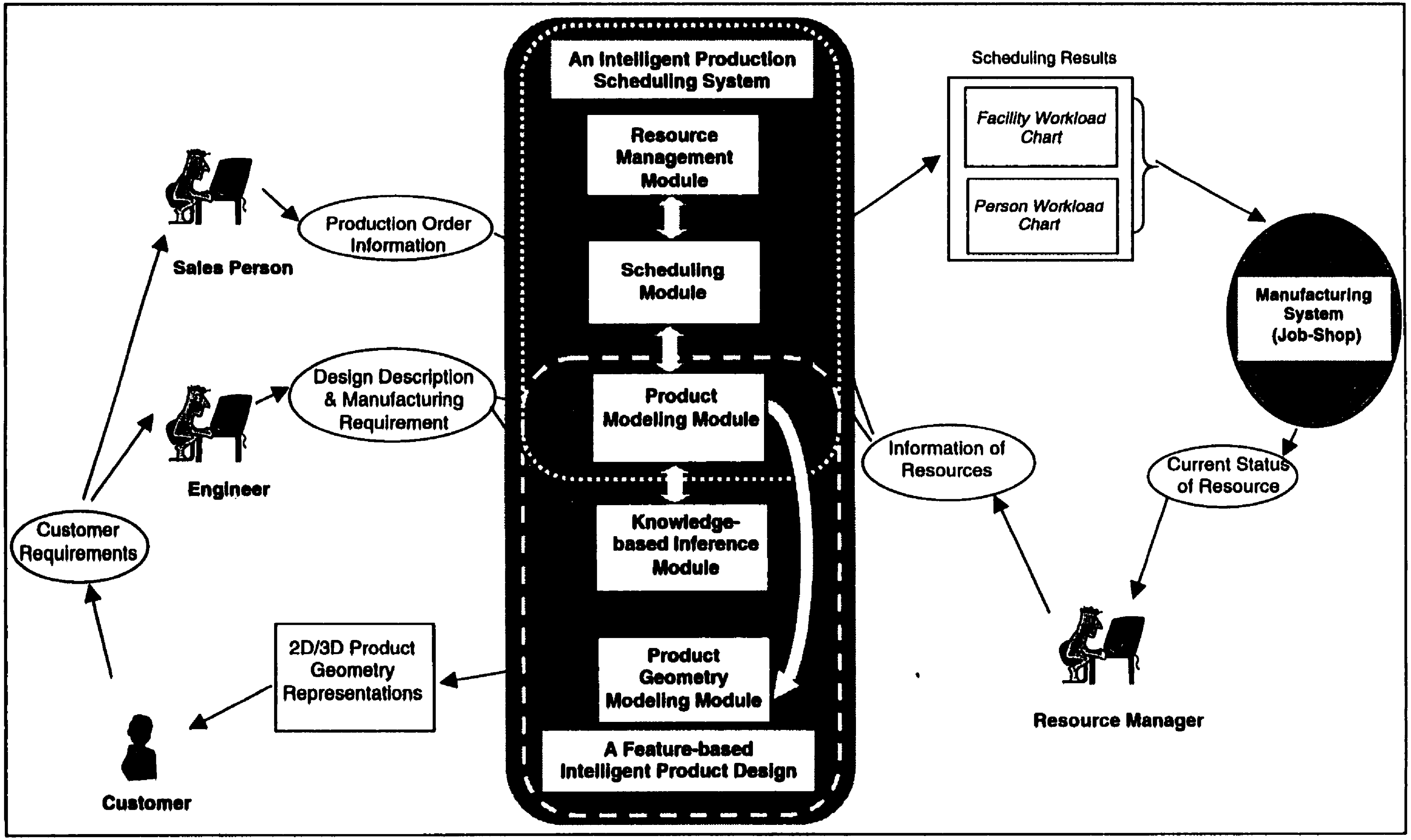

An Intelligent Production Scheduling System
Resource Management Module
Scheduling Module
Product Modeling Module
Knowledge-based Inference Module
Product Geometry Modeling Module
A Feature-based Intelligent Product Design
Sales Person
Production Order Information
Engineer
Design Description & Manufacturing Requirement
Customer Requirements
Customer
2D/3D Product Geometry Representations
Scheduling Results
Facility Workload Chart
Person Workload Chart
Manufacturing System (Job-Shop)
Current Status of Resource
Information of Resources
Resource Manager

Figure 3.1: An Integrated design and manufacturing environment

3.1.2 Functional Requirements for the Intelligent Production Scheduling System

The intelligent scheduling system was proposed to address a generic production scheduling problems that can be described in the following aspects:

(1) Resource Descriptions

A generic manufacturing job-shop comprises a variety of manufacturing facilities and a number of persons. The manufacturing cells usually include machining facilities, assembly facilities, packing facilities, painting facilities, and so on. Each facility comprises one or several machine tools, an input buffer and an output buffer, which are interconnected through a material handing system.

Both facilities and persons have several types of constraints, such as function constraints and time constraints. The function constraints indicate a facility or person can be allocated to carry out one or several types of manufacturing processes, or a person is able to operate one or several manufacturing facilities. The time constraints specify the time periods in which a facility or a person is available to conduct the production tasks.

(2) Order and Product Descriptions

A production order is created according to the customer requirements, such as the required product, amount of the product, and due date. The required product can be modeled to satisfy the customer requirements using the feature-based modeling approach. A product model can be generally described by a collection of features. The features for representing a product should be organized in a hierarchical data structure, since a product is usually composed of element features and some element features may also be made up of sub-element features.

The manufacturing requirements to produce each feature should be described by a number of tasks. Each of these tasks should be associated with a manufacturing process. Each process specifies the resource requirements and time period to conduct the task, including type, required facility, and duration time.

The duration time includes the loading, unloading, tool changeover and set-up time (both tool and workpiece) along with processing time (i.e., the time between the part being picked and returned to the local buffer). The transportation time is small compared to the operation time, thus is negligible.

The production tasks can be linked as a graph representing the precedence constraints to accomplish these tasks.

(3) *Schedule Descriptions*

In the generic production scheduling problem, schedule is originally generated by predictive scheduling and revised for responding to production demand changes and manufacturing resource changes by reactive scheduling. Upon receiving the customer requirements, the system generates a production order to manufacture a product. Then, the system identifies the optimal production schedule.

The system should conduct reactive scheduling by changing only partial original schedule for responding to the changes such as canceling orders, inserting orders that cannot be scheduled using predictive scheduling due to the urgent due date requirement, facility breakdowns, and personnel absence.

The constraints in design and manufacturing aspects should be considered in both predictive scheduling and reactive scheduling. Design constraints should be modeled based on feature-based product representation scheme, and manufacturing constraints should be described as available resources including facilities and persons.

To address the generic production scheduling problems described above and emphasize the research motivations introduced in Section 1.2, the system functional requirements are identified and summarized as follows:

1. Generating the production order according to the customer requirements, modeling the manufacturing requirements for the required product, and associating these requirements with the design descriptions and constraints.

2. Modeling manufacturing resources using an agent-based distributed approach for distributed implementation of intelligent scheduling.
3. Conducting predictive scheduling to identify the optimal schedule considering design and manufacturing constraints.
4. Conducting reactive scheduling to revise original schedule for responding to production demand changes and resource changes.
5. Developing a user-graphic-interface (UGI) environment for organizing and maintaining resource and product databases and displaying the scheduling results in the form of Gantt charts.

3.2 System Architecture

To achieve the system functional requirements specified in the previous section, an intelligent production scheduling system has been developed in this research. Architecture of this system is illustrated in Figure 3.2. The system consists of three modules: *product modeling module, resource management module*, and *scheduling module*.

In this system, the production orders are generated according to the customer requirements in the scheduling module. The manufacturing requirements for the required products, including tasks and precedence constraints for accomplishing these tasks, are described using the feature-based product representation scheme in the product modeling module. In the resource management module, manufacturing resources, including facilities and persons, are modeled as an agent-based distributed system in order to realize the distributed implementation of intelligent scheduling. The optimal production schedule can be identified using the artificial intelligent approach when a production order is created. If changes of production demands and resources happen during the production process, the system is able to revise the original schedule to respond to these changes.

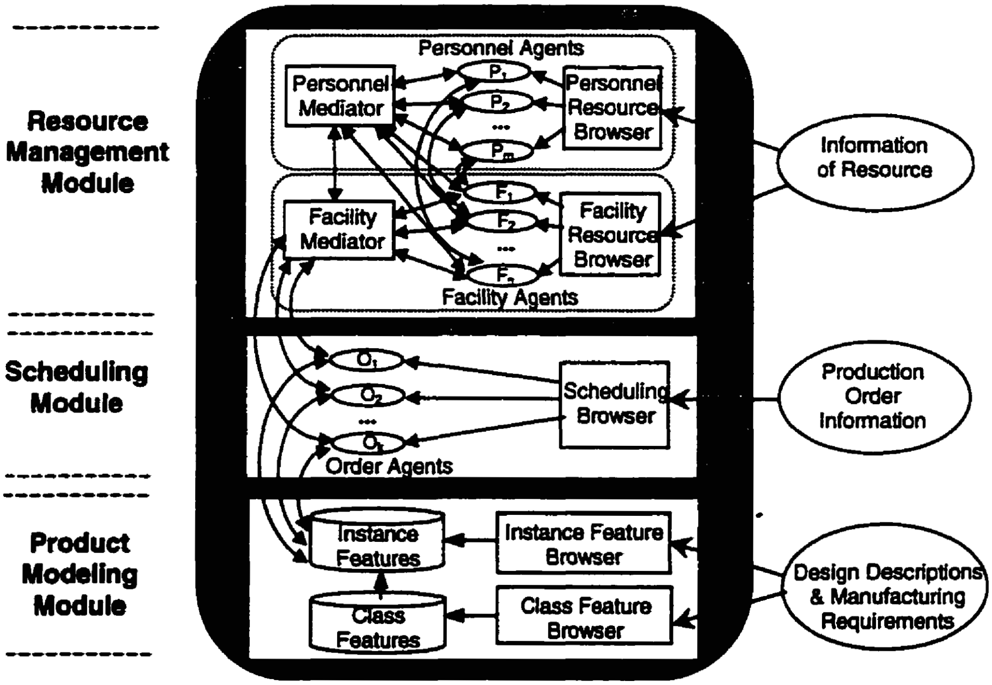


Figure 3.2: Architecture of the intelligent production scheduling system

3.2.1 Product Modeling Module

In the product modeling module, a product is modeled by its composing primitives, called features as described in Section 2.3.1. Features are defined at two different levels: class level and instance level, corresponding to standard product library and special product data respectively.

The class features are described using the class feature browser. The descriptions of feature consist of both qualitative data and quantitative data and the relations among these data, including attributes (parameters), attribute relations, element features (constituent building blocks), feature relations, etc.

In the instance feature browser, instance features are generated to represent product data using class features as templates. All the class feature descriptions are inherited by the corresponding instance features automatically. The descriptions in the instance features can be modified during product modeling process. Since the data in instance features are associated by their relations, any change of partial product data descriptions can be propagated to other parts automatically using these relations.

The manufacturing requirements to produce each feature are also defined as part of feature descriptions at the class level and instance level. These requirements are described by a number of tasks. Each of these tasks is associated with a manufacturing process. Each process specifies the resource requirements and duration time to conduct this process. These tasks are linked as a graph representing the precedence constraints to accomplish these tasks.

3.2.2 Resource Management Module

Manufacturing resources are modeled using a distributed approach in the resource management module, as shown in Figure 3.2. Each resource is represented as an agent, which is defined by different types of constraints, such as time constraints and function constraints. In this module, the resource agents are coordinated by two mediators, facility resource mediator and personnel resource mediator. The mediator is a distributed decision-making support agent for coordinating the activities of a multi-agent system.

In this system, two types of resources, facilities and persons, are considered. The facility resource mediator is used to coordinate all the facility resource agents, and the personnel mediator to coordinate all the personnel agents. The facility agents and personnel agents can communicate and negotiate with each other in the scheduling process. Thus, two kinds of resource browsers, the facility resource browser and the personnel resource browser, were developed for managing and modeling the facility agents and personnel agents, respectively.

3.2.3 Scheduling Module

The production scheduling module aims at conducting predictive scheduling function and reactive scheduling function.

In this module, the scheduling browser is used as the user interface for conducting predictive scheduling function and reactive scheduling function. When receiving the customer requirements, the system generates a production order to manufacture a required product. The optimal production schedule is identified based on heuristic search and agent-based negotiation among the order agent, the resource mediators, and the resource agents. The heuristic search is guided by the design and manufacturing constraints. The bidding and mediator mechanisms are employed in the negotiation among agents.

If changes of production demands and resources happen, such as canceling old orders, inserting urgent orders, facility breakdowns, and personnel absence, the user can enter the information about these changes to the system through the scheduling browser and resource browsers. Then, the system conducts reactive scheduling to revise the original schedule for responding to these changes. Two intelligent reactive scheduling mechanisms in this system are used to revise original schedules for responding to production demand changes and resource changes, respectively. The intelligent reactive scheduling is conducted based on the match-up rescheduling approach and the agent-based negotiation in the resource management module.

Chapter 4
Predictive Scheduling

This chapter focuses on the predictive scheduling aspect of the intelligent production scheduling system. In Section 4.2, manufacturing requirements, including tasks and precedence constraints for accomplishing these tasks, are described using the feature-based product representation scheme. Section 4.3 introduces the distributed resource management module, in which manufacturing resources including facilities and persons are defined as agents that are coordinated by two mediators: facility mediator and personnel mediator. An intelligent predictive scheduling mechanism is proposed in Section 4.4. The optimal production process sequences and their timing parameters and resource allocation for producing the products ordered by customers are identified based upon heuristic search and agent-based negotiation.

4.1 Introduction

In this research, an intelligent predictive scheduling mechanism is developed to identify an optimal production schedule for producing an ordered product considering the corresponding manufacturing requirements, design constraints, and manufacturing constraints. The optimal production schedule includes optimal sequences of production processes and timing parameters and resource allocation for each process.

In the intelligent production scheduling system, features are used as product primitives for product modeling. Design constraints to be considered in production scheduling are formed during design process. The relations among composing features of the product to be scheduled are viewed as the design constraints. Manufacturing requirements to produce the product, including tasks and precedence constraints for accomplishing these tasks, are modeled as part of the feature descriptions. Manufacturing resource are the manufacturing constraints to be considered in production scheduling. The optimal schedule for producing the products ordered by customers is identified using artificial intelligence techniques in the multi-agent environment.

4.2 Feature-based Product Modeling and Manufacturing Requirement Representation

In this research, the product primitives are modeled as features. The concept of feature was originally used for representing the component geometry to be produced by a certain manufacturing process [Shah 95]. Typical geometric features are holes, slots, etc. produced by manufacturing processes of drilling, milling, and so on. Production process planning can be easily carried out, if the composing manufacturing features of the component can be identified. In this research, the concept of feature was extended to model the design primitives such as gears and shafts to satisfy design functional requirements. This concept of feature was firstly proposed in Xue's previous research [Xue 93; Xue 94].

In this system, features are described at two different levels, class level and instance level, corresponding to generic product libraries and specific product data respectively. Instance features are generated using the class features as their templates. This mechanism was implemented using object-oriented programming approach [Yadav 98; Xue 99c].

The manufacturing requirements to produce each feature are also defined at the class level and instance level. These requirements are described by a number of production tasks, which correspond to the manufacturing processes, and precedence constraints for accomplishing these tasks.

4.2.1 Class Features and Instance Features

In this system, features are described at two different levels, class level and instance level, corresponding to generic product libraries and specific product data respectively, as shown in Figure 4.1 and Figure 4.2 [Xue 99a].

(1) Class Features

Class features are organized in a hierarchical data structure. A new class feature is defined using an existing class feature as its super-class. The new class feature can inherit all the descriptions in its super-class feature automatically. The top level class feature is a built-in class feature called *Feature*. Examples of class feature definitions are shown in Figure 4.1.

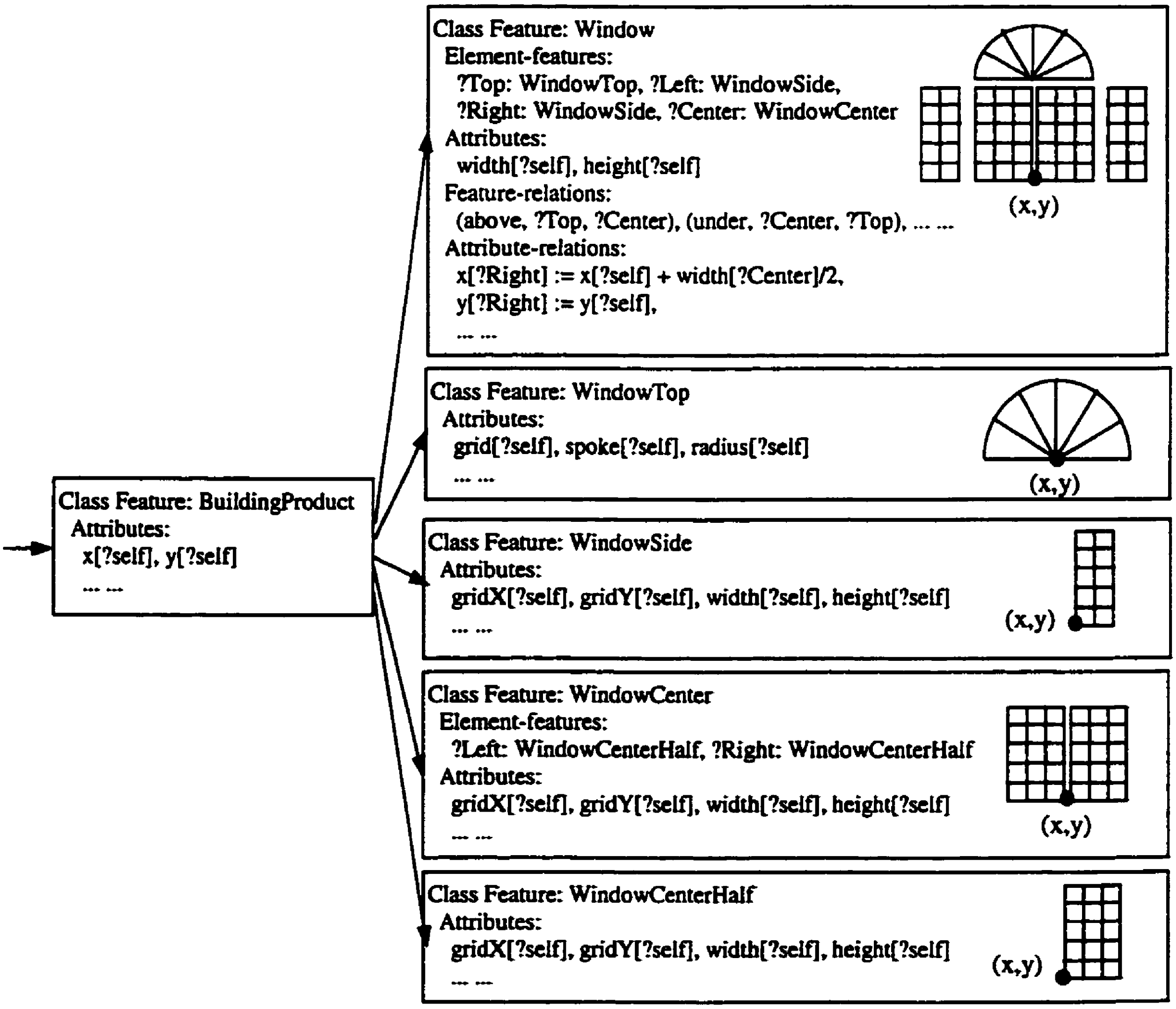

Class Feature: Window
Element-features:
?Top: WindowTop, ?Left: WindowSide,
?Right: WindowSide, ?Center: WindowCenter
Attributes:
width[?self], height[?self]
Feature-relations:
(above, 7Top, ?Center), (under, ?Center, 7Top), ...
Attribute-relations:
x[?Right] := x[?self] + width[?Center]/2,
y[?Right] := y[?self],
... ...
(x,y)
Class Feature: WindowTop
Attributes:
grid[?self], spoke[?self], radius[?self]
... ...
(x,y)
Class Feature: BuildingProduct
Attributes:
x[?self], y[?self]
... ...
Class Feature: WindowSide
Attributes:
gridX[?self], gridY[?self], width[?self], height[?self]
... ...
(x,y)
Class Feature: WindowCenter
Element-features:
?Left: WindowCenterHalf, ?Right: WindowCenterHalf
Attributes:
gridX[?self], gridY[?self], width[?self], height[?self]
... ...
(x,y)
Class Feature: WindowCenterHalf
Attributes:
gridX[?self], gridY[?self], width[?self], height[?self]
... ...
(x,y)

Figure 4.1: Representation of class features

A class feature is described in the following aspects: element-features, attributes, qualitative relations among features, and quantitative relations among attributes, etc.

An element feature is a feature that composes the feature being defined. For instance, the *Window* class feature, shown in Figure 4.1, is composed of four element features. An element feature is defined by a variable and its class feature type. A variable is described by a string starting with "?". In the class feature *Window*, the four element features are associated with four variables: *?Top*, *?Left*, *?Right*, and *?Center*. When a class feature is used to generate an instance feature, the element features should also be created according to their class feature types. The variables, representing the element features, are used in other parts of the class feature definition. For instance, the four variables are used for defining feature relations and attribute relations in *Window*. The feature itself is associate with a special variable *?self*.

An attribute is a piece of quantitative description of the feature. An attribute is described by an attribute name and an attribute value. Attributes are used in the form of *attribute[feature]* in other parts of the feature definition. For instance, *width[?Center]* in Figure 4.1 represents the attribute width of the feature *?Center*.

Qualitative relations among features are described by predicates. A predicate takes the form of $(x_1, x_2, ..., x_n)$, where $x_1, x_2, ..., x_n$ are terms of this predicate represented by strings, integers, floats, variables, and attributes. A predicate without variable terms is called a fact.

A quantitative relation among attributes is defined by a function with a number of input attributes and one output attribute. An attribute relation takes the form of

<attribute name> := <expression>

The element feature variables and attributes are allowed in the expression. Examples of attribute relations are shown in Figure 4.1.

In addition to the four aspects introduced earlier, class features can also be defined in many other aspects, including constraints, 2D and 3D geometric descriptions, etc.

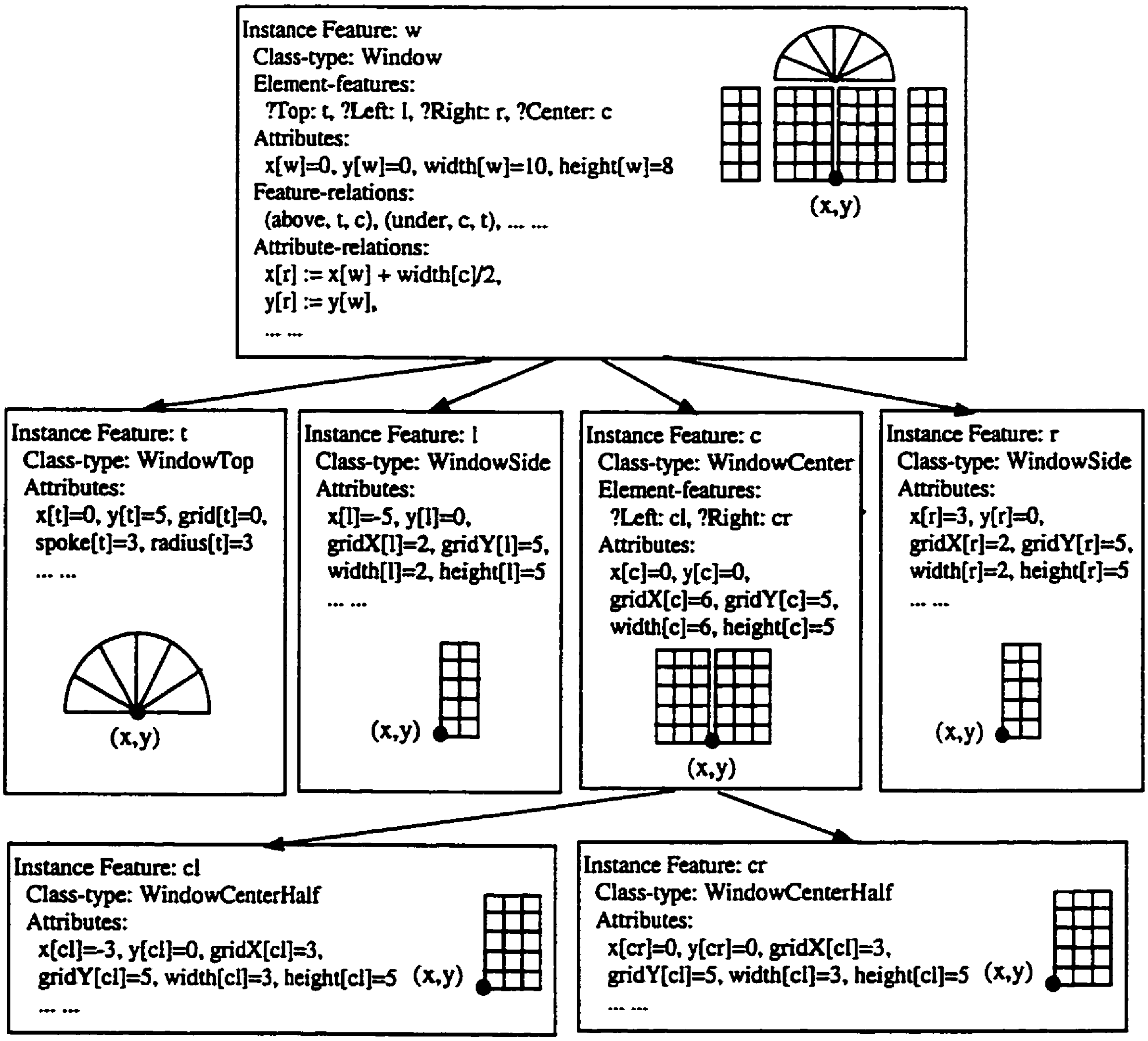


Figure 4.2: Representation of instance features

[Yadav, 98]. The manufacturing requirements for producing each feature can also be modeled as part of the feature definition (to be introduced in Section 4.2.3)

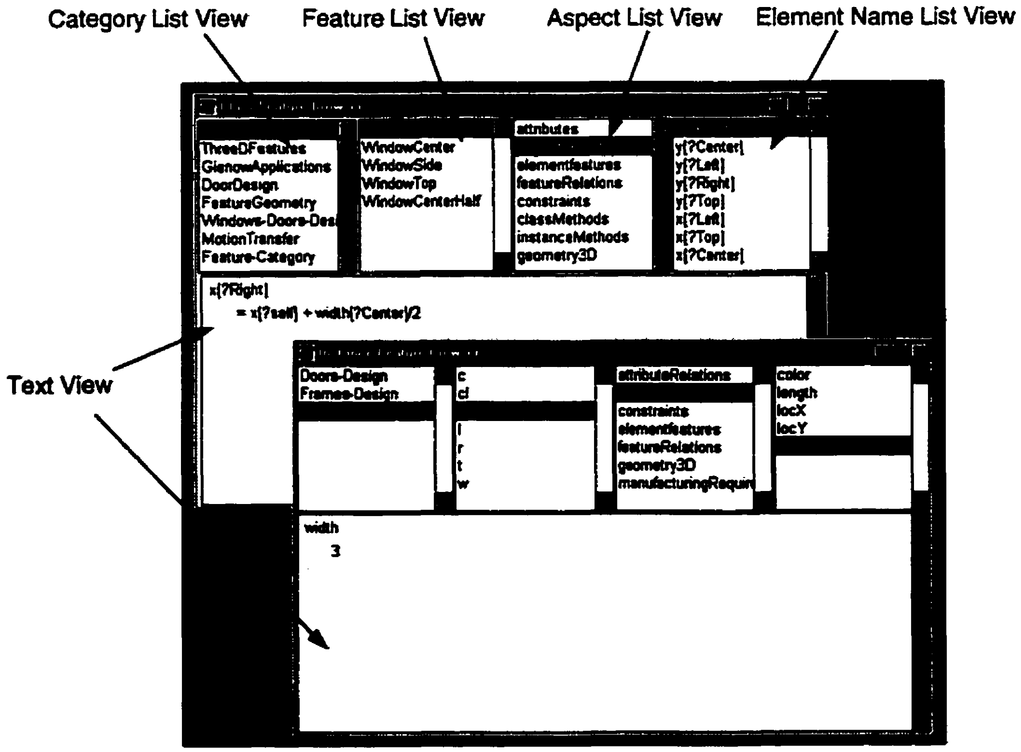
(2) Instance Features

Instance features are generated from class features for modeling products ordered by customers. When a class feature is selected for generating an instance feature, the element features defined in this class feature should also be generated as instance features. All the descriptions in the class feature and its super-class features are inherited by the generated instance feature automatically. Figure 4.2 shows the instance features that are generated from the class features shown in Figure 4.1 for representing a window product.

In an instance feature, the variables in its class feature definition, representing element features, are replaced by the names of the actually created element instance features. For example, in the instance feature *w* shown in Figure 4.2, the four element feature variables are replaced by the four instance features. Descriptions of instance features can be modified, added, and deleted [Yadav 98]

(3) Class Feature Browser and Instance Feature Browser

In this system, *Class Feature Browser* and *Instance Feature Browser* are user interfaces for defining class features, generating and modifying instance features. A snapshot of these browsers is shown in Figure 4.3. Each browser consists of four list views and one text view as illustrated in this figure.



```
x[?Right]
    = x[?self] + width[?Center]/2
```

```
width
    3
```

Figure 4.3: A snapshot of class feature browser and instance feature browser

In the class feature browser, all the class features are stored in different categories due to the large number of class features. The categories play no role in the product modeling process except for helping to find out the required class features easily. The categories are listed in the category list view. By selecting one of the categories, the features in that category are shown in the feature list view. By selecting a feature in the feature list view, a list of aspects is shown in the aspect list view. The aspects include attributes, element-features, attribute-relations, feature-relations, geometry3D, constraints, manufacturing-requirements, etc. The aspect list is a built-in list that is common to all features. The element name list view shows the names of the feature elements in corresponding aspect. These element names are represented with different formats. The text view is used to edit and modify the feature definitions.

The instance feature browser also consists of four list views and one text view. Different products are generated in separate categories. Instance features can be added to or removed from a category using the feature list view. All the class feature descriptions are inherited by the corresponding instance features automatically. Instance features can be modified manually using this browser [Yadav 98].

4.2.2 Manufacturing Requirement Representation

In this system, the manufacturing requirements to produce each feature can be modeled as part of feature definition at the class level and the instance level.

As introduced in 4.2.1, in the class feature browser and the instance browser, among the aspects for describing each feature, the aspect *Manufacturing-Requirements* describes the manufacturing requirements that will result in the completion of production of the defined feature. These requirements are described by a number of production tasks, which correspond to the manufacturing processes. These tasks are linked as a graph representing the precedence constraints to accomplish these tasks. Each task is associated with a manufacturing process. A process specifies the resource requirements and time period to carry out this process. A production task is described using the following template in the text view:

<Task Name>
Ancestors: “<ancestor1 ancestor2>”
Descendents: “<descedent1 descedent2>”
Process: “Type: <type>, Facility: <type>, Duration: <time (min)>”

To describe a production task using the template, the user is required to replace the text within brackets <> with the actual task descriptions. Ancestors and descendents represent the precedence constraints to accomplish the defined task. If task A must be conducted before task B, A is called an ancestor task of B, and B is called a descendent task of A. The task is defined by one manufacturing process. *Process* specifies the process type in *Type: <type>*, the type of required facility in *Facility: <type>*, and the time period to carry out this operation in *Duration: <time (min)>*.

In this system, the feature-based representation of a product is organized in a hierarchical data structure, i.e. a product is composed of element features and some element feature may also be made up of sub-element features. Due to the hierarchical representation of a product, the individual task graphs for all features in the product form a tree data structure. It means a task in an instance feature is carried out in production only when all the tasks of this feature's element features have been completed. This representation is called the feature-based production task representation, as illustrated in the following example, as shown in Figure 4.4 [Sun 99a; Xue 99b].

As introduced in Section 4.2.1, the window product as shown in Figure 4.2 is described by 7 instance features, which are generated using the class features as their templates. The top-level instance feature, w, is generated from a class feature called *Window* that is composed of 4 element features: *?Top*, *?Left*, *?Right*, and *?Center*. When the class feature, *Window*, is used to generate its instance feature, w, the four element features should also be generated as instance features automatically. In this example, these 4 instance features are generated as *t*, *l*, *r*, and *c*, respectively. The instance feature, *c*, is further described by two element features: center left part, *cl*, and center right part, *cr*.

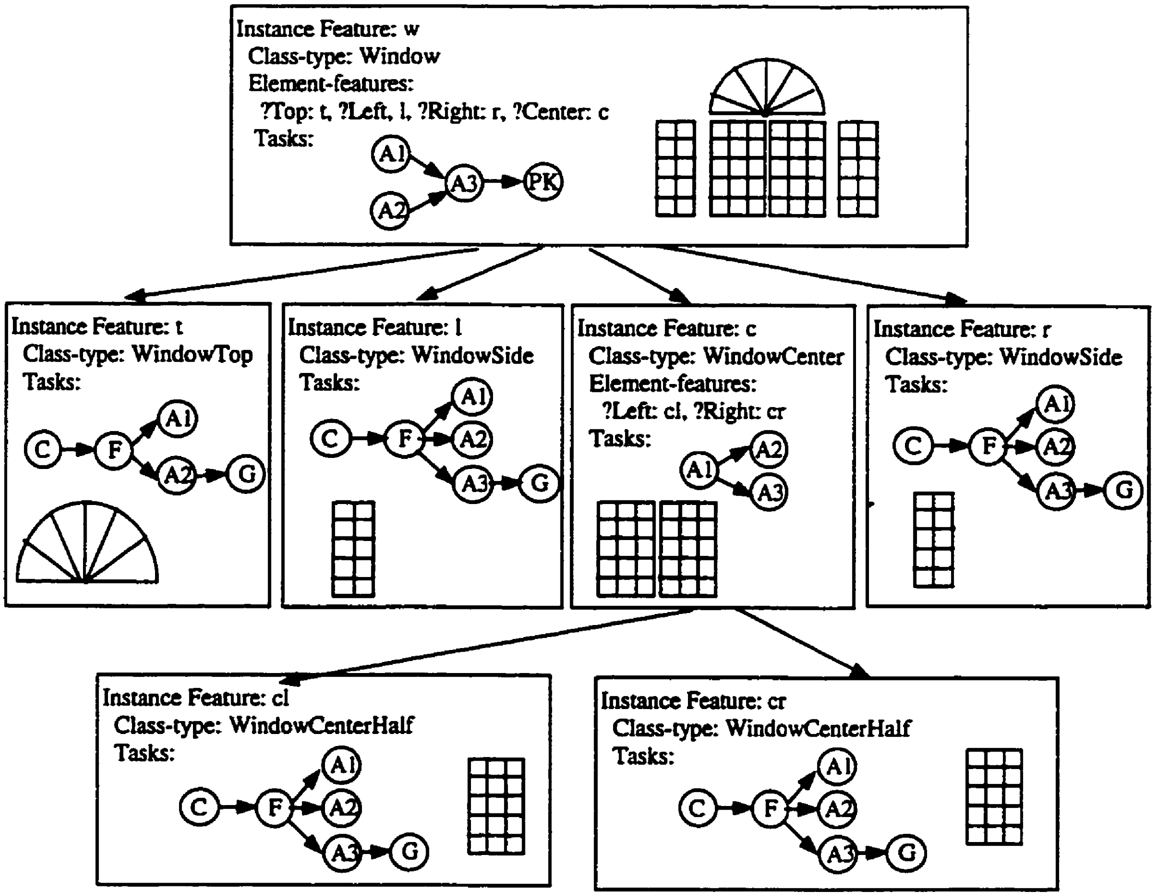


Figure 4.4: Feature-based manufacturing requirement representation

The manufacturing requirements for producing the product are modeled as part of the feature descriptions, as shown in Figure 4.4. In each feature, the manufacturing requirements to produce this feature are defined by tasks and each task is associated with a manufacturing process. The window production includes the following manufacturing processes: material cutting, frame construction, auxiliary assembly, glass installation, element assembly, and product packing. In this system as used for window production, these types of tasks are denoted by the following terms:

- *cutting* — material cutting process in which the frame bars are generated by cutting the raw materials.
- *framing* — frame construction process in which the frame of feature is constructed with the frame bars.

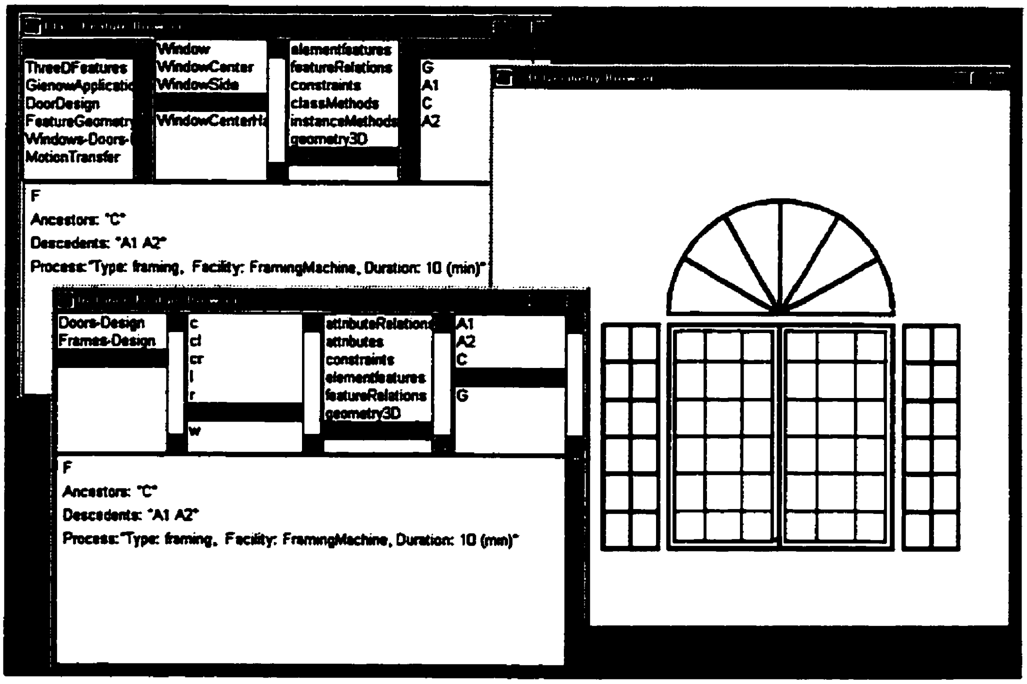
- *assemblyA* — auxiliary assembly process in which auxiliary parts, such as connection parts, glazing beads, casing bars, and locking device, etc., are installed on the frame of feature.
- *assemblyE* — element assembly process in which the composing element features are assembled together to form the corresponding feature.
- *glazing* — glass installation process in which the glasses are installed.
- *packing* — product packing process in which the product is packed for delivery to the customer.

Thus, the window manufacturing should include the following types of facilities:

- *cutting machine* — the facility which is used for material cutting process
- *framing machine* — the facility which is used for frame construction process
- *assembly unit* — the facility which can be used for auxiliary assembly process, auxiliary assembly process , or glazing process
- *packing unit* — the facility which is used for packing process

For instance, the top component, *t*, shown in Figure 4.4, can be produced by 5 tasks, including cutting, *C*, framing, *F*, auxiliary assembly, *A1* and *A2*, and glazing, *G*. These tasks are linked as a graph, representing the precedence constraints to accomplish these tasks. The task *F* has one ancestor task, *C*, and two descendant tasks, *A1* and *A2*. For instance, the framing task, *F*, and the auxiliary assembly task, *A1*, in the top component feature, *t*, shown in Figure 4.3, are defined as:

F
Ancestors: "C"
Descendants: "A1 A2"
Process: "Type: framing, Facility: FramingMachine, Duration: 10 (min)"
A1
Ancestors: "F"
Descendants: ""
Process: "Type: assemblyA, Facility: AssemblyUnit, Duration: 10 (min)"



F
Ancestors: "C"
Descedents: "A1 A2"
Process:"Type: framing, Facility: FramingMachine, Duration: 10 (min)"

F
Ancestors: "C"
Descedents: "A1 A2"
Process:"Type: framing, Facility: FramingMachine, Duration: 10 (min)"

Figure 4.5: A snapshot of manufacturing requirement definition

Figure 4.5 shows a snapshot of manufacturing requirement definition for this window product in the class feature browser and instance feature browser. The manufacturing requirements for producing it are listed in Table 4.1.

Table 4.1: Manufacturing requirements for an instance of class feature Window

Instance Features	Manufacturing Requirements					
	Tasks	Descriptions				
		Ancestors	Descendents	Types	Facilities	Duration (min)
w	w.A1		w.A3	assemblyA	AssemblyUnit	10
	w.A2		w.A3	assemblyA	AssemblyUnit	10
	w.A3	w.A1, wA2	w.PK	assemblyE	AssemblyUnit	15
	w.PK	w.A3		packing	PackingUnit	20
t	t.C		t.F	cutting	CuttingMachine	10
	t.F	t.C	t.A1, t.A2	framing	AssemblyUnit	10
	t.A1	t.F		assemblyA	AssemblyUnit	10
	t.A2	t.F	t.G	assemblyA	AssemblyUnit	10
	t.G	t.A2		glazing	AssemblyUnit	15

Table 4.1: Manufacturing requirements for an instance of class feature Window (continued)

Instance Features	**Manufacturing Requirements**					
	Tasks	**Descriptions**				
		Ancestors	**Descendants**	**Types**	**Facilities**	**Duration (min)**
r	r.C		r.F	cutting	Cutting Machine	10
	r.F	r.C	r.A1, r.A2, r.A3	framing	FramingMachine	10
	r.A1	r.F		assemblyA	AssemblyUnit	10
	r.A2	r.F		assemblyA	AssemblyUnit	10
	r.A3	r.F	r.G	assemblyA	AssemblyUnit	10
	r.G	r.A3		glazing	AssemblyUnit	10
l	l.C		l.F	cutting	Cutting Machine	10
	l.F	l.C	l.A1, l.A2, l.A3	framing	FramingMachine	10
	l.A1	l.F		assemblyA	AssemblyUnit	10
	LA2	l.F		assemblyA	AssemblyUnit	10
	l.A3	l.F	l.G	assemblyA	AssemblyUnit	10
	l.G	l.A3		Glazing	AssemblyUnit	10
c	c.A1		c.A2, c.A3	assemblyE	AssemblyUnit	15
	c.A2	c.A1		assemblyA	AssemblyUnit	10
	c.A3	c.A3		assemblyA	AssemblyUnit	10
cr	cr.C	cr.F	cr.F	cutting	CuttingUnit	10
	cr.F	cr.C	cr.A1, cr.A2, cr.A3	framing	FramingUnit	10
	cr.A1	cr.F		assemblyA	AssemblyUnit	10
	cr.A2	cr.F		assemblyA	AssemblyUint	10
	cr.A3	cr.F	cr.G	assemblyA	AssemblyUnit	10
	cr.G	cr.A3		glazing	AssemblyUnit	15
cl	cl.C		cl.F	cutting	CuttingMachine	10
	cl.F	cl.C	cl.A1, cl.A2, cl.A3	framing	FramingMachine	10
	cl.A1	cl.F		assemblyA	AssemblyUnit	10
	cl.A2	cl.F		assemblyA	AssemblyUnit	10
	cl.A3	cl.F	cl.G	assemblyA	AssemblyUnit	10
	cl.G	cl.A3		glazing	AssemblyUnit	15

4.3 Agent-Based Resource Organization

Manufacturing resources are the manufacturing constraints to be considered in production scheduling. In this system, the agent-based distributed approach is used to organize and manage manufacturing resources in the resource management module.

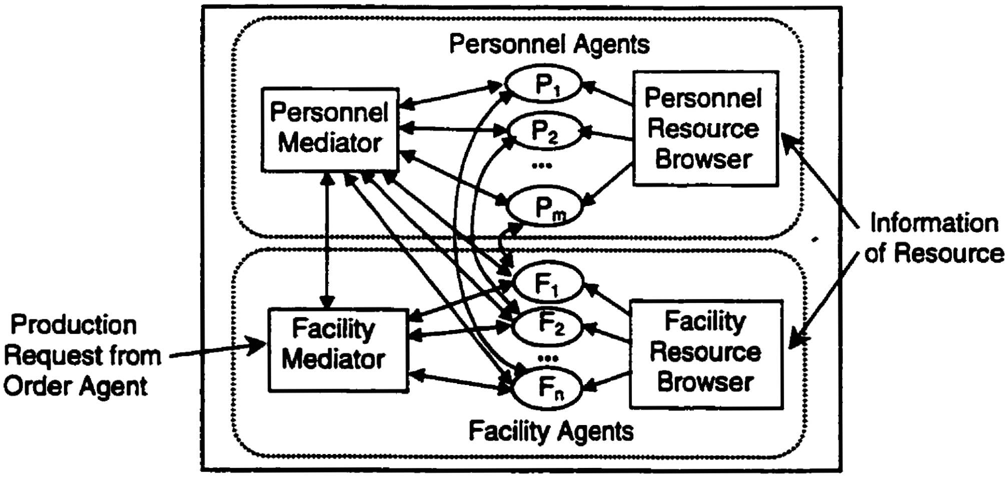


Figure 4.6: Architecture of the resource management module

Two types of resources, facilities and persons, are considered in this system. A facility denotes a resource unit that comprises one or several machines, space, buffer and auxiliary tools, which are grouped together for one or several particular manufacturing tasks. The persons work with the facilities to accomplish the production tasks.

Since each resource has its own characteristics and knowledge, such as type, functions and time constraints, and requires particular activities for making decisions during the scheduling process, these resources are described as agents, including facility agent and personnel agent. The activities of these two types of agents are coordinated by two mediators, facility mediator and personnel mediator, as shown in Figure 4.6. The facility agents and personnel agents communicate and negotiate with each other, through the two mediators, using the relations among these agents during the scheduling process.

During the intelligent predictive scheduling, allocation of resource and assignment of timing parameters for task to be scheduled are conducted based on agent-based negotiation mechanism in the resource management module. In the production scheduling process, the agent-based negotiation is conducted at two different levels: order-facility level and facility-person level. The order agent first communicates with the facility resource agents through the facility resource mediator, based upon the manufacturing requirements of tasks defined in the scheduled product. These facility resource agents further communicate with the person resource agents through the personnel resource mediator. Section 4.4 will discusses the details about the agent-based negotiation in the intelligent predictive scheduling.

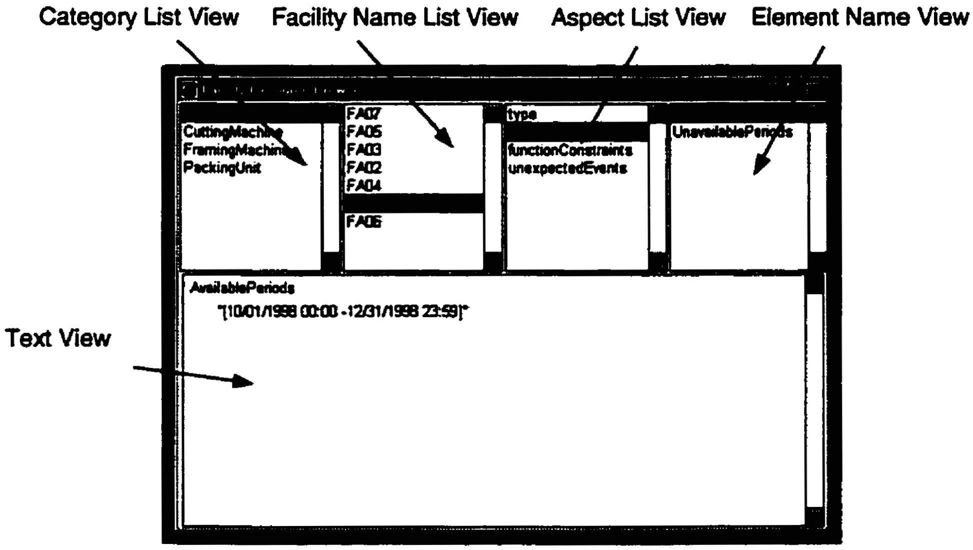
In the intelligent reactive scheduling, the system conducts the rescheduling based on agent-based negotiation among the resource agents and the mediators. The agent-based negotiation mechanisms for intelligent reactive scheduling will be introduced in Chapter 5: Reactive Scheduling.

The two kinds of resource browsers, the facility resource browser and the personnel resource browser, are developed for managing and describing the relevant resource databases. Following this section, the resource agent descriptions are introduced in detail.

4.3.1 Facility Agent Description and Facility Resource Browser

A facility resource agent is defined by its type, manufacturing functions, and time constraints including available periods and unavailable periods. *Facility Resource Browser* is the interface environment to manage and describe the facility resource agents. A snapshot of this browser is shown in Figure 4.7.

The facility resource browser consists of four list views and one text view as illustrated in Figure 4.7. In this browser, all resources are stored in different categories due to the large number of resources. The categories play no role in the resource agent modeling except for helping to find out the required resource agent easily. To add a new resource, either an existing category should be selected or a new category should be added. By selecting one of the categories, the resources in that category are shown in the resource list view. By selecting a resource in the resource list view, a list of aspects is shown in the aspect list view. The aspects include *Type, Function-Constraints, Time-Constraints, and Unexpected-Events.* The aspect list is a built-in list that is common to all resources. Element name list view shows the names of the resource elements in corresponding aspects. These element names are represented with different formats. Text view is used to edit and display the resource descriptions.



AvailablePeriods
"[10/01/1998 00:00 -12/31/1998 23:59]"

Figure 4.7: A snapshot of facility resource browser

Since the production of windows includes the following processes: material cutting, frame construction, auxiliary assembly, glass installation, element assembly, and product packing, the facilities for window manufacturing should include cutting machines, framing machines, assembly units, and packing units. An example of facility agent is described in Table 4.2.

Details regarding the aspects described or displayed in the facility resource browser are introduced in the following of this section except for the aspect of *Unexpected-Events,* which will be discussed in Chapter 5.

Table 4.2: An example of a facility agent description

Facility Name: *FA01*	
Aspects	***Descriptions***
Type	*Type* *"AssemblyUnit"*
Functions-Constraints	*Major* *"assemblyA"* *Others* *"assemblyE"*
Time-Constraints	*AvailablePeriods* *"[10/01/1998 00:00 - 12/31/1998 23:59]"* *UnavailablePeriods* *"[11/10/1998 15:00 - 11/10/1998 16:30], [11/20/1998 15:00 - 11/20/1998 16:30]"*
Unexpected-Events	*UnexpectedEvent1* *ObservationTime: "[11/02/1998 8:00]"* *UnavailablePeriod: "[11/02/1998 15:00 - 11/02/1998 16:30], [11/05/1998 15:00 - 11/05/1998 16:30]"*

(1) Type

A factory usually has different types of facilities for accomplishing various production tasks. For instance, in the general manufacturing job-shop, there are cutting machines, lathes, drilling machines, milling machines, assembly facilities, painting facilities, packing facilities, and so on. In a building product job-shop, some special facilities such as frame machines and glazing facilities are used for window manufacturing. The type of a facility is described using the following template in the text view.

```
Type
    "<facilityType>"
```

For example, in the facility agent *FA01*, the type is defined as

```
Type
    "AssemblyUnit"
```

In the manufacturing requirement representation, each production task specifies a certain type of facility. In the example shown in Figure 4.4, the production task *A1* of instance feature *t* specifies the process *AssemblyA* which will be accomplished on an assembly unit.

(2) Function-Constraints

A facility may have one or several manufacturing functions. Among these functions, one is defined as the major function and the others as the additional functions. For example, an assembly unit can be used to carry out the *assemblyA* (auxiliary assembly), *assemblyE* (element assembly). The *assemblyA* may be defined as its major function, that is, this assembly unit is mainly used to carry out the tasks corresponding to auxiliary assembly in order to reduce the frequency of changing tools. The function constraints of facility are defined using the following template in the text view.

```
Major
    "<function1>"
Others
    "<function2> <function3>"
```

For example, in an assembly unit

```
Major
    "assemblyA"
Others
    "assemblyE"
```

In the manufacturing requirement representation, each production task is associated with a certain type of process. The facility to be allocated to this production task should have a function to match the required process type. In the example shown in Figure 4.4, the production task *A1* of feature *t* should be assigned to an assembly unit with the *Major* function of *assemblyA*. But if there is no such assembly unit available in required time period, another assembly unit that has function *assemblyA* in its *Others* functions may take over this task.

(3) Time-Constraints

The time constraints specify the requirements that the generated schedules must satisfy. The time constraints are described by two elements: *Available-Periods* and *Unavailable-Periods*. The available-period specifies the period in which the facility is available to be used and the unavailable-period indicates when the facility is not available due to maintenance or repair. The examples of these time-constraint elements are shown as follows:

```
AvailablePeriods
    "[10/01/1998 00:00—12/31/1998 23:59]"
```

```
UnavailablePeriods
    "[10/11/1998 00:00—10/21/1998 10:00], [12/1/1998 00:00 —12/1/1999 14:00]"
```

The time is described by the scheme of *month/day/year hour:minute.*

(4) Unexpected-Events

During production process, the generated schedule has to be changed when some facilities break down or need to be urgent inspection and repair. The aspect of unexpected-events is used to describe such disturbance for the system to doing rescheduling. This aspect will be introduced in Section 5.3.3.

4.3.2 Personnel Agent Descriptions and Personnel Resource Browser

A person resource agent is defined by function constraints and time constraints. The function constraints specify facilities that the person can be assigned to work with. The time constraints includes available periods, regular schedule, and unavailable periods.

Personnel Resource Browser is the interface environment to manage and describe the personnel resources. A snapshot of this browser is shown in Figure 4.8. This browser consists of four list views and one text view, similar to *Facility Resource Browser* illustrated in Figure 4.7.

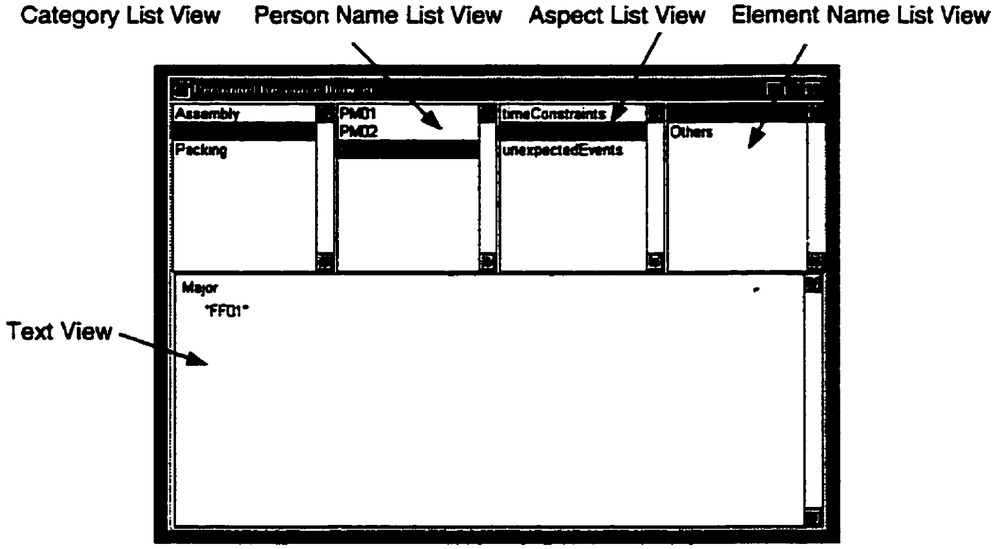


Figure 4.8: A snapshot of personnel resource browser

The management and description of personnel resources in personnel resource browser are similar to the management and description of facility resources in the facility resource browser. The aspects of description of personnel resources *include Function-Constraints, Time-Constraints*, and *Unexpected-Events*. An example of personnel agent is described as Table 4.3.

These aspects described or displayed in the resource browser are introduced with details in the rest of this section, except for the aspect of *Unexpected-Events*, which will be introduced in Chapter 5.

Table 4.3: An example of a personnel agent description

Person Name: *PM03*	
Aspects	***Descriptions***
Functions-Constraints	*Major* *"FF01"* *Others* *"FC01 FC02"*
Time-Constraints	*AvailablePeriods* *"[11/01/1998 00:00 - 12/31/1998 23:59]"* *RegularSchedule* *Day Work Time Break Time* *Monday [08:30-18:00] [12:30-13:00]* *Tuesday [08:30-18:00] [12:30-13:00]* ... *UnavailablePeriods* *"[11/07/1998 12:00 - 11/07/1998 16:30], [11/10/1998 08:00 - 11/10/1998 12:30]"*
Unexpected-Events	*UnexpectedEvent1* *ObservationTime: "[11/02/1998 8:00]"* *UnavailablePeriod: "[11/10/1998 15:00 - 11/10/1998 16:30], [11/20/1998 15:00 - 11/20/1998 16:30]"*

(1) Function-Constraints

A person may be able to operate one or several manufacturing facilities. Among these facilities, one is defined as the major facility and the others as the supernumerary facilities. For example, a person can operate the facilities: *FF01*, *FC01* and *FC02*. The facility *FF01* may be defined as the *Major* function of this person, while the facilities *FC01* and *FC02* as the *Others* function of this person. That is, this person mainly works with *FF01* to carry out the production tasks. When the relevant persons assigned to facility *FC01* and *FC02* can not be available in the required period, this person can act as a substitute on the facilities *FC01* and *FC02*. The function constraints of person are defined using the following template in the text view.

Major

"<function1>"

Others

"<function2> <function3>"

For example, in the personnel agent *PM03*

Major

"FF01"

Others

"FC01 FC02"

(2) Time-Constraints

For personnel resources, the time constraints are described by three elements: ***Available-Periods***, ***Regular-Schedule***, and ***Unavailable-Periods***. The available-period specifies the period in which the person is available to take the production tasks. The regular-schedule shows the routine timetable that the person should follow in every week. The unavailable-period indicates when the person is not available due to some special reasons. The examples of these time-constraint elements are shown as follows:

AvailablePeriods

"[10/01/1998 00:00—12/31/1999 23:59]"

RegularSchedule

Day	*Work Time*	*Break Time*
Monday	*[08:00-16:30]*	*[12:00-12:30]*
Tuesday	*[08:00-16:30]*	*[12:00-12:30]*
Wednesday	*[08:00-16:00]*	*[12:00-12:30]*
Thursday	*[08:00-16:30]*	*[12:00-12:30]*
Friday	*[08:00-15:00]*	*[12:00-12:30]*
Saturday	*[]*	*[]*
Sunday	*[]*	*[]*

UnavailablePeriods

"[10/11/1998 00:00—10/21/1998 10:00], [2/1/1999 00:00 —2/1/1999 14:00]"

---

(3) Unexpected-Events

During production process, the generated schedule has to be changed when some assigned persons are absent for some unexpected reasons. The aspect of unexpected-events is used to describe such disturbance for the system to do rescheduling. This aspect will be introduced in Section 5.3.4.

4.3.3 An Example

This section provides an extended example of the descriptions of manufacturing resources for window production.

Since the production of windows includes the following processes: material cutting, frame construction, auxiliary assembly, glass installation, element assembly, and product packing, eleven facilities are defined in this example including 2 cutting machines, 1 framing machine, 7 assembly units, and 1 packing unit. Eleven persons are signed to work on the relevant facilities in this example. Table 4.4 and Table 4.5 list the descriptions of manufacturing resources for window production.

Table 4.4: Descriptions of facility resources for window production

Facilities	**Descriptions**			
	Types	**Function-Constraints**		**Time-Constraints**
		Major	**Others**	
FC01	CuttingMachine	cutting		AvailablePeriods "[10/01/1998 0:00— 12/31/1998 23:59]" UnavailablePeriods "[10/16/1998 8:00—10/16/1998 8:30]"

Table 4.4: Descriptions of facility resources for window production (continued)

Facilities	**Descriptions**			
	Types	**Function-Constraints**		**Time-Constraints**
		Major	**Others**	
FC02	CuttingMachine	cutting		AvailablePeriods "[10/01/1998 0:00—12/31/1998 23:59]" UnavailablePeriods ""
FF01	FramingMachine	framing		*Same as Time-Constraints in FC02*
FA01	AssemblyUnit	assemblyA		*Same as Time-Constraints in FC02*
FA02	AssemblyUnit	assemblyA		*Same as Time-Constraints in FC02*
FA03	AssemblyUnit	assemblyA	assemblyE	*Same as Time-Constraints in FC02*
FA04	AssemblyUnit	assemblyA	assemblyE	*Same as Time-Constraints in FC02*
FA05	AssemblyUnit	assemblyE		*Same as Time-Constraints in FC02*
FA06	AssemblyUnit	glazing		*Same as Time-Constraints in FC02*
FA07	AssemblyUnit	glazing		*Same as Time-Constraints in FC02*
FP01	PackingUnit	packing		*Same as Time-Constraints in FC02*

Table 4.5: Descriptions of personnel resources for window production

Persons	**Descriptions**		
	Function-Constraints		**Time-Constraints**
	Major Facility	**Other Facilities**	
PM01	***FC01***		AvailablePeriods "[10/01/1998 00:00—12/31/1998 23:59]" RegularSchedule Day Work Time Break Time Mon [08:00-18:00] [12:30-13:00] Tue [08:00-18:00] [12:30-13:00] Wed [08:00-18:00] [12:30-13:00] Thu [08:00-18:00] [12:30-13:00] Fri [08:00-18:00] [12:30-13:00] Sat [] [] Sun [] [] UnavailablePeriods "[10/16/1998 8:00—10/16/1998 8:30]"

Table 4.5: Descriptions of personnel resources for window production (continued)

Person	**Descriptions**		
	Function-Constraints		**Time-Constraints**
	Major Facility	**Other Facilities**	
PM02	***FC02***		*Same as Time-Constraints in PM01*
PM03	***FF01***	***FC01,*** ***FC02***	*Same as Time-Constraints in PM01 except for* UnavailablePeriods “”
PA01	***FA01***		*Same as Time-Constraints in PM03*
PA02	***FA02***		*Same as Time-Constraints in PM03*
PA03	***FA03***		*Same as Time-Constraints in PM03*
PA04	***FA04***		*Same as Time-Constraints in PM03*
PA05	***FA05***		*Same as Time-Constraints in PM03*
PA06	***FA06***		*Same as Time-Constraints in PM03*
PA07	***FA07***		*Same as Time-Constraints in PM03*
PP01	***FP01***	***FC02***	*Same as Time-Constraints in PM01*

4.4 An Intelligent Predictive Scheduling Mechanism

In this research, an intelligent predictive scheduling mechanism is proposed to identify an optimal schedule for production order considering design and manufacturing constraints. The optimal production schedule includes optimal sequences of production processes and their timing parameters and resource allocation. The optimal schedule is achieved by a constructive scheduling approach using heuristic search. During the scheduling process, allocation of resources and timing parameters to each production task is carried out using the agent-based negotiation approach in the distributed resource management module.

4.4.1 Constructive Scheduling Approach Using Heuristic Search

The constructive scheduling approach achieves a complete schedule incrementally through a sequence of intermediate partial schedules using constraints as guidance [Fox 89]. Heuristic search is a method to find a solution from a start state to a goal state through a sequence of intermediate states based upon the evaluations to these states using a heuristic function [Winston 92]. In this research, heuristic search is used to conduct constructive scheduling.

In the proposed constructive scheduling approach, each state represents a partial schedule developed so far using heuristic search. A schedule is described by the sequences to conduct the required manufacturing tasks and assignments of resources and timing parameters to these tasks. A start state is an empty schedule, while a goal state is the schedule in which all the tasks have been allocated with the required resources and timing parameters. All the states are described as the nodes in a search tree.

The algorithm of constructive scheduling using heuristic search is given in Figure 4.9 and illustrated in Figure 4.10.

Step 1. Initialize the start node and identify the to-be-scheduled features and the to-be-scheduled tasks

Step 2. Generate the sub-nodes of the start node using the to-be-scheduled tasks of the to-be-scheduled features

For each of sub-nodes which corresponds to one of the *to-be-scheduleo tasks* (the *current to-be-scheduled task*),

- The order agent negotiates with the resource agents in the *updatea resource management module* to identify the resources and timing parameters to the *current to-be-scheduled task*. The agent-based negotiation mechanism will be introduced Section 4.4.2.
- The sub-node preserves the *scheduling result*.
- The *scheduling result* is evaluated by heuristic function to achieve an *evaluation value.*

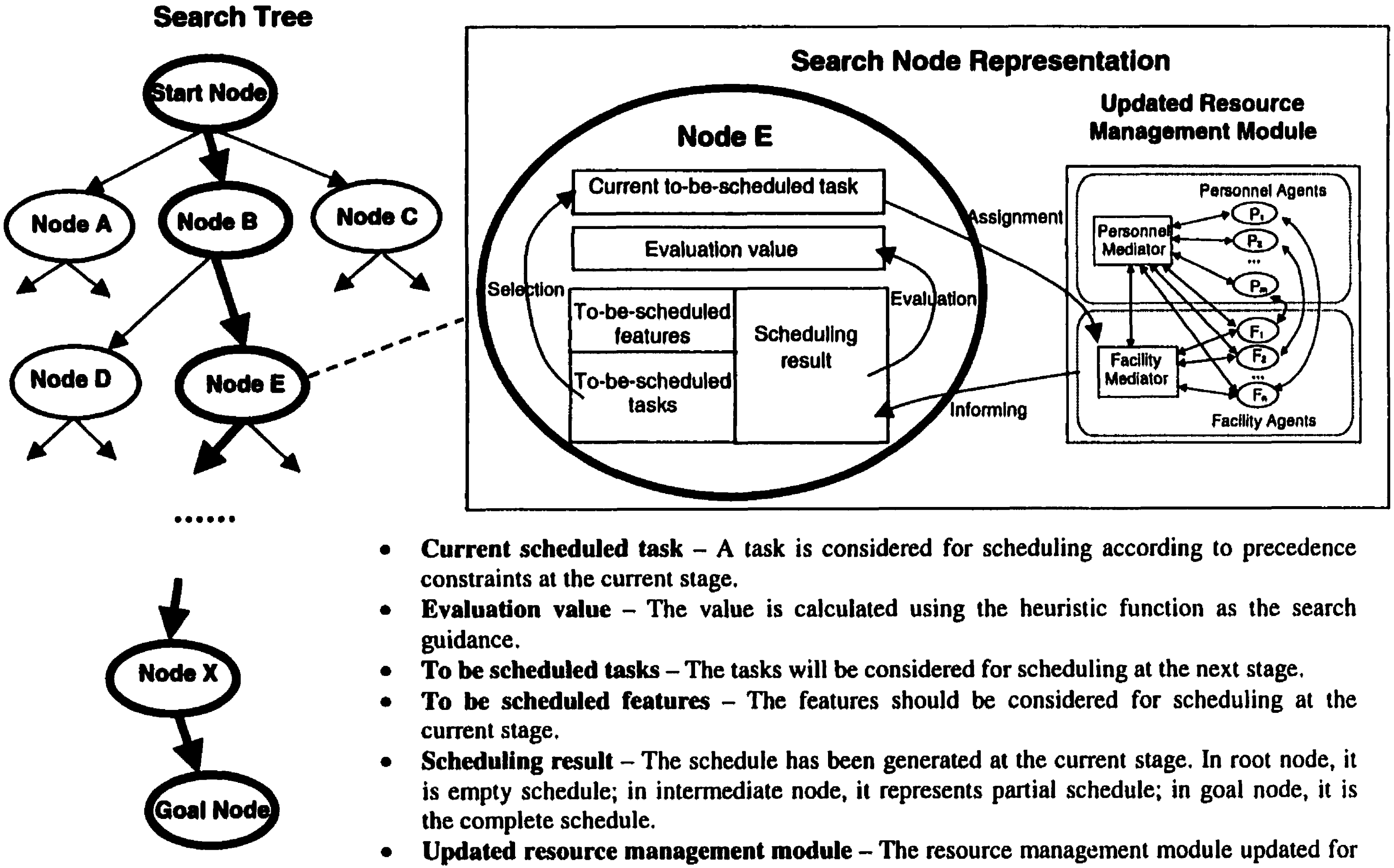
Step 3. Select the best node from the search tree based upon the evaluation value and then generate the sub-nodes of the best nodes using the to-be-scheduling tasks of the to-be-scheduled features

For each of sub-nodes, allocation of resources, assignment of timing parameters, evaluation of *scheduling result* are conducted using the same method as Step 2.

Step 4. Check whether the best node is the goal node

When the best node is the *goal node*, the search stops. Otherwise, the search goes to Step 3.

Figure 4.9: The algorithm for constructive scheduling using heuristic search



- **Current scheduled task** – A task is considered for scheduling according to precedence constraints at the current stage.
- **Evaluation value** – The value is calculated using the heuristic function as the search guidance.
- **To be scheduled tasks** – The tasks will be considered for scheduling at the next stage.
- **To be scheduled features** – The features should be considered for scheduling at the current stage.
- **Scheduling result** – The schedule has been generated at the current stage. In root node, it is empty schedule; in intermediate node, it represents partial schedule; in goal node, it is the complete schedule.
- **Updated resource management module** – The resource management module updated for each new generated search node. This module preserves the scheduling results that have achieved so far.

Figure 4.10: Constructive scheduling approach using heuristic search with agent-based resource management module

In order to achieve optimal production schedule, two heuristic functions can be used to evaluate scheduling result during the scheduling process in this research:

(1) $F_{max} = T_f - T_a$

T_f – the latest task finish time considering all the scheduled tasks

T_a – the arrival time of a production order

(2) $S_{min} = T_d - T_s$

T_d – the due time of production order

T_s – the earliest task start time considering all the scheduled tasks

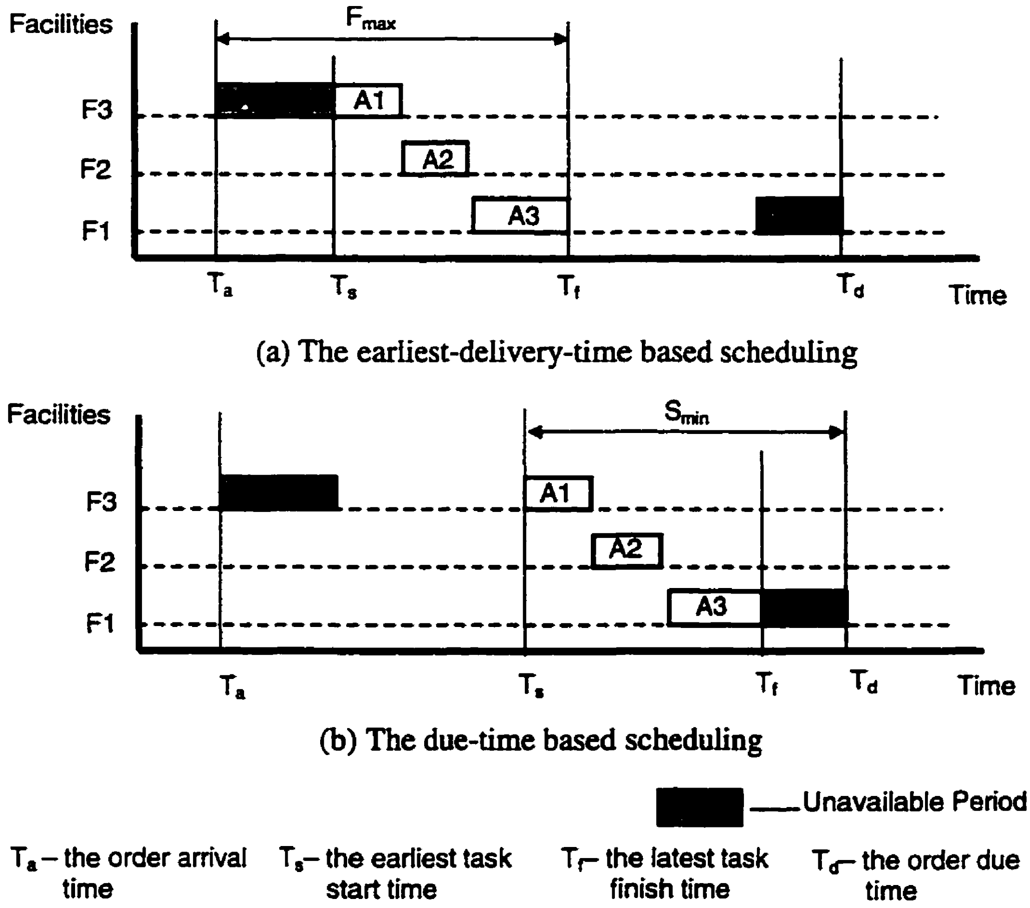
Two scheduling strategies were also developed based upon the two heuristic functions:

(1) The earliest-delivery-time based scheduling – to provide the product to the customer as early as possible by selecting the node with the minimum value of the F_{max} as the best node.

(2) The due-time based scheduling – to start the product manufacturing as late as possible to reduce the space for storing the produced product by selecting the node with maximum value of the S_{min} as the best node.

Two examples are shown in Figure 4.11 to illustrate the two scheduling strategies. In this example, the scheduled production order *A* consists of three manufacturing *tasks A1*, *A2*, and *A3*. The Gantt charts in Figure 4.11 (a) and in Figure 4.11 (b) illustrate the schedules generated using the earliest-delivery-time based scheduling strategy and the due-time based scheduling respectively strategy.

The earliest-delivery-time based scheduling is conducted following the algorithm as shown in Figure 4.12, and the due-time based scheduling following the algorithm as shown in Figure 4.13.



(a) The earliest-delivery-time based scheduling

(b) The due-time based scheduling

Figure 4.11: Examples of two scheduling strategies

The Earliest-Delivery-Time Based Scheduling Strategy

1. **Initialize the start node and identify the to-be-scheduled tasks**

   The *to-be-scheduled tasks* are the tasks that should be considered for scheduling at this stage. The *to-be-scheduled tasks* include all the tasks defined in the bottom features of the product. A bottom feature is the one without any element features and should be identified as the *to-be-scheduled features.*

2. **Generate the sub-nodes of the start node using the to-be-scheduled tasks**

   The *to-be-scheduled tasks* that don't have ancestor tasks are used to generate these sub-nodes. In each of these sub-nodes,

   2.1 The scheduled tasks should be removed from the list of the *to-be-scheduled tasks.*

   2.2 The order agent negotiates with the resource agents in the resource management module to identify the resources and timing parameters to accomplish this task, according to the manufacturing requirements defined in the task and customer's requirements.

Figure 4.12: The earliest-delivery-time based scheduling strategy

After resource allocation is completed, each sub-node is associated with an *updated resource management module* that is updated to preserve the scheduling results achieved so far at this stage.

2.3 The generated nodes are evaluated using the earliest-delivery-time objective function F_{max} in terms of the scheduling results.

3. **Select the best node from the search tree based upon the evaluations to the nodes and then generate the sub-nodes of the best node using the to-be-scheduled tasks**

3.1 The *to-be-scheduled tasks* that have no unscheduled ancestor tasks are used for generating the sub-nodes.

3.2 The allocation of resources and assignment of timing parameters for the new generated nodes are identified using agent-based negotiation with the *updated resource management module*, as Step 2.2. The evaluation using the objective function F_{max} for the newly generated nodes are conducted using the same methods as introduced in Step 2.

3.3 The list of the *to-be-scheduled tasks* for each sub-node is then updated. The scheduled tasks should be removed from this list.

3.4 If all the tasks defined in a feature's element features are scheduled, this feature should be identified as a *to-be-scheduled feature*, and then the tasks of this feature should be added to the list of the *to-be-scheduled tasks*.

4. **Check whether the best node is the goal node**

If the best node doesn't have any *to-be-scheduled tasks*, i.e., the list of the *to-be-scheduled tasks* is empty, this node is the goal node. When the best node is the goal node, the search stops. Otherwise, the search goes to Step 3.

Figure 4.12: The earliest-delivery-time based scheduling strategy (continued)

The Due-Time Based Scheduling Strategy

1. **Initialize the start node and identify the to-be-scheduled tasks**

The *to-be-scheduled tasks* are the tasks that should be considered for scheduling at this stage. The *to-be-scheduled tasks* include all the tasks defined in the top features of the product. A top feature is not an element feature of any feature, and should be identified as a *to-be-scheduled feature*.

2. **Generate the sub-nodes of the start node using the to-be-scheduled tasks**

The *to-be-scheduled tasks* that don't have descendent tasks are used to generate these sub-nodes. In each of these sub-nodes,

2.1 The scheduled tasks should be removed from the list of the *to-be-scheduled tasks*.

2.2 The order agent negotiates with the resource agents in the resource management module to identify the resources and timing parameters to accomplish this task, based on the manufacturing requirements defined in the task and customer's requirements.

Figure 4.13: The due-time based scheduling strategy

After resource allocation is completed, each sub-node is associated with *an updated resource management module* that is updated to preserve the scheduling results achieved so far at this stage.

2.3 The generated nodes are evaluated using the due-time objective function S_{min} in terms of the scheduling result.

3. **Select the best node from the search tree based upon the evaluations to the nodes and then generate the sub-nodes of the best node using the *to-be-scheduled tasks*.**

3.1 The *to-be-scheduled tasks* that have no unscheduled descendent tasks are used for generating the sub-nodes.

3.2 The allocation of resources and assignment of timing parameters for the new generated nodes are identified using agent-based negotiation with the *updated resource management module*, as Step 2.2. The evaluation using the objective function S_{min} for the newly generated nodes are conducted using the same methods as introduced in Step 2.

3.3 The list of the *to-be-scheduled tasks* for each sub-node is then updated. The scheduled tasks should be removed from this list.

3.4 If all the tasks of a feature are scheduled, this feature's element features should be identified as *to-be-scheduled features*, and the tasks in this feature's element features should be added to the list of the *to-be-scheduled tasks*.

4. **Check whether the best node is the goal node.**

If the best node doesn't have any *to-be-scheduled tasks*, i.e., the list of the *to-be-scheduled tasks* is empty, this node is the goal node. When the best node is the goal node, the search stops. Otherwise, the search goes to Step 3.

Figure 4.13: The due-time based scheduling strategy (continued)

Since a product is usually composed of many instance features, and each of these features is produced by a number of tasks, the search space generated during the intelligent scheduling process is very large, therefore resulting in the difficulty to achieve the desired results within the limited time period. To improve the search efficiency, the beam search method was employed in this research (Winston, 1992). In this method, when the number of nodes at a certain level of the search tree excesses the pre-defined number, the nodes with poor evaluation measures are deleted from the search tree, as introduced in Section 2.3.3.

4.4.2 Allocation of Resource and Assignment of Task Timing Parameters Using Agent-Based Negotiation

During the scheduling process, allocation of resources and assignment of task timing parameters are conducted based upon the agent-based negotiation in the multi-agent environment. The agent-based negotiation mechanism was developed by combining the bidding mechanism and the mediation mechanism.

The bidding mechanism was implemented based on the contract-net protocol [Smith 80]. In this protocol, the agent that needs to solve the problem broadcasts a call-for-bids message to all the potential contractor agents, waits for replying messages for a period of time, and then awards the contract to the agent that can provide the best offer according to the contract selection criteria. In the intelligent scheduling process, each to-be-scheduled task is announced by the order agent during the heuristic search. When the facility mediator then receives this announcement, it sends the messages to all the relevant facility agents it knows. These facility agents subsequently start to evaluate the requirements of the task, communicate with the relevant person agents through the personnel mediator, and make bids to the facility mediator. Each bid is made based on the capabilities and planed schedules of the facility and the relevant person. The facility mediator at last assigns the required production task to the relevant agents with the best bids.

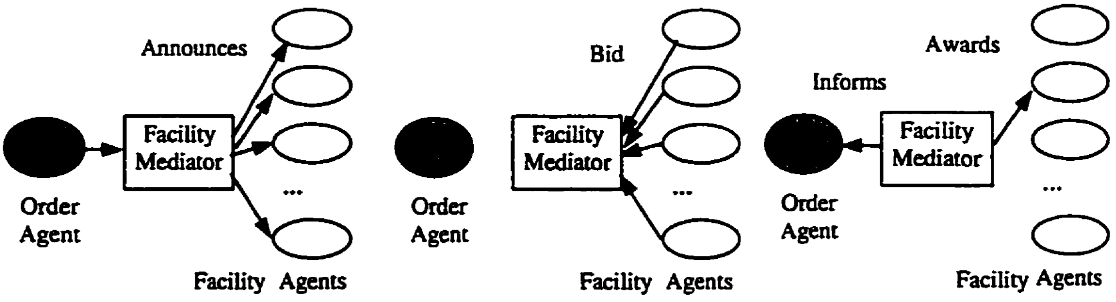
The mediation mechanism is used to coordinate the activities of the relevant agents to improve the scheduling efficiency. In this research, two mediators, facility mediator and personnel mediator, are used to coordinate the activities of the facility resource agents and person resource agents, respectively.

The agent-based negotiation in the scheduling process is conducted at two different levels: order-facility negotiation level and facility-person negotiation level.

(1) Order-Facility Negotiation Level

At the order-facility level, the negotiation is conducted to allocate the resources and assign timing parameters to the requested task, based on the bids generated through the negotiation at facility-person level. The negotiation mechanism at the order-facility level is shown in Figure 4.14.

In this mechanism, the facility mediator should identify the earliest possible start time or the latest possible finish time, before announcing the current scheduled task.



Step 1: Requirement announcement   Step 2: Bidding   Step 3: Awarding

Step 1. The facility mediator receives a request to schedule a task from an order agent and then announces it.

When the facility mediator receives a request to schedule the current scheduled task from an order agent, it sends out this request to the relevant facility agents based on its knowledge.

Step 2. Each relevant facility agent makes a bid and then sends it to the facility mediator.

Upon receiving the announced message, each relevant facility agent checks the its available time periods and starts *the negotiation at the facility-person level* to select proper personnel agent and identify the best bid. The facility agent sends its best bid which is represented by the proposed start time, finish time and person, to the facility mediator.

Before announcing the current scheduled task, the facility mediator identifies the earliest possible start time or the latest possible finish time.

Step 3. The facility mediator selects the best bid and awards the task to the facility agent and the relevant personnel agent, and then informs the order agent the resource allocation and timing parameter assignment for the scheduled task.

The facility mediator at last selects the facility agent that provides the best bid, such as the bid with the minimum value of the latest task finish time in the earliest-delivery-time based scheduling, or the bid with the maximum value of the earliest task start time in the due-time based scheduling. When the decision is made, the facility mediator sends the award the request task to the selected facility agent and personnel agent.

Figure 4.14: The negotiation mechanism at the order-facility level

- In the earliest-delivery-time based scheduling, ***the earliest possible start time*** of the current to-be-scheduled task is the time from which the task is available to be carried out. It is equal to the latest of finish time values of all tasks that have been scheduled so far. If there is no such task, the earliest possible start time is equal to the latest of the finish time values of all the tasks in the element features of the feature that includes the current scheduled task.

For example, an instance feature of class feature *Window*, as introduced in Section 4.2, is the scheduled product. If the current to-be-scheduled task is *Task w.A1*, and *Task w.A2* have been scheduled at this stage, the earliest possible start time of this task should be the latest of the finish times of *Task w.A1* and *Task w.A2*. If the current scheduled task is *Task w.A1*, and *Task w.A2* has not been scheduled, the earliest possible start time should be the latest of the finish times of all the tasks in the element features, *t*, *r*, *l*, and *c*.

- In the due-time based scheduling, ***the latest possible finish time*** of the current scheduled task is the time at which the task should be finished. It is equal to the earliest of start time values of all tasks that have been scheduled so far. If there is no such task, the latest possible finish time is equal to the earliest of the start time values of all the tasks in the super features of the feature that includes the current scheduled task.

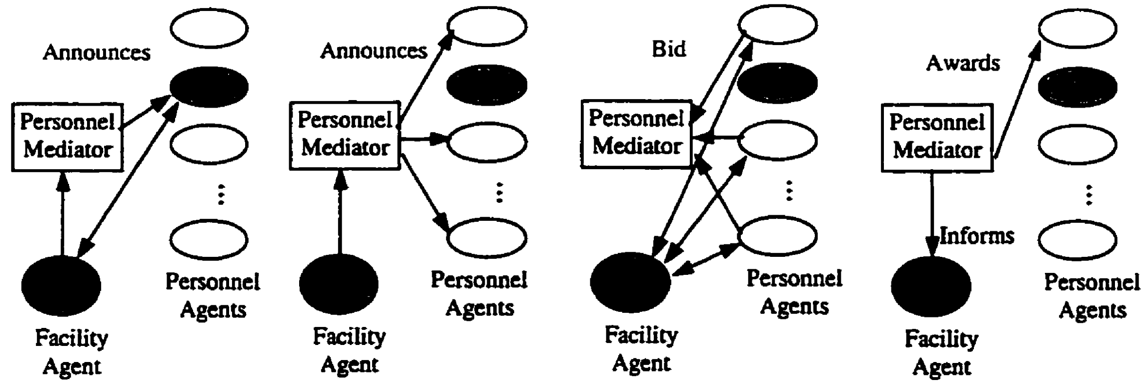
  For example, an instance feature of class feature *Window*, as introduced in Section 4.2, is the scheduled product. If the current to-be-scheduled task is *Task c.A1* and *Task c.A2* and *Task c.A3* have been scheduled at this stage, the latest possible finish time of this task should be the earliest of start time values of *Task c.A2* and *Task c.A3*. If the current scheduled task is *Task c.A3* and *Task c.A2* has not been scheduled, the latest possible finish time should be the earliest of the start time values of all the tasks in its super feature, *w*.

(2) Facility-Person Negotiation Level

At the facility-person level, the negotiation is conducted to select a proper person to associate with the selected facility and to identify the best bid considering the time constraints of both the facility agent and the personnel agent. The negotiation mechanism at the facility-person level is shown in Figure 4.15.

An example is given in Figure 4.16, to illustrate the proposed agent-based negotiation mechanism.

During the earliest-delivery-time based scheduling process, the allocation of resources and assignment of timing parameters need to be identified for the task *C* —



Step 1: Announcing to one personnel agent

Step 2.1: Announcing to other personnel agents

Step 2.2: Bidding

Step 2.3: Awarding

Step 1. **The facility agent communicates with a relevant personnel agent to make a bid.**

After receiving the announced message, first the facility agent communicates with a relevant person, who is mainly assigned to this facility, through the personnel mediator to generate a bid for the requested task.

Step 2. **The facility agent negotiates with other personnel agents through the personnel mediator to make the better bids, and then select the best one.**

If the bid does not meet the required constraints, the facility agent then starts negotiation with the other personnel agents through the personnel mediator. The contract-net bidding mechanism is used to identify the best personnel agent for this facility.

2.1 Upon receiving a task request from the facility agent, the personnel mediator announces it to the relevant personnel agents according to its knowledge.

2.2 Each of these person agents then negotiates with the facility agent to identify the bid for the requested task.

2.3 The personnel mediator selects a proper personnel agent that can provide the best bid. This bid is then sent to the facility mediator as the bid, which will be used by the facility agent at the order-facility negotiation level.

Figure 4.15: The negotiation mechanism at the facility-person level

Type: cutting, Facility: CuttingMachine, Duration: 10 (min). Assuming the earliest possible start time of this task has been identified as 08:00 AM before scheduling this task.

First, the order agent sends the request for scheduling *Task C* to the facility mediator and the facility mediator announces this request to the two relevant facility agents: *FC01* and *FC02*. Their types are *CuttingMachines* and *Major* functions are *cuttings*. *Major* functions of *PM01* and *PM02* are *FC01* and *FC02* respectively. *PM03* and *PP01* can be assigned to work on *FC02,* because the *Others* functions of both *PM03* and *PP01* include *FC02.*

Upon receiving the request announcement, the two facility agents negotiate with the

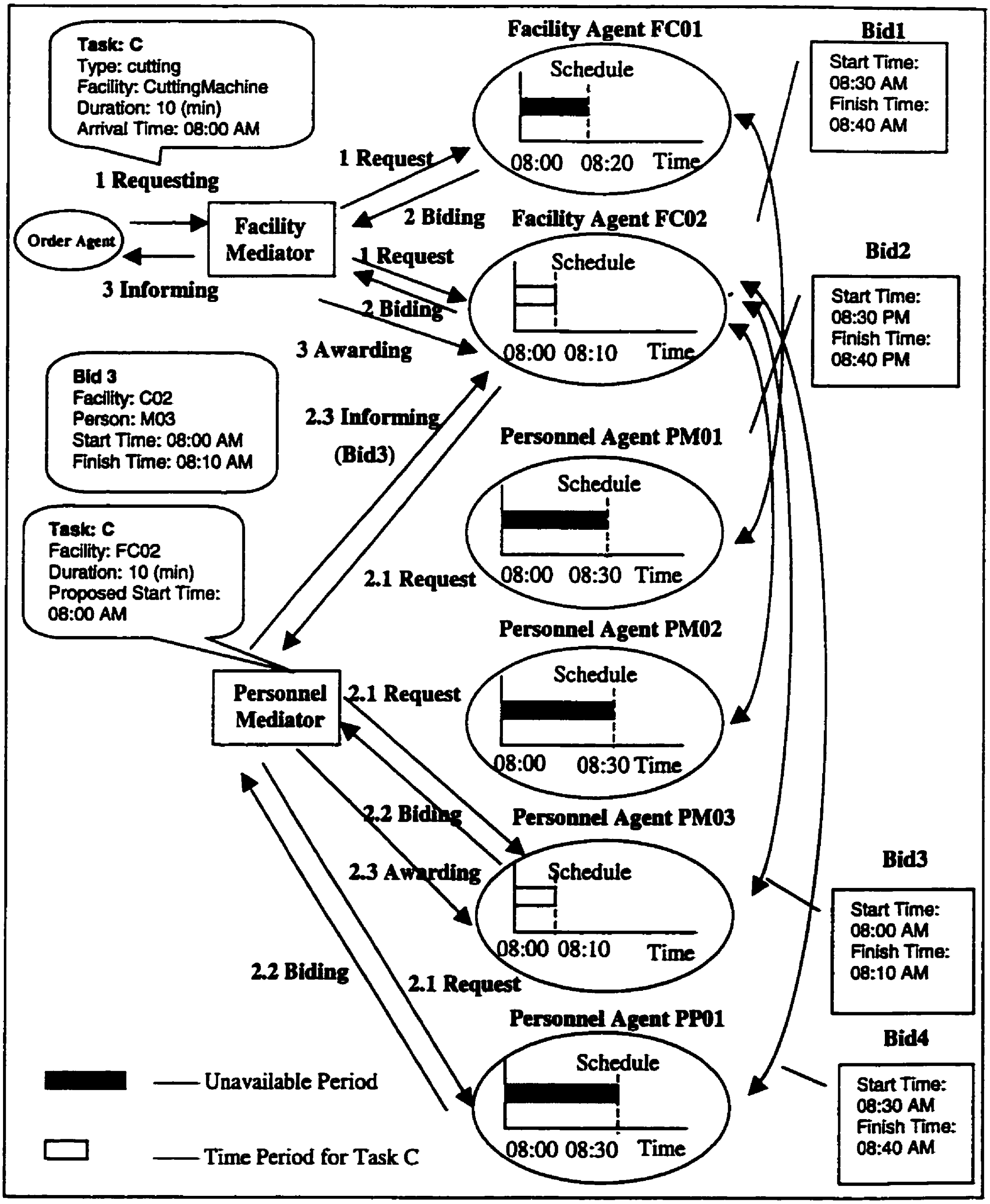


Figure 4:16: An example of agent-based negotiation for allocation of resources and assignment of timing parameters for a required task

relevant personnel agents to identify the bids.

- The facility agent *FC01* negotiates with the personnel agent *PM01* to make a bid, *Bid1* — Start Time: 08:30 AM and Finish Time: 08:40 AM.
- The facility agent *FC02* firstly negotiates with the personnel agent *PM02* through the personnel mediator to make a bid, *Bid2* — Start Time: 08:30 PM and Finish Time: 08:40 AM. It is defined in this system that if the span between the earliest possible start time and the assigned start time in the bid is over 30 minutes, the system will look for better bids. Therefore, the facility agent *FC02* further negotiates with other personnel agents: *PM03* and *PP01* in order to made a better bid. These two personnel agents mainly work with the facilities *FF01* and *FA01* respectively and also can be assigned to work on the facility *FC02* as the substitutes. After the negotiation, they make two bids: *Bid3* — Start Time: 08:00 AM and Finish Time: 08:10 AM and *Bid4* — Start Time: 08:30 AM and Finish Time: 08:40 AM. *Bid3* is the best bid and the personnel agent *PM03* is selected to work on the facility agent *FC02* for *Task C*.

When finishing the negotiation at the facility-person level, the facility agents *FC01* and *FC02* sends their best bids to the facility mediator. The facility mediator finally selects *Bid3* as the best bid and then awards *Task C* to the facility agent *FC02* and the personnel agent *PM03*. At last the facility mediator informs the order agent the message of resource allocation and timing parameter assignment for *Task C*.

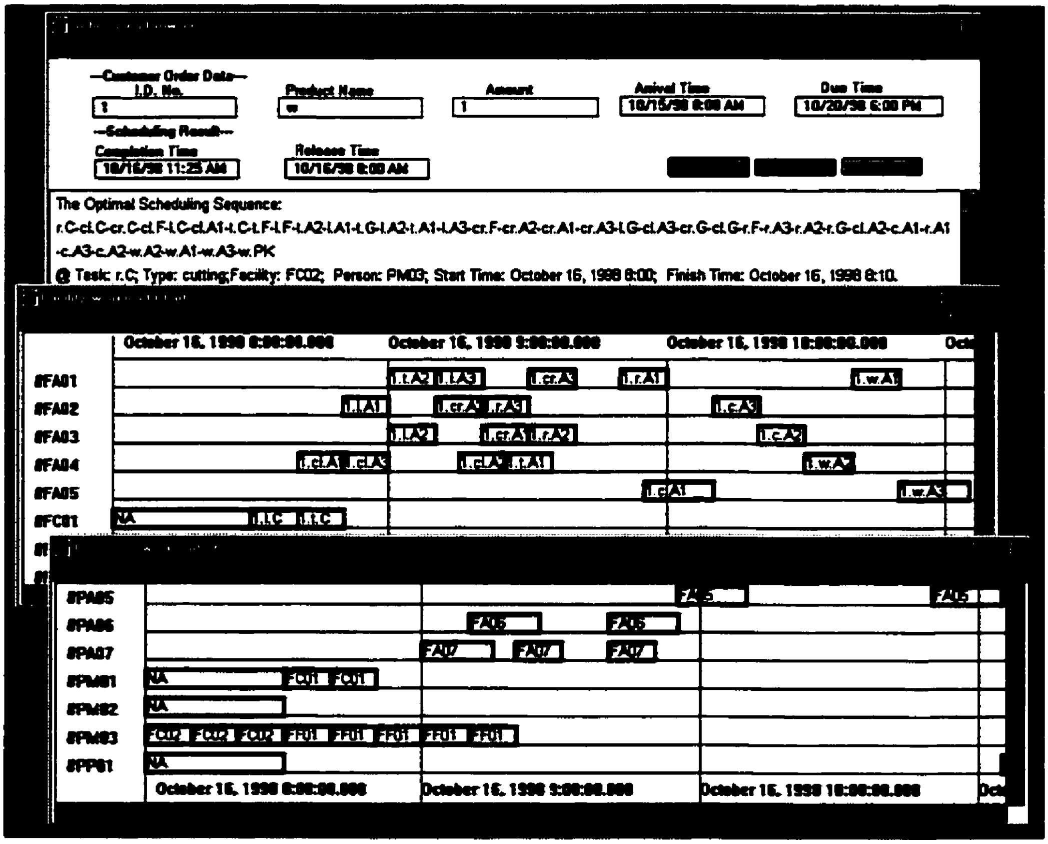
4.4.3 Scheduling Browser and Resource Workload Charts

As shown in Figure 4.17, in this system, ***Scheduling Browser*** is a user interface for generating the production order and conducting the scheduling process. The scheduling results are represented in the text view of scheduling browser and on the resource workload charts: ***Facility Workload Chart*** and ***Personnel Workload Chart***.

(1) Scheduling Browser

In the scheduling browser as shown in Figure 4.17, the production order data is entered including the I.D. number of the order, product name, amount, arrival time of the order, and the due time of order. The ordered product should be generated as an instance feature firstly in the instance feature browser. The arrival time of order represents the time at which the production order arrives at the factory. The due time is the time by which the production should be completed.

After generating a production order, the system begins to conduct the scheduling process for the order. There are two objective functions: *Due-Time Based Scheduling* and *Earliest-Delivery-Time Based Scheduling*. The user can select one of them from the menu *Objectives* as the evaluation measure to generate the optimal schedules. When the system has accomplished the scheduling process, the scheduling results for the current



—Customer Order Data—

I.D. No.	Product Name	Amount	Arrival Time	Due Time
1	w	1	10/15/98 8:00 AM	10/20/98 6:00 PM

—Scheduling Result—

Completion Time	Release Time
10/16/98 11:25 AM	10/16/98 8:00 AM

The Optimal Scheduling Sequence:
r.C-cl.C-cr.C-cl.F-l.C-cl.A1-l.C-l.F-l.F-l.A2-l.A1-l.G-l.A2-l.A1-l.A3-cr.F-cr.A2-cr.A1-cr.A3-l.G-cl.A3-cr.G-cl.G-r.F-r.A3-r.A2-r.G-cl.A2-c.A1-r.A1-c.A3-c.A2-w.A2-w.A1-w.A3-w.PK

@ Task: r.C; Type: cutting;Facility: FC02; Person: PM03; Start Time: October 16, 1998 8:00; Finish Time: October 16, 1998 8:10.

Figure 4.17: A snapshot of the scheduling browser and the resource workload charts

production order are shown in the scheduling browser. *Release Time* is the time at which the production order will begin to be manufactured. *Completion Time is* the time at which the production order will be finished. The scheduling result is shown with detail in the text view of scheduling browser. The syntax of the results is shown by the following format:

```
The Optimal Scheduling Sequence:
<task1-task2-task3-...... >
@task1: Type: <type>; Facility: < name>; Person: <name>; Start Time: < time>; Finish
  Time: <time>.
@task2: Type: <type>; Facility: < name>; Person: <name>; Start Time: < time>; Finish
  Time: <time>.
@task3: Type: <type>; Facility: < name>; Person: <name>; Start Time: < time>; Finish
  Time: <time>.
......
```

(2) Resource Workload Charts

In this system, the resource workload charts serve as aid for visualizing the scheduling results in form of Gantt Chart. The resource workload charts includes *Facility Workload Chart* and *Personnel Workload Chart*, as shown in Figure 4.17. These workload charts display all the generated schedules of the resources. The schedules can be shown with different levels of details, such as a monthly schedule, a weekly schedule, a daily schedule, and an hourly schedule, using the *Zoom In* and *Zoom Out* functions from menu *Zoom*. For the current production order, if the generated scheduling result is satisfied by the user, the result will be added on the workload charts after the user confirms it using the *Confirm* button in the scheduling browser.

In the facility workload chart, rectangles represent the unavailable periods marked with *NA* and the work periods that have been assigned to production tasks. The production tasks are represented using the format *<Order ID>.<Feature Name>.<Task Name>*. For example, the task *A1* of feature *t* in *Order 1* is expressed as *1.t.A1*. In the personnel workload chart, the same expression method as the facility workload chart is used except for marking the work periods with the corresponding facility name for the scheduled task instead of the scheduled task name.

4.4.4 The Extended Examples of the Two Scheduling Strategies

In this section, an extended example is used to illustrate the two scheduling strategies, including the earliest-delivery-time based scheduling and the due-time-based scheduling. *Order 1* arrives at 8:00 AM on October 15, 1998 and is scheduled with the earliest-delivery-time objective. *Order 2* has the due time request of 6:00 PM on October 19, 1998 and is scheduled with the due-time objective. The product to be ordered is called *w*, an instance feature of class feature *Window*, as introduced in Section 4.2. The manufacturing resources for window production were described in Section 4.3.3.

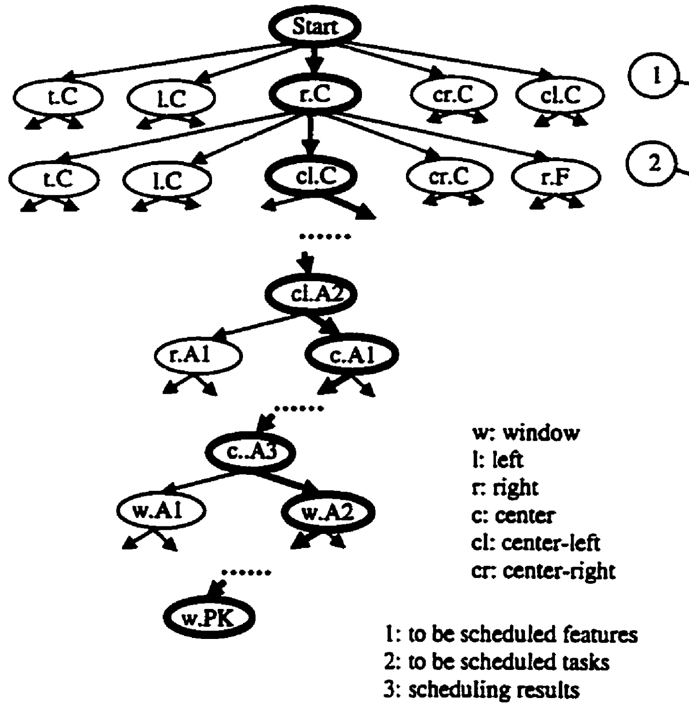
(1) An Example of the Eariliest-Dellevery-Time Based Scheduling Strategy

Figure 4.18 illustrates the earliest-delivery-time based scheduling process for *Order 1* in this example.

First, a root node is created. All the tasks in the 5 bottom features are considered as the to-be-scheduled tasks for the root node. Since the tasks *t.C, l.C, r.C, cl.C,* and *cr.C* have no unscheduled ancestor tasks, these tasks are used for generating the sub-nodes of the root node. If the *r.C* is the best node, the 5 to-be-scheduled tasks, which have no unscheduled ancestor tasks, are used for generating the sub-nodes of the *r.C,* as shown in Figure 4.17. The descriptions of the nodes in the best-path (shown using the dark lines), including the to-be-scheduled features, the to-be-scheduled tasks, and the scheduling results, are illustrated using 3 examples: *Start, cl.A2,* and *w.PK.*

- The node *Start* represents the starting state in the search. Five bottom level instance features, *cl, cr, t, l,* and *r,* are considered in scheduling. The scheduling results are described by empty lists.
- The node *cl.A2* describes the state when all the tasks in the *cl* and *cr* instance features have been scheduled. At this stage, the instance features, *cl* and *cr,* should

be removed from the list of the to-be-scheduled features. The instance feature, *c*, should be added to the list of the to-be-scheduled features. In the scheduling result



Start

cl, cr, t, l, r	cl: cr: t: l: r:
cl.C, ... cr.C, ... t.C, ... l.C, ... r.C, ...	

cl.A2

r, c	cl: C-F-A1-A3-G-A2 cr: C-F-A2-A1-A3-G t: C-F-A2-G-A1 l: C-F-A1-A2-A3-G r: C-F-A3-A2-G c:
r.A1, c.A1, c.A2, c.A3	

w.PK

	cl: C-F-A1-A3-G-A2 cr: C-F-A2-A1-A3-G t: C-F-A2-G-A1 l: C-F-A1-A2-A3-G r: C-F-A3-A2-G-A1 c: A1-A3-A2 w: A2-A1-A3-PK

cl.A2

Features	Scheduling Results
cl	C: 8:10-8:20 → F: 8:30-8:40 → A1: 8:40-8:50 → A3: 8:50-9:00 → G: 9:00-9:15 → A2: 9:15-9:25
cr	C: 8:20-8:30 → F: 9:00-9:10 → A2: 9:10-9:20 → A1: 9:20-9:30 → A3: 9:30-9:40 → G: 9:40-9:55
t	C: 8:40-8:50 → F: 8:50-9:00 → A2: 9:00-9:10 → G: 9:10-9:25 → A1: 9:25-9:35
l	C: 8:30-8:40 → F: 8:40-8:50 → A1: 8:50-9:00 → A2: 9:00-9:10 → A3: 9:10-9:20 → G: 9:20-9:30
r	C: 8:00-8:10 → F: 9:10-9:20 → A3: 9:20-9:30 → A2: 9:30-9:40 → G: 9:40—9:50
c	

w.PK

Features	Scheduling Results
cl	C: 8:10-8:20 → F: 8:30-8:40 → A1: 8:40-8:50 → A3: 8:50-9:00 → G: 9:00-9:15 → A2: 9:15-9:25
cr	C: 8:20-8:30 → F: 9:00-9:10 → A2: 9:10-9:20 → A1: 9:20-9:30 → A3: 9:30-9:40 → G: 9:40-9:55
t	C: 8:40-8:50 → F: 8:50-9:00 → A2: 9:00-9:10 → G: 9:10-9:25 → A1: 9:25-9:35
l	C: 8:30-8:40 → F: 8:40-8:50 → A1: 8:50-9:00 → A2: 9:00-9:10 → A3: 9:10-9:20 → G: 9:20-9:30
r	C: 8:00-8:10 → F: 9:10-9:20 → A3: 9:20-9:30 → A2: 9:30-9:40 → G: 9:40-9:50 →A1: 9:50-10:00
c	A1: 9:55-10:10 → A3: 10:10-10:20 → A2: 10:20-10:30
w	A2:10:30-10:40 → A1: 10:40-10:50 → A3: 10:50-11:05 → PK: 11:05-11:25

Figure 4.18: An example for the earliest-delivery-time based scheduling

lists, all the tasks of the instance features, *cl* and *cr*, have been scheduled. The tasks of the other to-be-scheduled instance features are only partially scheduled.

- The node, *w.PK*, is the goal node in the search. At this stage, all the tasks in all the instance features have been scheduled. Therefore, the lists of the to-be-scheduled features and the to-be-scheduled tasks are empty lists. The final scheduling results are the sequences of the tasks, resource allocations to these tasks, and timing parameter assignments to these tasks.

Figure 4.19 illustrates the predictive scheduling results generated by the system.

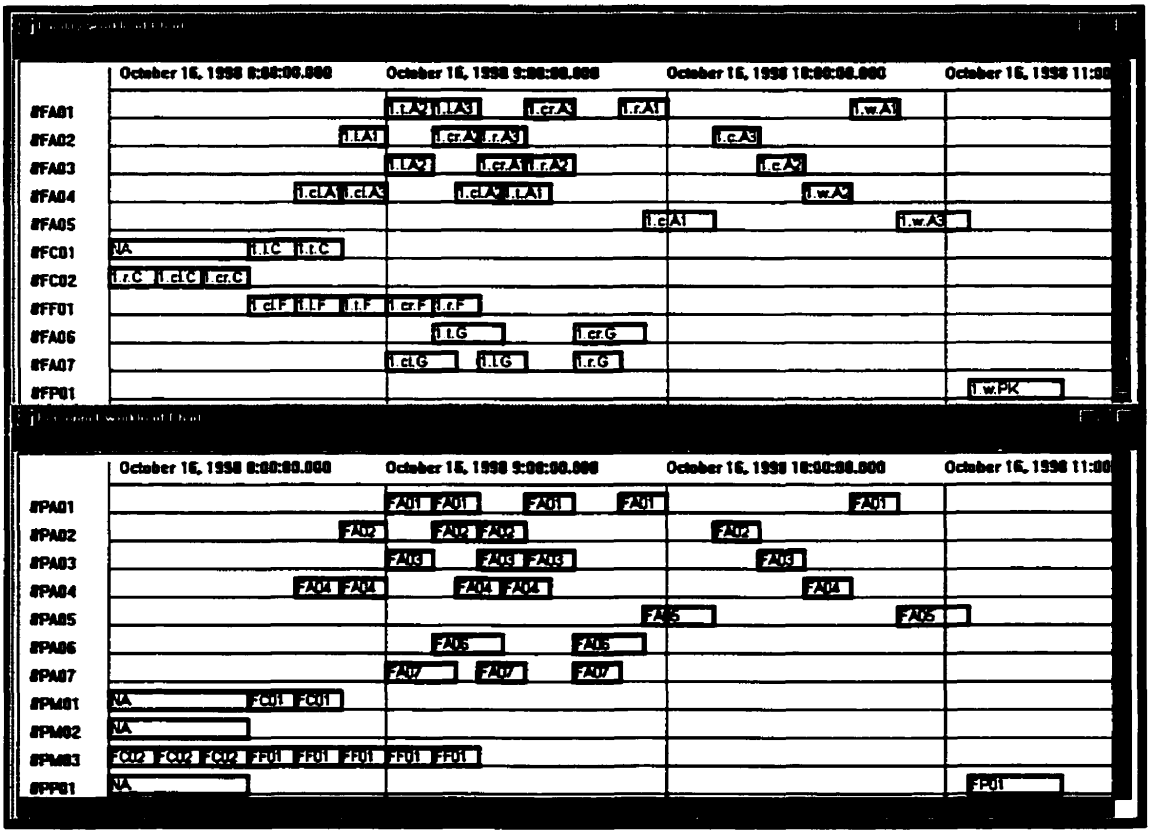
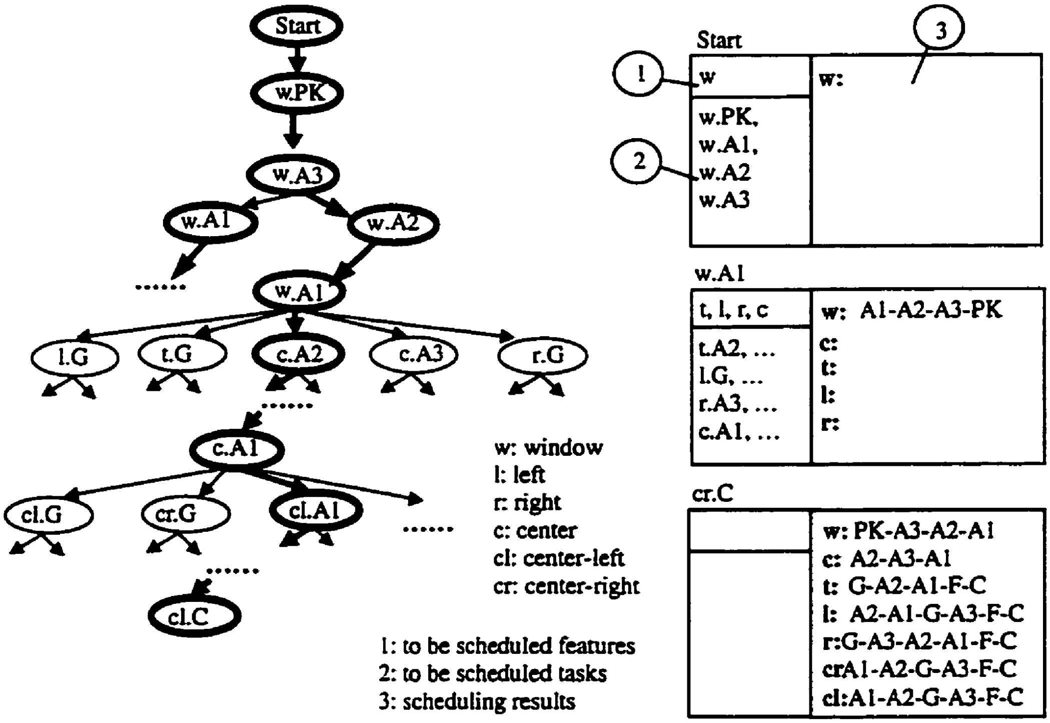


Figure 4.19: A snapshot of the earliest-delivery-time based scheduling result shown on the resource workload charts

(2) An Example of the Due-Time Based Scheduling Strategy

Figure 4.20 illustrates the due-time based scheduling process for *Order 2* of this example.



w.A1

Features	Scheduling Results
w	PK: 18:00-17:40➔A3: 17:40-17:25➔A2: 17:25-17:15➔A1: 17:15-17:05
c	
t	
l	
r	

cl.C

Features	Scheduling Results
w	PK: 18:00-17:40➔A3: 17:40-17:25➔A2: 17:25-17:15➔A1: 17:15-17:05
c	A2: 17:05-16:55➔ A3: 16:55-16:45 ➔ A1: 16:45-16:35
t	G: 17:05-16:50 ➔ A2: 16:45-16:35 ➔ A1: 16:20-16:00 ➔ F: 15:25-15:15 ➔ C: 15:25-15:15
l	A2: 17:05-16:55 ➔ A1: 16:55-16:45 ➔ G: 16:45-16:35 ➔ A3: 16:35-16:25 ➔ F: 16:00-15:50➔C: 15:50-15:40
r	G: 17:05-16:55 ➔ A3: 16:55-16:45 ➔ A2: 16:45-16:35 ➔ A1: 16:35-16:25 ➔ F: 16:25-16:15➔C: 16:15-16:05
cr	A1: 16:30-16:20 ➔ A2: 16:20-16:10 ➔ G: 16:10-15:55➔ A3: 15:55-15:45 ➔ F: 15:45-15:35➔C: 15:35-15:25
cl	A1: 16:30-16:20 ➔ A2: 16:20-16:10 ➔ G: 16:10-15:55➔ A3: 15:55-15:45 ➔ F: 15:25-15:15➔C: 15:15-15:05

Figure 4.20: An example for the due-time based scheduling

First, a root node is created. All the tasks in the top features are considered as the to-be-scheduled tasks for the root node. Since the task *w.PK* has no unscheduled descendant tasks, these tasks are used for generating the sub-nodes of the root node. Next, the task *w.A3,* which has no unscheduled descendant tasks, is used for generating the sub-node of the *w.PK.* Then, the 2 to-be-scheduled tasks, *w.A1* and *w.A2,* which have no unscheduled descendant tasks, are used for generating the sub-nodes of the *w.A3,* as shown in Figure 4.18. The descriptions of the nodes in the best-path (shown using the dark lines), including the to-be-scheduled features, the to-be-scheduled tasks, and the scheduling results, are illustrated using 3 examples: *Start, w.A1,* and *cl.C.*

- The node *Start* represents the starting state in the search. One top level instance feature, w, is considered in scheduling. The scheduling results are described by empty lists.

- The node *w.A1* describes the state when all the tasks in the w instance features have been scheduled. At this stage, the instance features, *w,* should be removed from the list of the to-be-scheduled features. The instance features, *c, r, l,* and *t,* should be added to the list of the to-be-scheduled features. In the scheduling result lists, all the tasks of the instance feature, *w,* have been scheduled.

- The node, *cl.C,* is the goal node in the search. At this stage, all the tasks in all the instance features have been scheduled. Therefore, the lists of the to-be-scheduled features and the to-be-scheduled tasks are empty lists. The final scheduling results are the sequences of the tasks, resource allocation to these tasks, and timing parameter assignments to these tasks.

Figure 4.21 illustrates the predictive scheduling results generated by the system.

4.5 Summary

This chapter presented the predictive scheduling mechanism of the intelligent production scheduling system. The manufacturing requirements, including the tasks and the sequential constraints for accomplishing these tasks, are described using the feature-

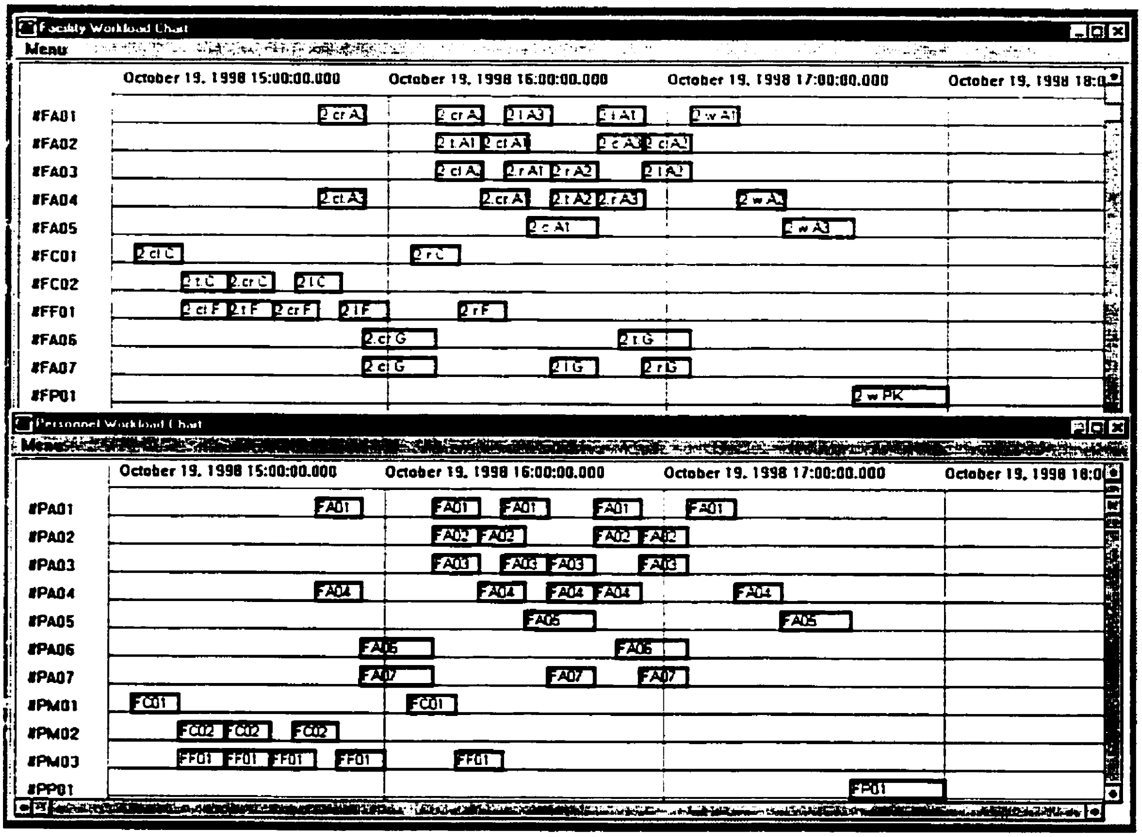


Figure 4.21: A snapshot of the due-time based scheduling result shown on the resource workload charts

based product representation scheme. Each task specifies its resource requirements and time period to conduct this task. The manufacturing resources, including facilities and persons, are modeled as agents. Two mediators, facility mediator and person mediator, are used to coordinate the activities of these resource agents. The optimal production schedule for producing the products ordered by customers is achieved using heuristic search and agent-based negotiation. The introduced method provides a new approach to predictive production scheduling by considering the constraints of both design and manufacturing aspects.

Chapter 5

Reactive Scheduling

This chapter presents the reactive scheduling aspect of the intelligent production scheduling system. Two intelligent reactive scheduling mechanisms have been developed in this research to revise original schedules for responding to production demand changes and resource changes respectively. The intelligent reactive scheduling is conducted based on the match-up rescheduling approach and the agent-based negotiation in a multi-agent environment. Section 5.2 introduces the reactive scheduling mechanism for responding to production demand changes, including canceling old orders and inserting urgent orders. Section 5.3 introduces the reactive scheduling mechanism for responding to resource changes, including facility breakdowns and personnel absence.

5.1 Introduction

In this research, two intelligent reactive scheduling mechanisms have been developed for schedule revision to respond to production demand changes and resource changes. The production demand changes include (1) canceling old orders that have been scheduled previously and (2) inserting new orders that can not be scheduled using predictive scheduling due to the urgent due time requirements. The resource changes include (1) facility breakdowns and (2) personnel absence.

The intelligent reactive scheduling is conducted based on match-up rescheduling approach and agent-based negotiation. The match-up rescheduling approach was proposed based on the assumption that enough idle time is presented in the original schedule. In the match-up rescheduling approach, when unexpected events, such as facility breakdowns and personnel absence, disrupt the original schedule, the original schedule should be changed partially so that the revised schedule can maximally match up with the original schedule [Bean 91; Zweben 94]. In this research, the match-up rescheduling approach was implemented in a multi-agent environment for responding to production demand changes and resource changes. During the reactive scheduling process, agent-based negotiation is used to reassign timing parameters and reallocate resources to the production tasks that need to be rescheduled.

5.2 Reactive Scheduling for Production Demand Changes

An intelligent reactive scheduling mechanism was proposed in this research for responding to production demand changes. Normally, the intelligent production scheduling system conducts predictive scheduling for production orders based on first-come-first-serve rule. However, reactive scheduling is required for the two special cases in production demand changes:

(1) To cancel an old order — If customer requests to cancel an order that has been scheduled previously, the system will revise the original schedule in order to improve the schedule quality for the remaining orders.

(2) To insert an urgent order — If a feasible schedule for an order can not be identified using the predictive scheduling due to urgent due-time requirement, the order should be treated as an urgent order and inserted in the original schedule. However, if the revised schedule for inserting this urgent order is not feasible, such as some orders cannot be scheduled in the period from the current time to their due-times, the urgent order will be rejected.

In this section, the intelligent reactive scheduling mechanism will be described in the following two aspects: (1) revising schedule using match-up rescheduling approach, and (2) reassigning timing parameters using agent-based negotiation.

5.2.1 Revising Schedule Using Match-up Rescheduling Approach

A simple scenario is given in Figure 5.1 to illustrate the match-up rescheduling approach. Two kinds of orders are presented in the original schedule as shown in Figure 5.1 (a).

- The orders *A* and *B* were scheduled using the earliest-delivery-time based scheduling.
- The orders *C* and *D* were scheduled using the due-time based scheduling.

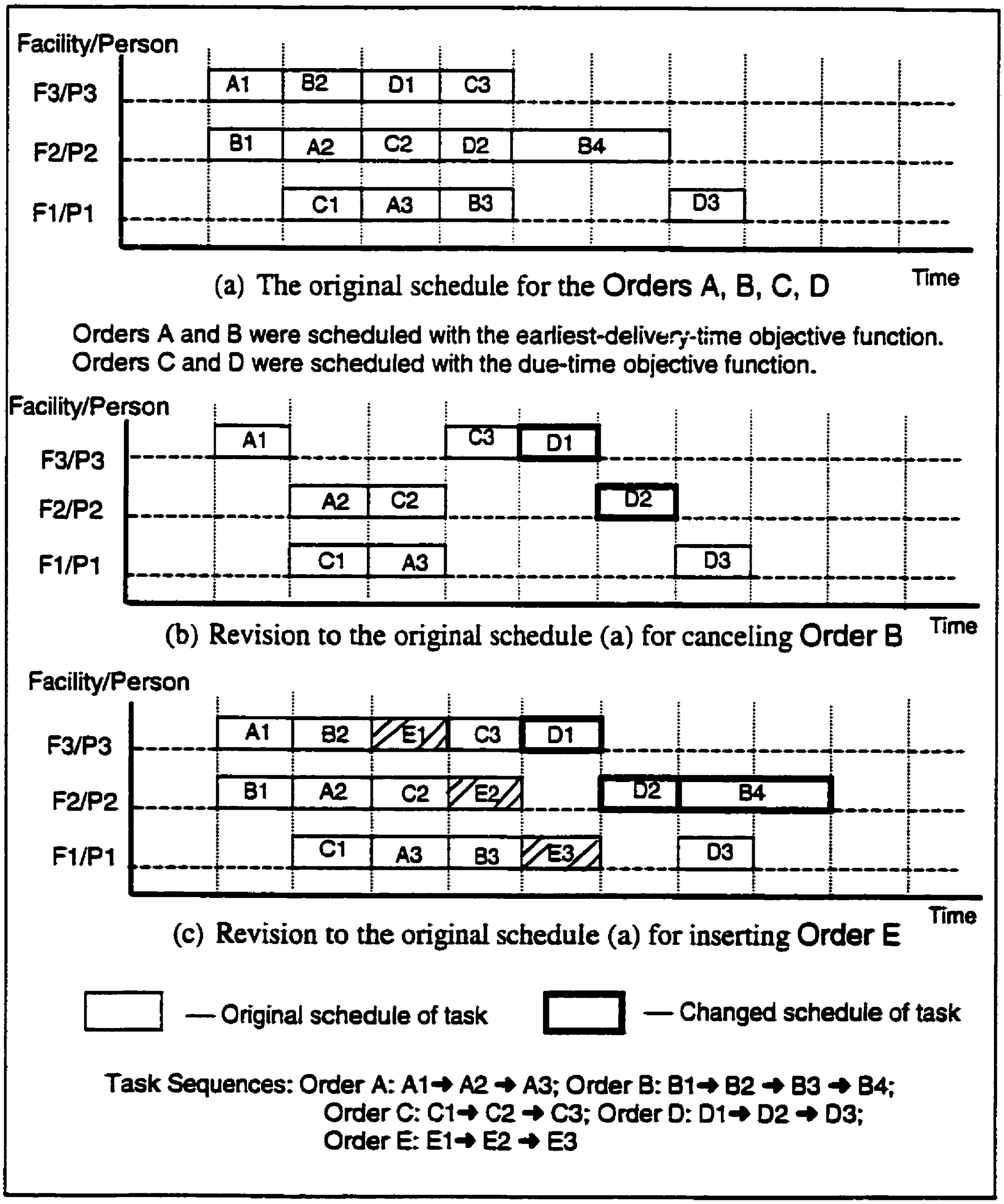
Figure 5.1 (b) and (c) present the revisions to the original schedule (a) using the match-up rescheduling approach. Figure 5.1 (b) illustrates the revised schedule for canceling *Order B*. In this case, only *Task D1* and *Task D2* are shifted to new time slots. Figure 5.1 (c) shows the revised schedule for inserting *Order E* due to its urgent due-time requirement. In this case, the tasks *D1, D2*, and *B4* are reassigned new timing parameters, including start time and finish time, and the other tasks still keep their original timing parameters. It is shown that the original schedule is only partially changed for responding to the production demand changes without rescheduling all the tasks.

In this system, the following strategies are adopted in the match-up rescheduling approach:

(1) In the rescheduling process, the orders scheduled using the due-time based scheduling are revised before the orders scheduled using the earliest-delivery-time based scheduling, since the due-time constraints have to be satisfied.
In the example shown in Figure 5.1, the schedules for *Order C* and *Order D* should be revised before *Order A* and *Order B* in the rescheduling process.

(2) In the revised schedule, the task sequence for each to-be-rescheduled order is kept the same as its task sequence in the original schedule in order to satisfy the task precedence constraints and improve the rescheduling efficiency.

In the example shown in Figure 5.1, the task sequences for *Order A*, *Order B*, *Order C*, and *Order D* in the revised schedule are the same as those in the original schedule: *Order A: A1→ A2 → A3*; *Order B: B1→ B2 → B3 → B4*; *Order C: C1→ C2 → C3*; *Order D: D1→ D2 → D3*.



(a) The original schedule for the Orders A, B, C, D

Orders A and B were scheduled with the earliest-delivery-time objective function.
Orders C and D were scheduled with the due-time objective function.

(b) Revision to the original schedule (a) for canceling Order B

(c) Revision to the original schedule (a) for inserting Order E

Task Sequences: Order A: A1→ A2 → A3; Order B: B1→ B2 → B3 → B4;
Order C: C1→ C2 → C3; Order D: D1→ D2 → D3;
Order E: E1→ E2 → E3

Figure 5.1: Match-up rescheduling for production demand changes

(3) In the revised schedule, each rescheduled task is still allocated with the resources that were originally allocated to this task. The timing parameters of this task are reassigned to improve the overall schedule quality.

In the example shown in Figure 5.1, all the rescheduled tasks, *D1, D2,* and *B4,* are still allocated with the same facilities/persons, *F3/P3, F2/P2,* and *F2/P2,* respectively.

The intelligent production scheduling system conducts the match-up rescheduling approach in a multi-agent environment, as illustrated in Figure 5.2. Upon receiving a message of production demand change from the *Scheduling Browser*, the facility mediator conducts the match-up rescheduling approach based on agent-based negotiation among resource agents and mediators in the resource management module. The algorithm of the proposed match-up rescheduling is shown in Figure 5.3.

An example is given in Figure 5.4 to illustrate the match-up rescheduling algorithm for responding to production demand changes. In this example, an urgent order *E* needs to be inserted in the original schedule as shown in Figure 5.4 (a).

In Step 1, as shown in Figure 5.4 (b), *Order A, Order B, Order C,* and *Order D* are identified as the to-be-rescheduled orders and removed from the original schedule temporally. Then, *Order E* is scheduled using the due-time based scheduling at this stage.

In Step 2, as shown in Figure 5.4 (c), the orders scheduled previously using the due-time based scheduling, *Order C* and *Order D,* are to be rescheduled. The tasks *D1* and *D2* are identified as the to-be-rescheduled tasks at the beginning, due to their conflict with the tasks *E1* and *E2*. Based on the finish time values, *Task D2* is selected as the first current to-be-rescheduled task, *Task D3* that is to be started after the finish time of *Task D2* is recovered in the revised schedule. Then *Task D2* is reassigned the new timing parameters including start time and finish time. Since the timing parameters of *Task D2* are changed in the revised schedule, two tasks are identified as the to-be-rescheduled tasks: *Task C2* that precedes *Task D2* in the original schedule considering facility/person *F2/P2,* and *Task D1* that precedes *Task D2* in the task sequence of the original schedule for *Order D*. Only *Task C2* is added to the to-be-scheduled tasks, since *Task D1* has been already identified as to-be-rescheduled task at the beginning of this step. Next, *Task D1* is selected as the current to-be-rescheduled task and reassigned the new timing parameters. No to-be-rescheduled task is identified from *Task D1*, although the reassigned timing parameters of *Task D1* are different from its original timing parameters. This is because *Task D1* is the first task for *Order D* and no task precedes *Task D1* in the original schedules considering facility/person *F3/P3*. Similarly, the match-up rescheduling procedure is continued until *Task C2*, from which there is no task to be identified as the to-be-rescheduled task, is rescheduled. At the end of Step 2, task *C1*, is recovered with its original schedule.

In Step 3, as shown in Figure 5.4 (c), the orders scheduled previously using the earliest-delivery-time based scheduling, *Order A* and *Order B*, are to be rescheduled. The original schedule of *Task B4* is in conflict with the revised schedule of *Task D2* and thus *Task B4* is identified as the to-be-rescheduled task at the beginning. Next, *Task B4* is selected as the current to-be-rescheduled task and the tasks *A1, A2, A3, B1, B2,* and *B3,* whose finish time values precede the start time of *Task B4* in the original schedule, are recovered in the revised schedule. Then, *Task B4* is reassigned the new timing parameters. *Task B4* is the end task of *Order B* and has no following tasks in the original schedules considering facility/person *F2/P2* to be identified as to-be-rescheduled tasks. Thus the match-up rescheduling process stops after parameters of *Task B4* are assigned.

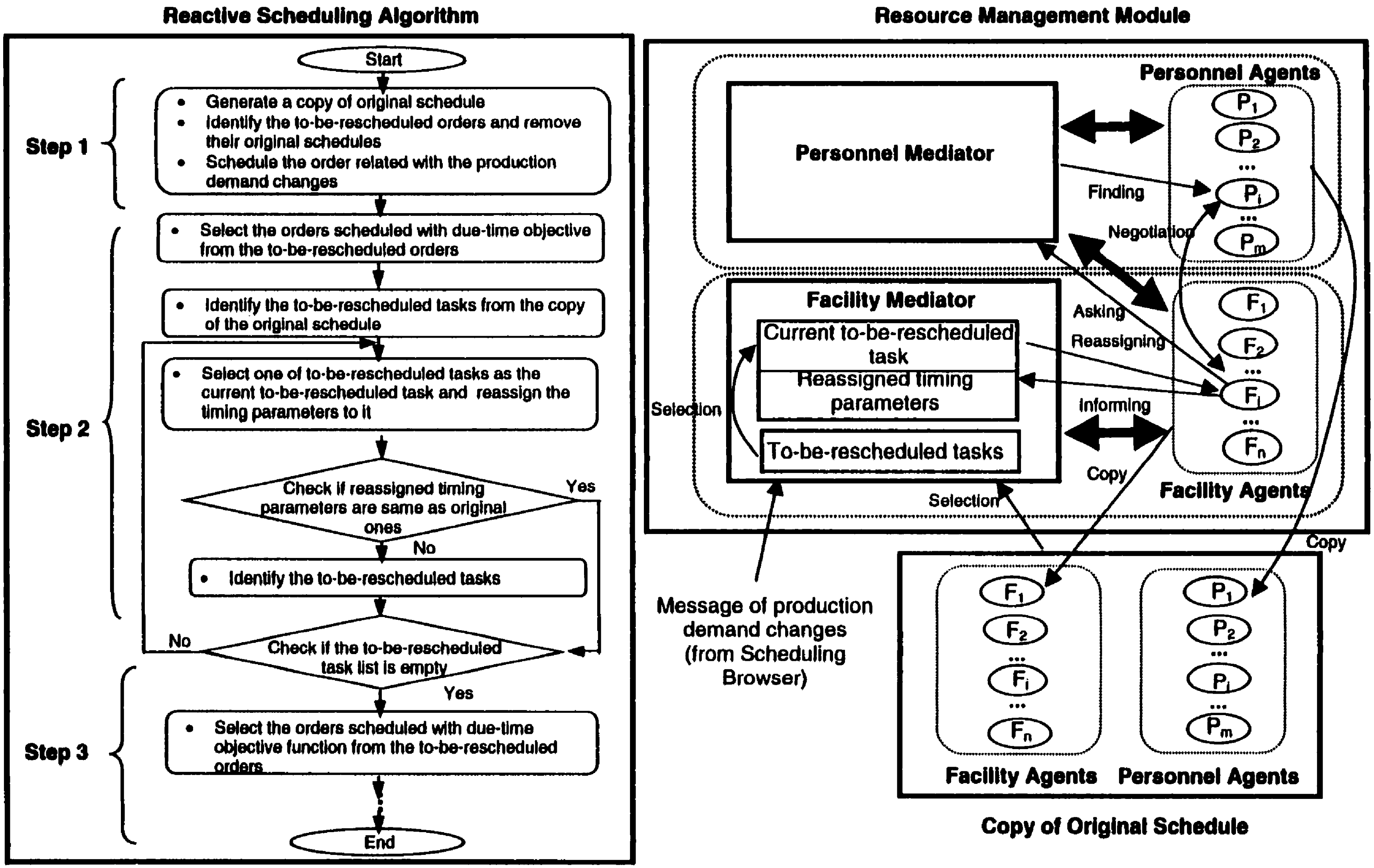

Reactive Scheduling Algorithm
Start
Step 1
Generate a copy of original schedule
Identify the to-be-rescheduled orders and remove their original schedules
Schedule the order related with the production demand changes
Step 2
Select the orders scheduled with due-time objective from the to-be-rescheduled orders
Identify the to-be-rescheduled tasks from the copy of the original schedule
Select one of to-be-rescheduled tasks as the current to-be-rescheduled task and reassign the timing parameters to it
Check if reassigned timing parameters are same as original ones
Yes
No
Identify the to-be-rescheduled tasks
Check if the to-be-rescheduled task list is empty
No
Yes
Step 3
Select the orders scheduled with due-time objective function from the to-be-rescheduled orders
End
Resource Management Module
Personnel Agents
Personnel Mediator
P_1
P_2
P_i
P_m
Finding
Negotiation
Facility Mediator
Current to-be-rescheduled task
Reassigned timing parameters
To-be-rescheduled tasks
Asking
Reassigning
Informing
Copy
Selection
Selection
F_1
F_2
F_i
F_n
Facility Agents
Copy
Message of production demand changes (from Scheduling Browser)
Facility Agents
Personnel Agents
Copy of Original Schedule

Figure 5.2: Reactive scheduling approach for responding to production demand changes

Step 1. Initialize for rescheduling

1.1 Generate a copy of the original schedules that are preserved in resource agents

1.2 Identify the to-be-rescheduled orders and remove their original schedules from the resource agents

The to-be-rescheduled orders are all the orders that have not been manufactured until the current time.

1.3 Schedule the order related to the production demand change

- In case of canceling an order, the to-be-cancelled order should not considered in rescheduling.
- In case of inserting an order, the to-be-inserted order is scheduled first using the due-time based scheduling approach.

Step 2. Reschedule the to-be-rescheduled orders that were previously scheduled using due-time based scheduling

2.1 Select the to-be-rescheduled orders that were previously scheduled using due-time based scheduling from the whole to-be-rescheduled orders

2.2 Identify the to-be-rescheduled tasks from the copy of the original schedules and sort the list of to-be-rescheduled tasks according to the finish time values of these tasks. The to-be-rescheduled task with largest finish time value is placed at the beginning of the list.

The to-be-rescheduled tasks are the tasks whose original schedules need to be revised.

- In case of canceling an order, the tasks that precede the tasks of the to-be-cancelled order in the original schedules should be identified as the to-be-rescheduled tasks at this stage.
- In case of inserting an order, the tasks whose original schedules are in conflict with the schedule of the inserted order should be identified as the to-be-rescheduled tasks at this stage.

2.3 Select first element of the to-be-rescheduled task list as the current to-be-rescheduled task, recover schedules of the tasks that

- start after the finish time of the current rescheduled task, and
- are preserved in the copy of the original schedules, and
- have never been rescheduled.

Reassign the parameters to the current to-be-rescheduled task using agent-based negotiation mechanism (to be introduced in Section 5.2.2)

The current to-be-rescheduled task should be removed from the list of to-be-rescheduled tasks.

2.4 Check if the reassigned timing parameters are same as those in the copy of the original schedules

If not, the following tasks belonging to the to-be-rescheduled orders should be added to the list of to-be-rescheduled tasks:

- the tasks preceding the current to-be-rescheduled task in the copy of the original schedules preserved in the relevant facility agent and personnel agent that were allocated for the current to-be-rescheduled task
- the tasks preceding the current to-be-rescheduled task in the task sequence of the corresponding order

Figure 5.3: The algorithm of match-up rescheduling approach for responding to production demand changes

- the tasks whose original schedules are in conflict with the revised schedule for the current to-be-rescheduled task

2.5 Check if the list of the to-be-rescheduled tasks is empty
If the list is not empty, go to Step 2.3.

2.6 Recover all the tasks that are preserved in the copy of the original schedules and have not been rescheduled so far in the rescheduling process

Step 3. Reschedule the to-be-rescheduled orders that were previously scheduled using the earliest-delivery-time based scheduling

3.1 Select the to-be-rescheduled orders that were previously scheduled using the earliest-delivery-time based scheduling from the whole to-be-rescheduled orders

3.2 Identify the to-be-rescheduled tasks from the copy of original schedules and sort the list of to-be-rescheduled tasks according to the start time values of these tasks. The to-be-rescheduled task with the smallest start time value is placed at the beginning of the list.

The to-be-rescheduled tasks are the tasks whose original schedules need to be revised. At this stage, the tasks whose original schedules are in conflict with the revised schedules should be identified as the to-be-rescheduled task.

3.3 Select first element of the to-be-rescheduled task list as the current to-be-rescheduled task, and recover the tasks that

- completed before the start time of the current to-be-rescheduled task, and
- are preserved in the copy of the original schedules, and
- have never been rescheduled.

Reassign parameters to the current to-be-rescheduled task using agent-based negotiation mechanism (to be introduced in Section 5.2.2)

The current to-be-rescheduled task should be removed from the list of the to-be-rescheduled tasks.

3.4 Check if the reassigned timing parameters are same as those in the copy of original schedules

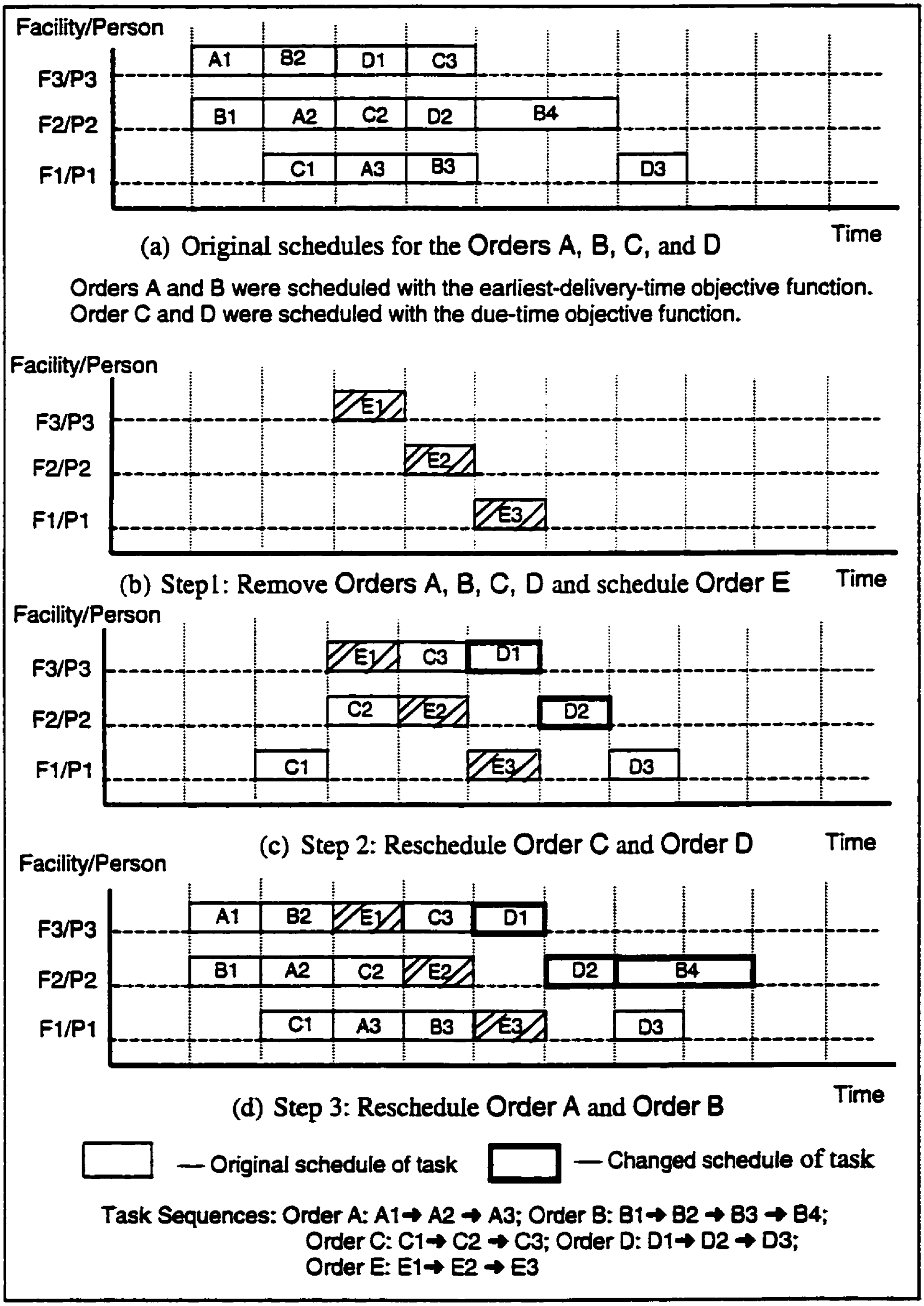
If not, the following tasks belonging to the to-be-rescheduled orders should be added to the list of to-be-rescheduled tasks:

- the tasks following the current to-be-rescheduled task in the copy of the original schedules preserved in the relevant facility agent and personnel agent that were allocated for the current to-be-rescheduled task
- the tasks following the current to-be-rescheduled task in the task sequence of the corresponding order
- the tasks whose original schedules are in conflict with the revised schedule for the current to-be-rescheduled task

3.5 Check if the list of the to-be-rescheduled tasks is empty
If the list is not empty, go to Step 3.3.

3.6 Recover all the tasks that are preserved in the copy of the original schedules and have not been rescheduled so far in the rescheduling process

Figure 5.3: The algorithm of match-up rescheduling approach for responding to production demand changes (continued)



(a) Original schedules for the Orders A, B, C, and D

Orders A and B were scheduled with the earliest-delivery-time objective function. Order C and D were scheduled with the due-time objective function.

(b) Step1: Remove Orders A, B, C, D and schedule Order E

(c) Step 2: Reschedule Order C and Order D

(d) Step 3: Reschedule Order A and Order B

Task Sequences: Order A: A1→ A2 → A3; Order B: B1→ B2 → B3 → B4;
Order C: C1→ C2 → C3; Order D: D1→ D2 → D3;
Order E: E1→ E2 → E3

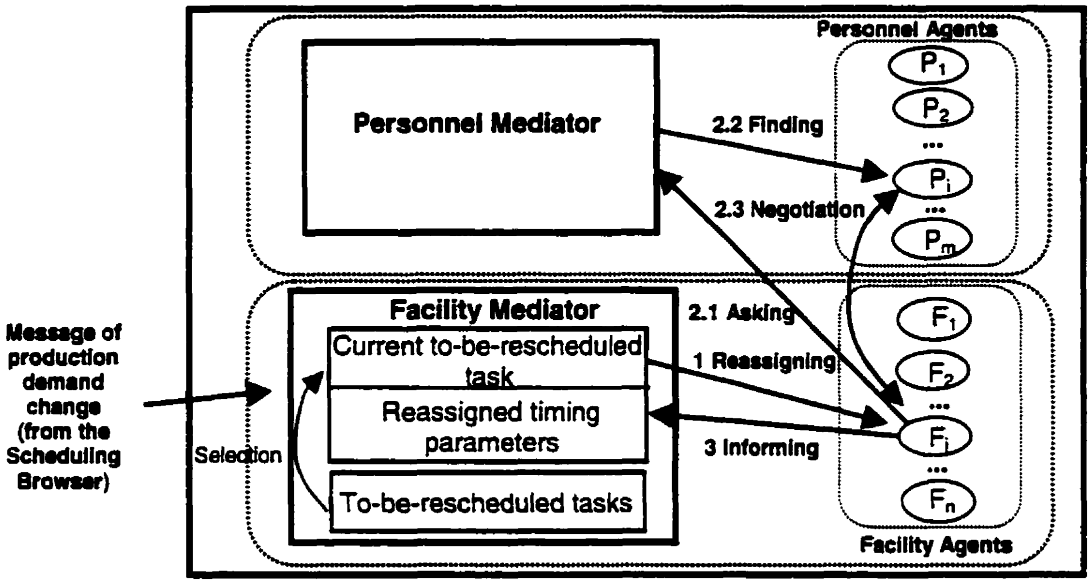
Figure 5.4: An example of match-up rescheduling for production demand changes

5.2.2 Reassigning Task Timing Parameters Using Agent-based Negotiation

In the rescheduling process, as introduced in Section 5.2.1, reassigning timing parameters to the to-be-rescheduled tasks is conducted based upon the agent-based negotiation among resource agents and mediators in the resource management module. Two task timing parameters, start time and finish time, are considered in this research. An agent-based negotiation mechanism is proposed for reassigning timing parameters to the to-be-rescheduled tasks, as shown in Figure 5.5.

An example is shown in Figure 5.6 to illustrate the procedure of reassigning timing parameters to a to-be-rescheduled task in rescheduling orders with the earliest-delivery-time objective function. During the rescheduling process, the current to-be-rescheduled task *A* needs to be reassigned new timing parameters. The resources, facility agent *F1* and personnel agent *P1*, were originally allocated to this task. The earliest possible start time is assumed to be T_{es} that has been determined in terms of the task precedence constraints and feature relations using the method introduced in Section 4.2.2. First, the facility mediator reassigns this task to the facility agent *F1*. Upon receiving this message, the facility agent *F1* finds the related personnel agent *P1* through the personnel mediator. Then the facility agent *F1* negotiates with the personnel agent *P1* to determine the proper time slot for the task *A*. The time slot should provide the minimum start time, while satisfying all the manufacturing requirements and constraints.

An example is shown in Figure 5.7 to illustrate reassigning a task in rescheduling the orders with the due-time objective function. During the rescheduling process, a current to-be-rescheduled task *B* needs to be reassigned with new timing parameters. The resources, facility agent *F1* and personnel agent *P1*, were originally allocated to this task. The latest possible finish time is assumed to be T_{lf} that has been determined by the task precedence constraints and feature relations using the method introduced in Section 4.2.2. Firstly, the facility mediator reassigns this task to the facility agent *F1*. Upon receiving this message, the facility agent *F1* finds the personnel agent *P1* through the personnel mediator. Then the facility agent *F1* negotiates with the personnel agent *P1* to determine the proper time slot for this task. The time slot should provide the maximum finish time, while satisfying all the manufacturing requirements and constraints.



Step 1. The facility mediator reassigns the current to-be-rescheduled task to the facility agent that was originally allocated to this task.

Before reassigning the current to-be-rescheduled task, the facility mediator identifies the following time value, using the method introduced in Section 4.4.2.

- For the orders scheduled with the earliest-delivery-time objective function, the earliest possible start time of the current to-be-rescheduled task is calculated as the latest finish time of all the rescheduled tasks for this feature. If this task is the first one in an instance feature to be considered in rescheduling, the earliest possible start time is calculated as the latest finish time considering all the tasks in the element features.
- For the orders scheduled with the due-time objective function, the latest possible finish time of the current to-be-rescheduled task is calculated as the earliest start time of all the rescheduled tasks for this feature. If this task is the first one in an instance feature to be considered in rescheduling, the latest possible finish time is calculated as the earliest start time considering all the tasks in the super features.

Step 2. The facility agent negotiates with the personnel agent that was originally assigned to the current to-be-rescheduled task in order to identify new timing parameters to the current to-be-rescheduled task.

2.1 The facility agent asks the personnel mediator to identify its relevant personnel agent.
2.2 The personnel mediator finds the relevant personnel agent.
2.3 The facility agent communicates with the personnel agent to identify new timing parameters to the current to-be-rescheduled task.

Step 3. The facility agent informs the facility mediator with the new timing parameters of the current to-be-rescheduled task.

Figure 5.5: The negotiation mechanism for reassigning task timing parameters

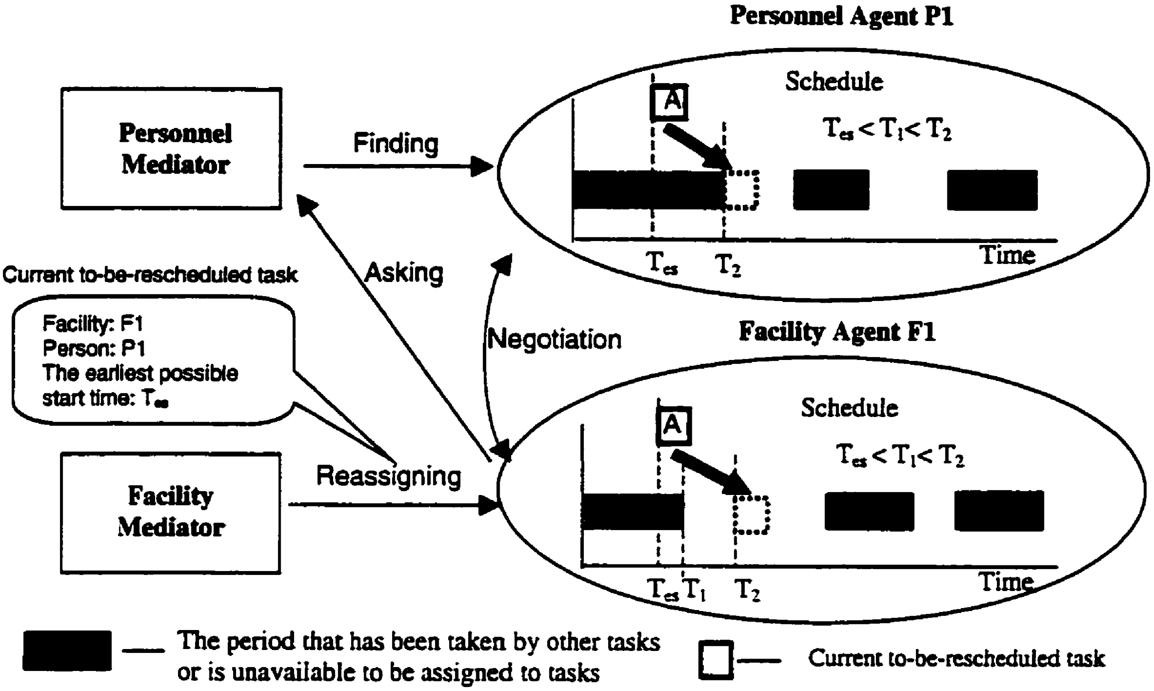


Figure 5.6: An example of reassigning a task to resource agents in rescheduling the orders with earliest-delivery-time objective function

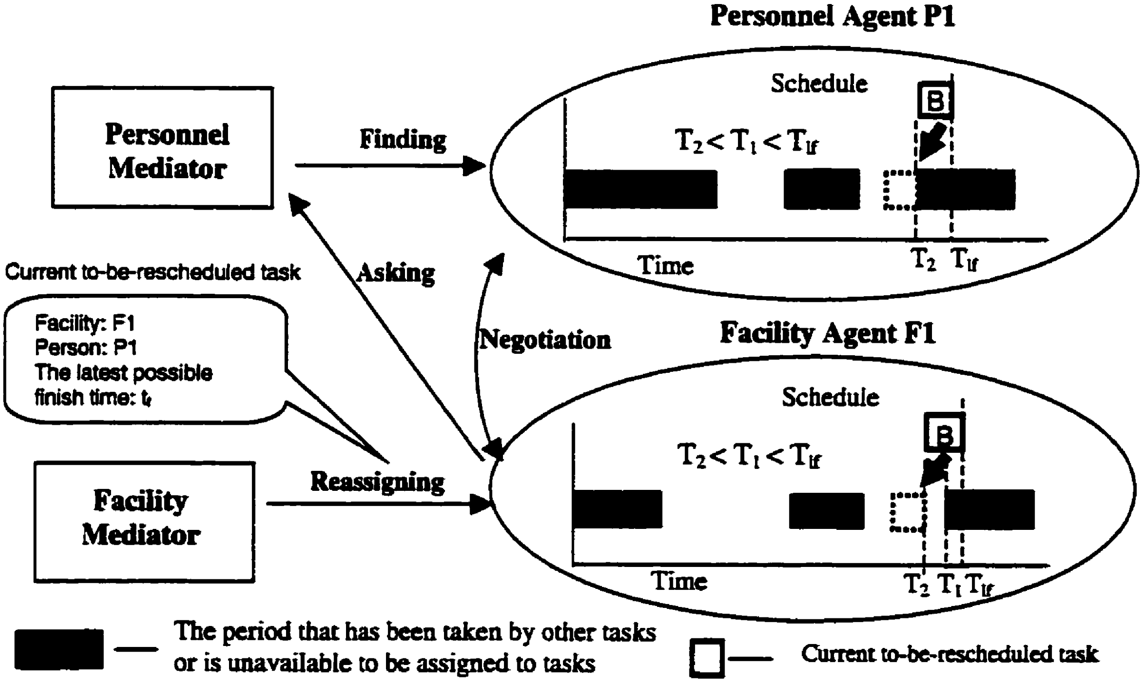


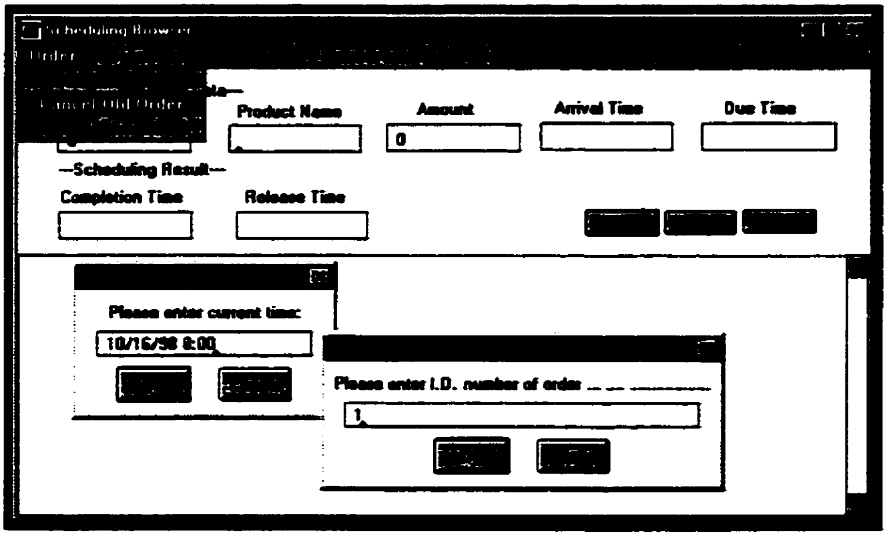
Figure 5.7: An example of reassigning a task to resource agents in rescheduling the orders with due-time objective function

5.2.3 Case Study I: Canceling an Old Order

In this section, an extended example is introduced to illustrate the reactive scheduling function of this system for responding to a production demand change — canceling an old order.

The resource management module used in this example is the same as the one introduced in Section 4.3. Five production orders, *Order 1*, *Order 2*, *Order 3*, *Order 4*, and *Order 5*, have been scheduled previously. Each order needs to product a window product that is represented by an instance feature, *c*, of class feature *WindowCenter* as introduced in Section 4.2.2. *Order 1*, *Order 2*, *Order 3*, and *Order 4* were scheduled using the due-time based scheduling. *Order 5* was scheduled using the earliest-delivery-time based scheduling.

At the current time, *10/16/98 8:00*, one of the scheduled orders, *Order 1*, is canceled based upon the customer's request. *Scheduling Browser* is used as the user interface to conduct the reactive scheduling for canceling old orders, as shown in Figure 5.8. Upon selecting the *Cancel Old Order* function from the *Order* menu, user indicates the *current time* and the *I.D. number of order* to be canceled using dialog boxes. The current time is used for identifying the orders to be rescheduled in the reactive



Product Name:
Amount: 0
Arrival Time:
Due Time:

—Scheduling Result—

Completion Time:
Release Time:

Please enter current time: 10/16/98 8:00

Please enter I.D. number of order: 1

Figure 5.8: The scheduling browser as used for reactive scheduling to cancel an old order

scheduling. If execution of an order has been completed at the current time, this order should not be rescheduled.

Figure 5.9 shows the original schedules for *Order 1*, *Order 2*, *Order 3*, *Order 4*, and *Order 5*. Figure 5.10 shows the revised schedules for canceling *Order 1*. Comparing the original schedules and the revised schedules, it is found that only partial timing parameters are changed for canceling *Order 1*. The task schedules for *Order 2*, *Order 3*, and *Order 4*, are shifted forward to new time slots, while part of the task schedules for *Order 5* are shifted backward to new time slots. The comparison between the original schedules and the revised schedules is shown in Table 5.1.

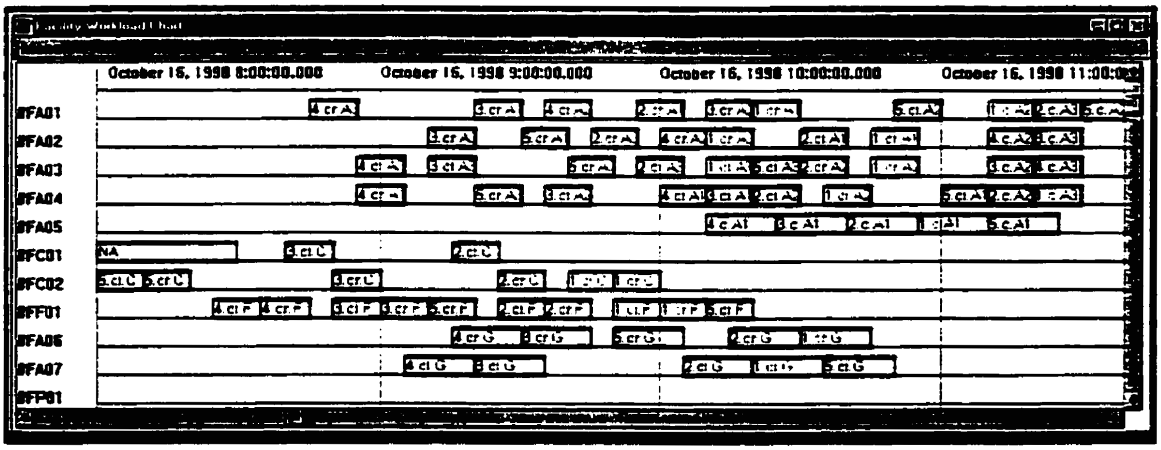

October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00:00
#FA01
#FA02
#FA03
#FA04
#FA05
#FC01
#FC02
#FF01
#FA06
#FA07
#FP01

Figure 5.9: The original schedules for *Orders 1, 2, 3, 4,* and *5*

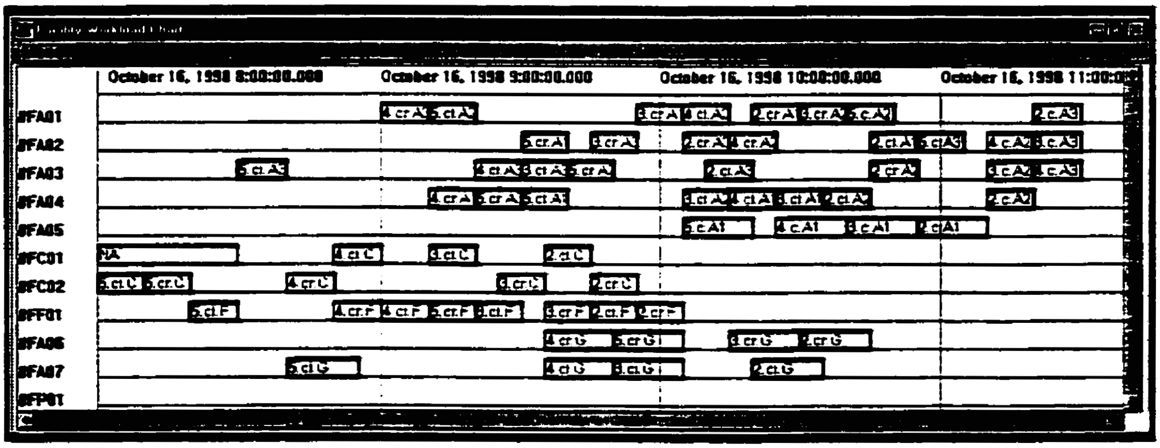

October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00:00
#FA01
#FA02
#FA03
#FA04
#FA05
#FC01
#FC02
#FF01
#FA06
#FA07
#FP01

Figure 5.10: The revised schedules for canceling *Order 1*

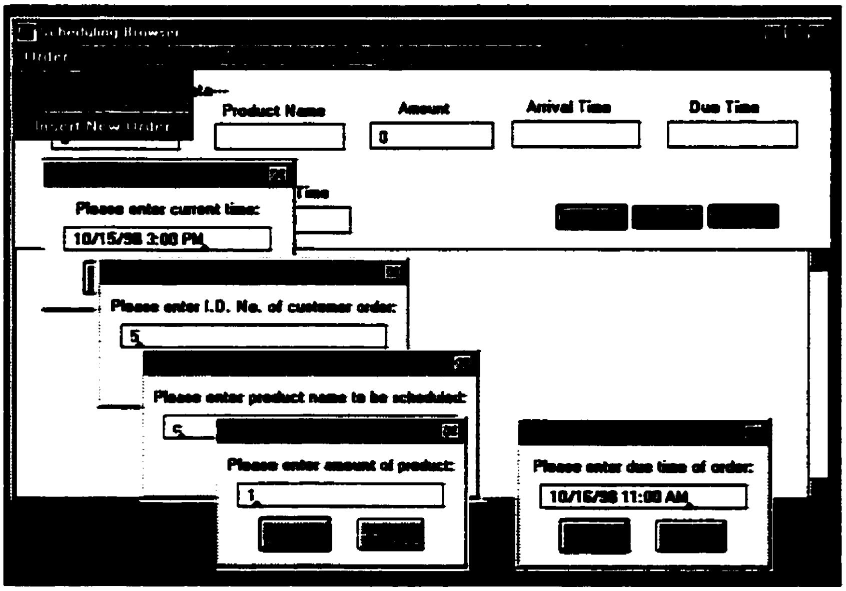
Table 5.1: The comparison between the original schedules and the revised schedules

Order	Arrival Time	Due Time	Original Schedule		Revised Schedule	
			Release	Completion	Release	Completion
1	10/15 8:00	10/16 11:30	10/16 9:40	10/16 11:30		
2	10/15 8:05	10/16 11:30	10/16 9:15	10/16 11:30	10/16 9:40	10/16 11:30
3	10/15 8:10	10/16 11:30	10/16 8:40	10/16 11:30	10/16 9:10	10/16 11:30
4	10/15 8:15	10/16 11:30	10/16 8:25	10/16 11:30	10/16 8:40	10/16 11:30
5	10/15 8:20		10/16 8:00	10/16 11:40	10/16 8:00	10/16 11:05

5.2.4 Case Study II: Inserting an Urgent Order

In this section, an extended example is used to illustrate the reactive scheduling function of this system for responding to an production demand change — inserting a new order, which cannot be scheduled using predictive scheduling due to the urgent due date requirement.

The resource management module used in this example is as the same as the one introduced in Section 4.3. Four production orders, *Order 1*, *Order 2*, *Order 3*, and



Amount: 0

Please enter current time:
10/15/98 3:00 PM

Please enter I.D. No. of customer order:
5

Please enter product name to be scheduled:
5

Please enter amount of product:
1

Please enter due time of order:
10/16/98 11:00 AM

Figure 5.11: The scheduling browser as used for reactive scheduling to insert an urgent order

Order 4, have been scheduled previously. Each order needs to produce a window product that is represented by an instance feature, *c*, of class feature *WindowCenter*, as introduced in Section 4.2.2. *Order 1*, *Order 2*, and *Order 3* were scheduled by the due-time based scheduling. *Order 4* was scheduled by the earliest-delivery-time based scheduling.

At the current time, *10/15/98 3:00 PM*, *Order 5* needs to be inserted with an urgent due-time requirement. *Scheduling Browser* is used as the user interface to conduct the reactive scheduling for inserting the new order, as shown in Figure 5.11. Upon selecting the *Insert New Order* function from the *Order* menu, user enters the *current time*, *I.D. number of order* to be inserted, *product name*, *amount of product*, and *due time of order* using the dialog boxes.

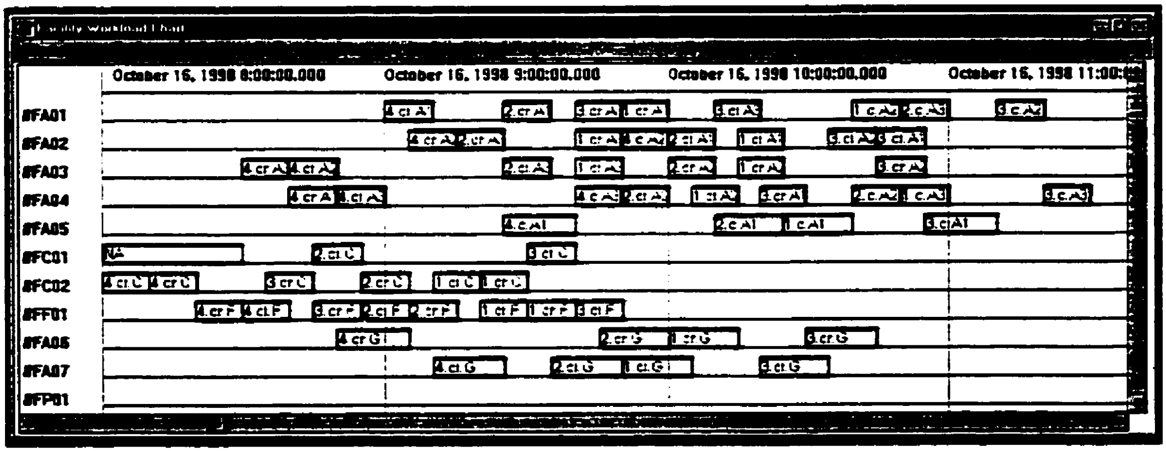

October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00:
#FA01
#FA02
#FA03
#FA04
#FA05
#FC01
#FC02
#FF01
#FA06
#FA07
#FP01

Figure 5.12: The original schedules for *Orders 1, 2, 3,* and *4*

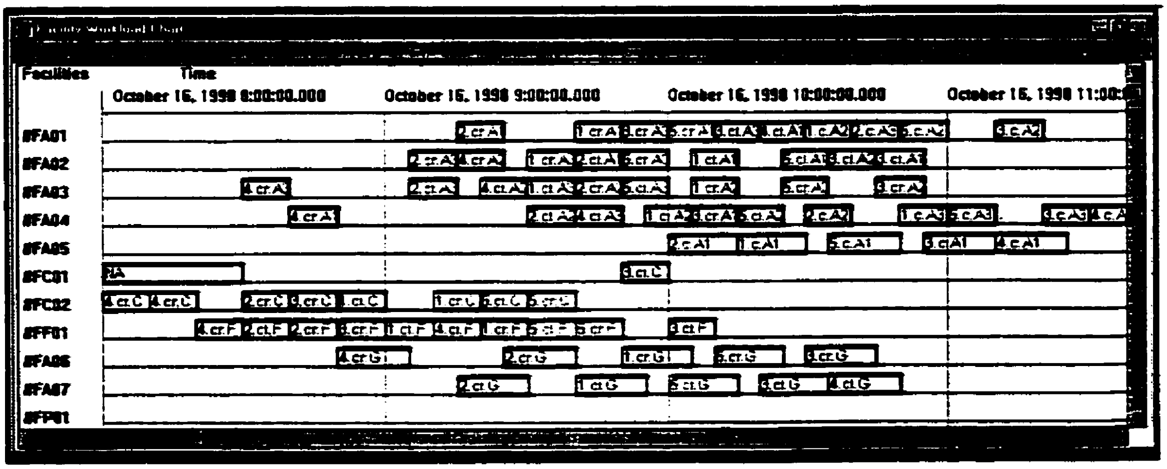


Figure 5.13: The revised schedules for inserting *Order 5*

Figure 5.12 shows the original schedules for *Order 1*, *Order 2*, *Order 3*, and *Order 4*. Figure 5.13 shows the revised schedules for inserting *Order 5*. Comparing the original schedules and the revised schedules, it is found that only partial timing parameters are changed for inserting *Order 5*. The task schedules for *Order 1*, *Order 2*, and *Order 3* are partially shifted forward to new time slots, while the task schedules for *Order 4* are partially moved backward to new time slots. The comparison between the original schedules and the revised schedules is shown in Table 5.2.

Table 5.2: The comparison between the original schedules and the revised schedules

Order	**Arrival Time**	**Due Time**	**Original Schedule**		**Revised Schedule**	
			Release	**Completion**	**Release**	**Completion**
1	10/15 8:00	10/16 11:00	10/16 9:10	10/16 11:00	10/16 9:00	10/16 11:00
2	10/15 8:05	10/16 11:00	10/16 8:45	10/16 11:00	10/16 8:30	10/16 11:00
3	10/15 8:10	10/16 11:30	10/16 8:50	10/16 11:30	10/16 9:10	10/16 11:30
4	10/15 8:20		10/16 8:00	10/16 10:00	10/16 8:00	10/16 11:40
5	10/15 15:00	10/16 11:10			10/16 9:20	10/16 11:10

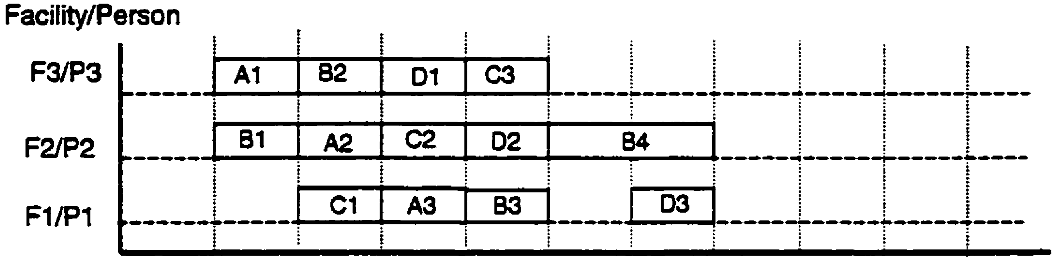
5.3 Reactive Scheduling for Resource Changes

An intelligent reactive scheduling mechanism was proposed in this research for responding to resource changes. Normally, the original schedule is executed during the production process in the job-shop. But, if unexpected resource changes happen, the intelligent production scheduling system should revise the original schedules for responding to those changes. The two cases, (1) facility breakdowns and (2) personnel absence, are considered as resource changes in this research. The intelligent reactive scheduling mechanism will be described in the following two aspects: (1) revising schedule using the match-up rescheduling approach, and (2) reassigning timing parameters and reallocating resources using agent-based negotiation.

5.3.1 Revising Schedules Using Match-up Rescheduling Approach

A simple scenario, as shown in Figure 5.14, is used to illustrate the match-up rescheduling approach for responding to the resource changes. Two kinds of orders are presented in the original schedule.

- The orders *A* and *B* were scheduled using earliest-delivery-time based scheduling.
- The orders *C* and *D* were scheduled using due-time based scheduling.



Original schedules for the orders A, B, C, D Time

Orders A and B were scheduled with the earliest-delivery-time objective function.
Orders C and D were scheduled with the due-time objective function.

Task Sequences: Order A: A1→ A2 → A3; Order B: B1→ B2 → B3 → B4;
Order C: C1→ C2 → C3; Order D: D1→ D2 → D3;

Figure 5.14: A scenario for illustrating match-up rescheduling for resource changes

The following strategies are adopted in the match-up rescheduling approach:

(1) At the beginning of rescheduling, the tasks that are disrupted directly by the resource changes will be moved to other resources that are able to be used as substitutes to take over these tasks in the same time slots.

In an example shown in Figure 5.15, the person *P4* can take over the tasks *A2* and *C2* during the person *P2's* absence.

(2) If substitute resources can not be found for the disrupted tasks, the rescheduling process should be continued. In this strategy, the orders scheduled using due-time based scheduling are revised before the orders scheduled using earliest-delivery-time based scheduling, since due-time constraints should be satisfied.

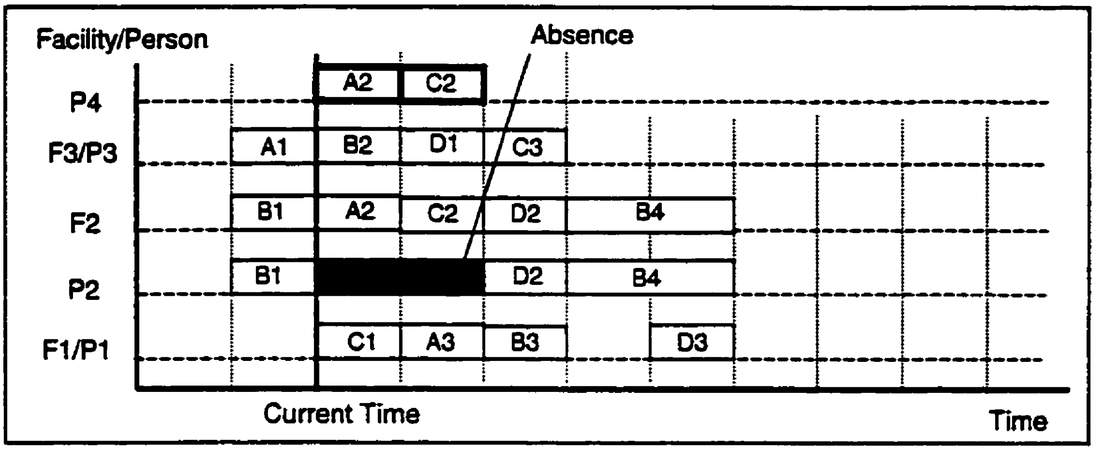


Figure 5.15: An example to illustrate the rescheduling strategy (1) for resource changes

In the example shown in Figure 5.16, the schedules for *Order C* and *Order D* are revised before *Order A* and *Order B* in rescheduling process.

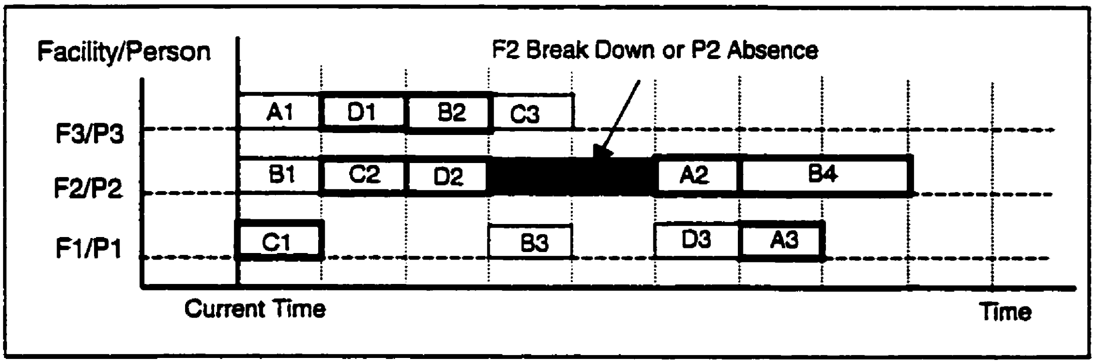
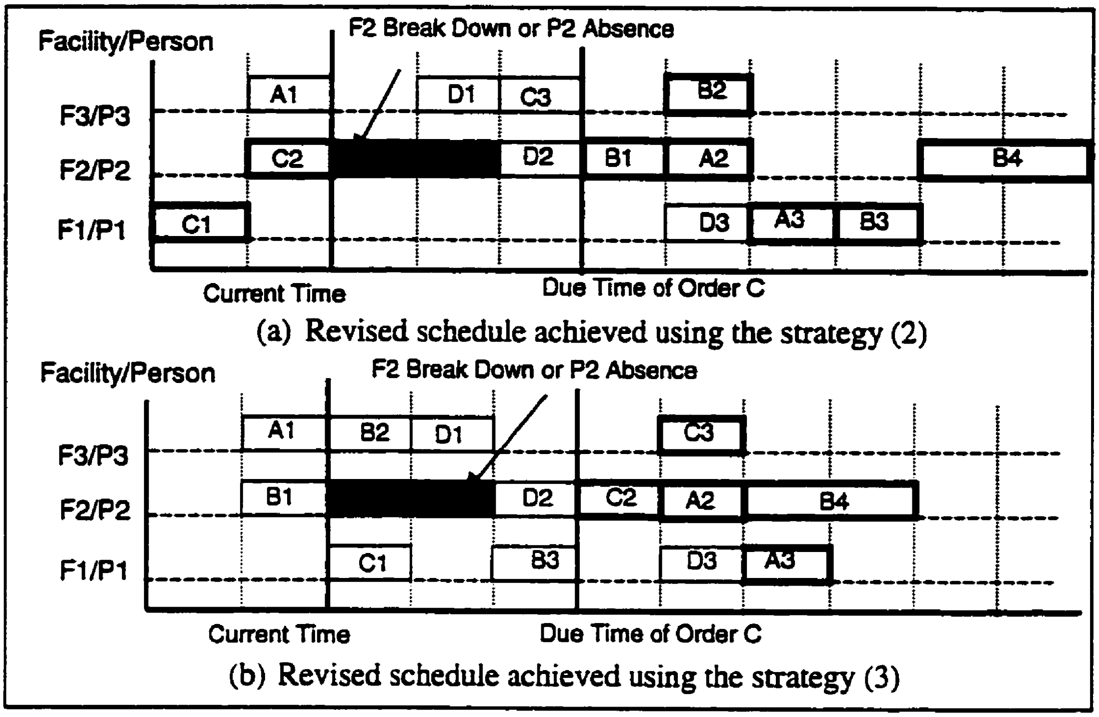


Figure 5.16: An example to illustrate the rescheduling strategy (2) for resource changes

(3) If the no feasible revised schedule can be achieved using the strategy (2), the affected orders will be rescheduled with modified due-dates after all other orders with due-time requirements have been rescheduled.

In the example shown in Figure 5.17, it is impossible to reschedule *Order C* in the period from the current time to the due-time using the strategy (2), as shown in Figure 5.17 (a). To solve this problem, the schedule is achieved using strategy (3), as shown in Figure 5.17 (b). This schedule is feasible to carry out *Order C*, although it cannot satisfy the due time requirement.

(4) The task sequence for each to-be-rescheduled order in the revised schedule is kept the same as that in the original schedule in order to satisfy the design



(a) Revised schedule achieved using the strategy (2)

(b) Revised schedule achieved using the strategy (3)

Figure 5.17: An example to illustrate the rescheduling strategy (3) for resource changes

constraints and the task precedence constraints, since these constraints have been considered in the original schedule.

For example, the revised schedules in Figure 5.15, Figure 5.16, and Figure 5.17 for *Order A*, *Order B*, *Order C*, and *Order D* still keep the task sequences the same as those in the original schedules: *Order A: A1* → *A2* → *A3*; *Order B: B1* → *B2* → *B3* → *B4*; *Order C: C1* → *C2* → *C3*; *Order D: D1* → *D2* → *D3*.

(5) In the revised schedule, if substitute resources can not be found for the disrupted tasks, each rescheduled task is still allocated with the resources that were originally assigned to this task for improving the rescheduling efficiency.

In the examples shown in Figure 5.16 and Figure 5.17, all the rescheduled tasks are still allocated with the same facilities/persons, i.e., the tasks *B3*, *A1*, *A2*, and *C1* are kept on the same facilities/persons *F3/P3*, *F1/P1*, *F2/P2*, and *F1/P1*, respectively.

The proposed match-up rescheduling approach is implemented in a multi-agent environment as illustrated in Figure 5.18.

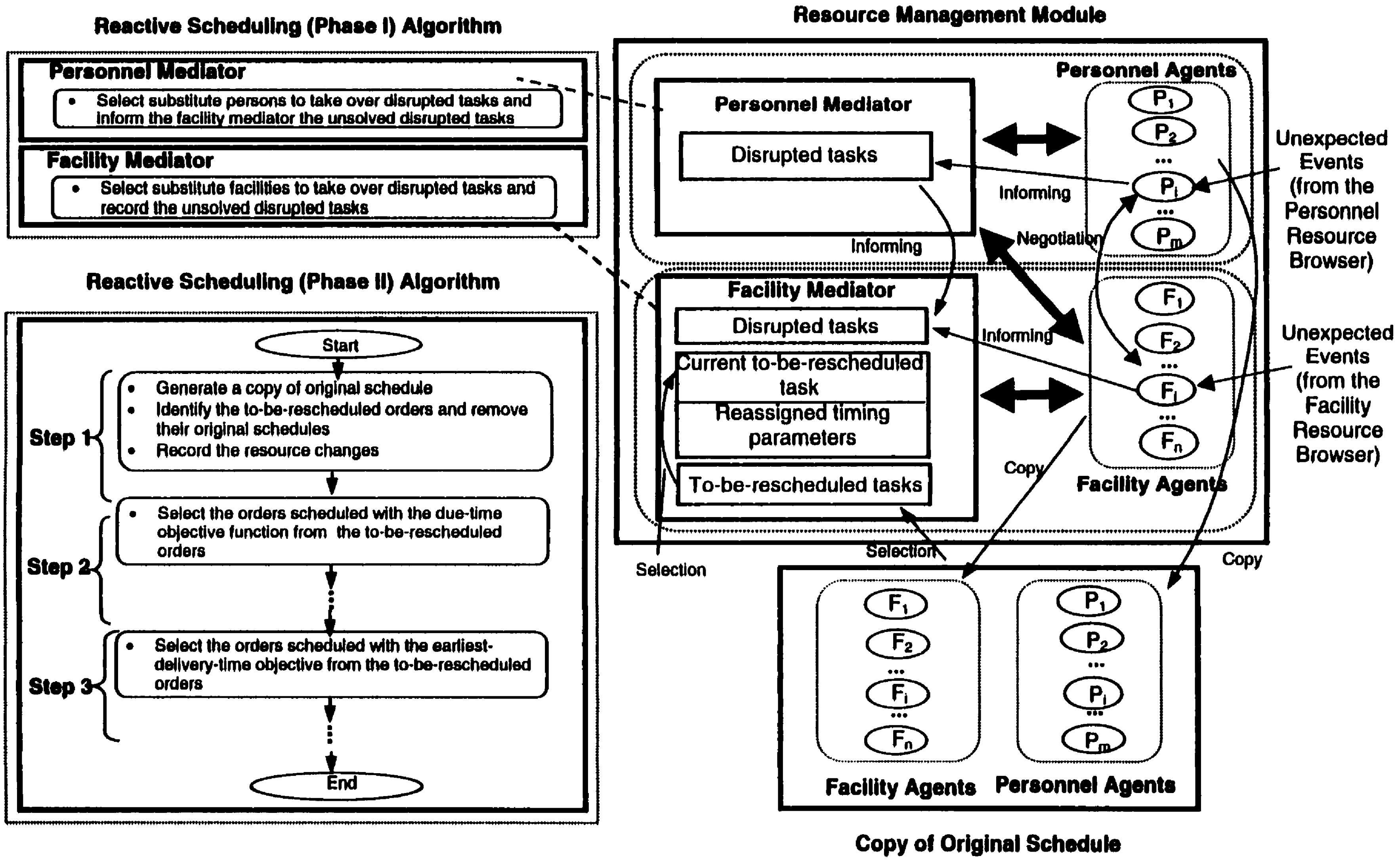

Reactive Scheduling (Phase I) Algorithm
Personnel Mediator
• Select substitute persons to take over disrupted tasks and inform the facility mediator the unsolved disrupted tasks
Facility Mediator
• Select substitute facilities to take over disrupted tasks and record the unsolved disrupted tasks
Reactive Scheduling (Phase II) Algorithm
Start
Step 1
• Generate a copy of original schedule
• Identify the to-be-rescheduled orders and remove their original schedules
• Record the resource changes
Step 2
• Select the orders scheduled with the due-time objective function from the to-be-rescheduled orders
Step 3
• Select the orders scheduled with the earliest-delivery-time objective from the to-be-rescheduled orders
End
Resource Management Module
Personnel Agents
Personnel Mediator
Disrupted tasks
Informing
Negotiation
Informing
P_1
P_2
P_i
P_m
Unexpected Events (from the Personnel Resource Browser)
Facility Mediator
Disrupted tasks
Current to-be-rescheduled task
Reassigned timing parameters
To-be-rescheduled tasks
Informing
F_1
F_2
F_i
F_n
Unexpected Events (from the Facility Resource Browser)
Facility Agents
Copy
Selection
Selection
Copy
Facility Agents
Personnel Agents
Copy of Original Schedule

Figure 5.18: Reactive scheduling approach for responding to resource changes

The algorithm of the proposed match-up rescheduling approach consists of two phases. In the first phase, when the system receives messages about resource changes, such as facility breakdowns or personnel absence, the system identifies the disrupted tasks, the tasks whose schedules are disrupted by the resource changes. After that, the personnel mediator and the facility mediator try to find out substitute persons and facilities to take over the disrupted tasks in the same time slots using the bidding mechanism introduced in Section 4.4.2. The algorithm in the first phase is shown in Figure 5.19.

```
<In Personnel Mediator>
    1  Receive messages about disrupted tasks which are identified by personnel agents
    2  Find out the substitute persons to take over the disrupted tasks without changing their original timing parameters using bidding mechanism, introduced in Section 4.4.2
    3  Inform the facility mediator about the disrupted tasks that can not be reassigned with the same timing parameters as those in the original schedule
<In Facility Mediator>
    1  Receive messages about disrupted tasks which are identified by facility agents
    2  Find out the substitute facilities to take over the disrupted tasks without changing their original timing parameters using bidding mechanism introduced in Section 4.4.2
    3  Record the disrupted tasks that can not be reassigned with the same timing parameters as those in the original schedule
```

Figure 5.19: The algorithm in the first phase of match-up rescheduling approach for resource changes

In the second phase, there are two strategies used by facility mediator for the remaining disrupted tasks that cannot be allocated with substitute persons and facilities without changing their original timing parameters.

- The first strategy — the orders scheduled using due-time based scheduling are revised before the orders scheduled using earliest-delivery-time based scheduling.

- The second strategy — If the no feasible revised schedule can be achieved using the first strategy, the affected orders will be rescheduled with modified due-dates after all other orders with due-time requirements have been rescheduled. The orders with earliest-delivery-time based scheduling requirements are considered at last.

In the rescheduling process, the user should select the first strategy at the beginning. If the revised schedule cannot satisfy the manufacturing constraints, the user should select the second strategy for rescheduling. The rescheduling algorithm in the second phase is shown in Figure 5.20.

Step 1. Initialize for rescheduling

1.1 Generate a copy of the original schedules that are preserved in resource agents

1.2 Identify the to-be-rescheduled orders and remove their original schedules from the facility agents and personnel agents

The to-be-rescheduled orders are all the orders that have not been manufactured until the current time.

1.3 Record the resource changes

Identify the time slots in which facility break-downs and person absence happen

Step 2. Reschedule the to-be-rescheduled orders that were previously scheduled using the due-time based scheduling

2.1 Select the to-be-rescheduled orders that were previously scheduled using the due-time based scheduling from the whole to-be-rescheduled orders

- When the first rescheduling strategy is employed, the orders that are directly affected by the resource changes are part of all the to-be-rescheduled orders.
- When the second rescheduling strategy is employed, the orders that are directly affected by the resource changes are not part of all the to-be-rescheduled orders at this stage.

2.2 Identify the to-be-rescheduled tasks from the copy of the original schedules and sort the list of to-be-rescheduled tasks according to the finish time values of these tasks. The to-be-rescheduled task with the largest finish time value is placed at the beginning of the list.

Figure 5.20: The algorithm in the second phase of match-up rescheduling approach for resource changes

The to-be-rescheduled tasks are the tasks whose original schedules need to be revised.

At this stage, the tasks whose original schedules are affected by facility break-downs and person absence should be identified as the to-be-rescheduled tasks.

2.3 Select first element of to-be-rescheduled task list as the current to-be-rescheduled task, recover schedules of the tasks that

- start after the finish time of the current to-be-rescheduled task, and
- preserved in the copy of the original schedules, and
- have never been scheduled.

Reassign the parameters to the current to-be-rescheduled task using agent-based negotiation mechanism (to be introduced in Section 5.3.2).

The current to-be-rescheduled task should be removed from the list of the to-be-rescheduled tasks.

2.4 Check if the reassigned timing parameters are same as those in the copy of the original schedules

If not, the following tasks belonging to the to-be-rescheduled orders should be added to the list of to-be-rescheduled tasks:

- the tasks preceding the current to-be-rescheduled task in the copy of the original schedules preserved in the relevant facility agent and personnel agent that were allocated for the current to-be-rescheduled task
- the tasks preceding the current to-be-rescheduled task in the task sequence of the corresponding order
- the tasks whose original schedules are in conflict with the revised schedule for the current to-be-rescheduled task

2.5 Check if the list of the to-be-rescheduled tasks is empty
If the list is not empty, go to Step 2.3.

2.6 Recover all the tasks that are preserved in the copy of the original schedules and have not been rescheduled so far in the rescheduling process

Step 3. Reschedule the to-be-rescheduled orders that were previously scheduled using the earliest-delivery-time based scheduling

3.1 Select the to-be-rescheduled orders that were previously scheduled using the earliest-delivery-time based scheduling from the whole to-be-rescheduled orders

- When the first rescheduling strategy is employed, the orders that are directly affected by the resource changes are not part of all the to-be-rescheduled orders.
- When the second rescheduling strategy is employed, the orders that are directly affected by the resource changes are part of all the to-be-rescheduled orders at this stage.

Figure 5.20: The algorithm in the second phase of match-up rescheduling approach for resource changes (continued)

3.2 Identify the to-be-rescheduled tasks from the copy of original schedules and sort the list of to-be-rescheduled tasks according to the start time values of these tasks. The to-be-rescheduled task with the smallest start time value is placed at the beginning of the list.

The to-be-rescheduled tasks are the tasks whose original schedules need to be revised. At this stage, the tasks whose original schedules are in conflict with the revised schedules should be identified as the to-be-rescheduled task.

3.3 Select first element of the to-be-rescheduled task list as the current to-be-rescheduled task, recover the tasks that

- completed before the start time of the current to-be-rescheduled task, and
- are preserved in the copy of the original schedules, and
- have never been rescheduled.

Reassign parameters to the current to-be-rescheduled task using agent-based negotiation mechanism (to be introduced in Section 5.3.2)

The current to-be-rescheduled task should be removed from the list of the to-be-rescheduled tasks.

3.4 Check if the reassigned timing parameters are same as those in the copy of original schedules

If not, the following tasks belonging to the to-be-rescheduled orders should be added to the list of to-be-rescheduled tasks:

- the tasks following the current to-be-rescheduled task in the copy of the original schedules preserved in the relevant facility agent and personnel agent that were allocated for the current to-be-rescheduled task
- the tasks following the current to-be-rescheduled task in the task sequence of the corresponding order
- the tasks whose original schedules are in conflict with the revised schedule for the current to-be-rescheduled task

3.5 Check if the list of the to-be-rescheduled tasks is empty
If the list is not empty, go to Step 3.3.

3.6 Recover all the tasks that are preserved in the copy of the original schedules and have not been rescheduled so far in the rescheduling process

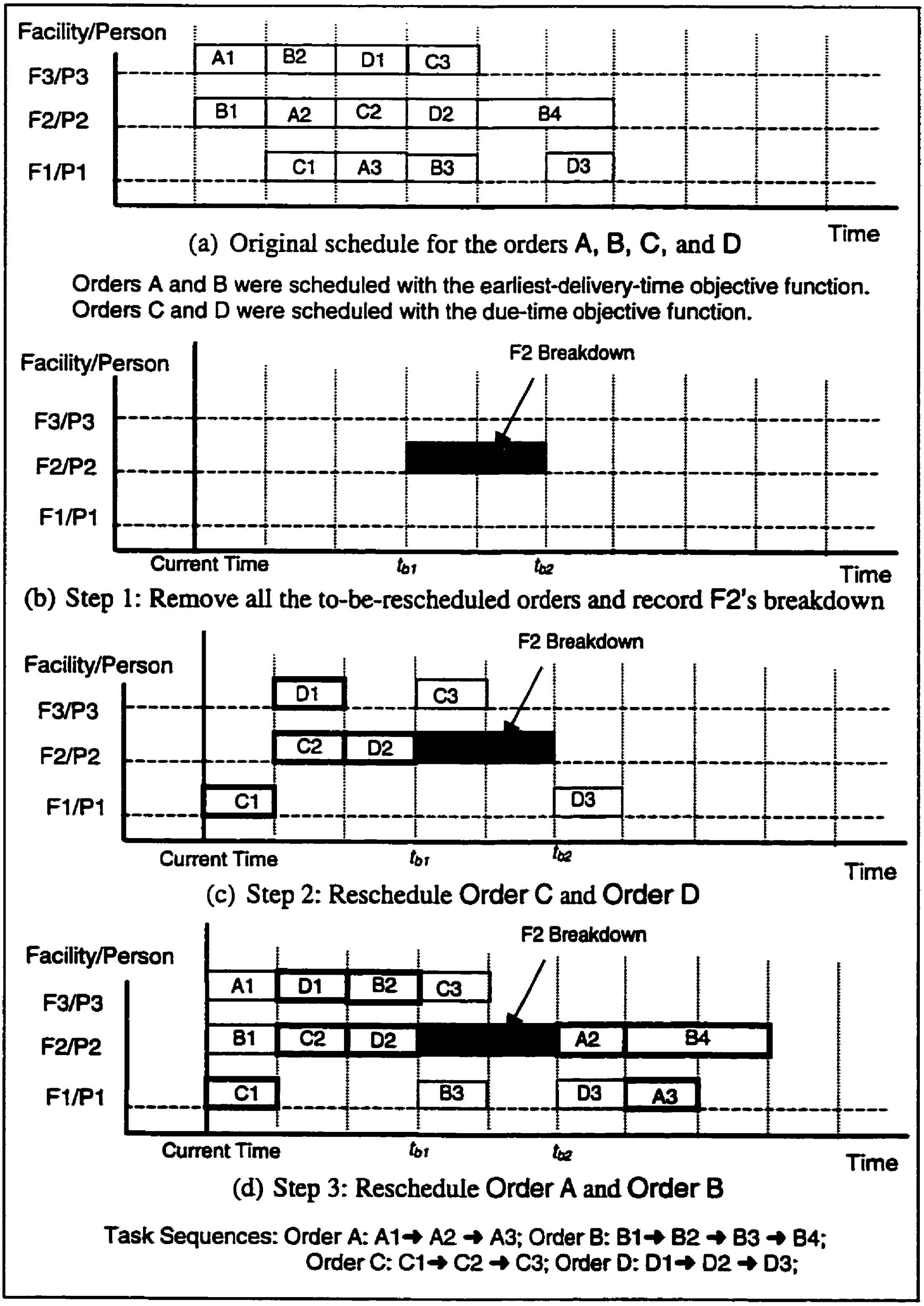
Figure 5.20: The algorithm in the second phase of match-up rescheduling approach for resource changes (continued)

An example is given in Figure 5.21 to illustrate the first rescheduling strategy for responding to resource changes. In this example, it is observed at the current time that the facility *F2* will not be available in the period *(t_{b1}— t_{b2})* and no substitute facilities are found to take over the disrupted tasks *D2* and *B4*.

In Step 1, as shown in Figure 5.21 (b), *Order A*, *Order B*, *Order C*, and *Order D* are identified as the to-be-rescheduled orders and their schedules are removed temporally. The time slot *(t_{b1}— t_{b2})* is recorded as the unavailable period for facility *F2*.

In Step 2, as shown in Figure 5.21 (c), the orders scheduled with the due-time objective function, *Order C* and *Order D*, are rescheduled. *Task D2* is identified as the to-be-rescheduled task at the beginning. Then, *Task D3* that starts after the finish time of *Task D2* is recovered with its original timing parameters in the revised schedule. As the current to-be-rescheduled task, *Task D2* is reassigned with new timing parameters including start time and finish time. Since the reassigned timing parameters of *Task D2* are different from the original timing parameters of this task, *Task D1* that precedes *Task D2* in the task sequence of the original schedule for *Order D* and *Task C2* that precedes the *Task D2* in the original schedules considering facility/person *F2/P2* are added to the list of to-be-rescheduled tasks. Next, *Task C2* is selected as the current to-be-rescheduled task. *Task C3* that starts after the finish time of *Task C2* is recovered with its original timing parameters in the revised schedule. Then *Task C2* is reassigned new timing parameters and *Task C1* is identified as a new to-be-rescheduled task, since *Task C1* precedes *Task C2* in the task sequence. Similarly, the rescheduling process continues until the *Task C1* is rescheduled. The revised schedules are feasible to carry out tasks for *Order C* and *Order D*.

In Step 3, as shown in Figure 5.21 (d), the orders scheduled with the earliest-delivery-time objective function, *Order A* and *Order B*, are rescheduled. Since the original schedules for task *B4* is affected by facility break-down and tasks *A2* and *B2* are in conflict with the revised schedules, these tasks are identified as the to-be-rescheduled tasks at the beginning of this step. *Task A2* is selected as the first current to-be-rescheduled task and the tasks *A1* and *B1* whose finish time values are smaller than the start time of task *A2* are recovered with their original timing parameters in the revised schedule. Task *A2* is then reassigned with new timing parameters and *Task A3* is identified as a new to-be-rescheduled task since *Task A3* follows *Task A2*. Similarly, the rescheduling process continues until *Task B4* is rescheduled.



(a) Original schedule for the orders A, B, C, and D

Orders A and B were scheduled with the earliest-delivery-time objective function.
Orders C and D were scheduled with the due-time objective function.

(b) Step 1: Remove all the to-be-rescheduled orders and record F2's breakdown

(c) Step 2: Reschedule Order C and Order D

(d) Step 3: Reschedule Order A and Order B

Task Sequences: Order A: A1→ A2 → A3; Order B: B1→ B2 → B3 → B4;
Order C: C1→ C2 → C3; Order D: D1→ D2 → D3;

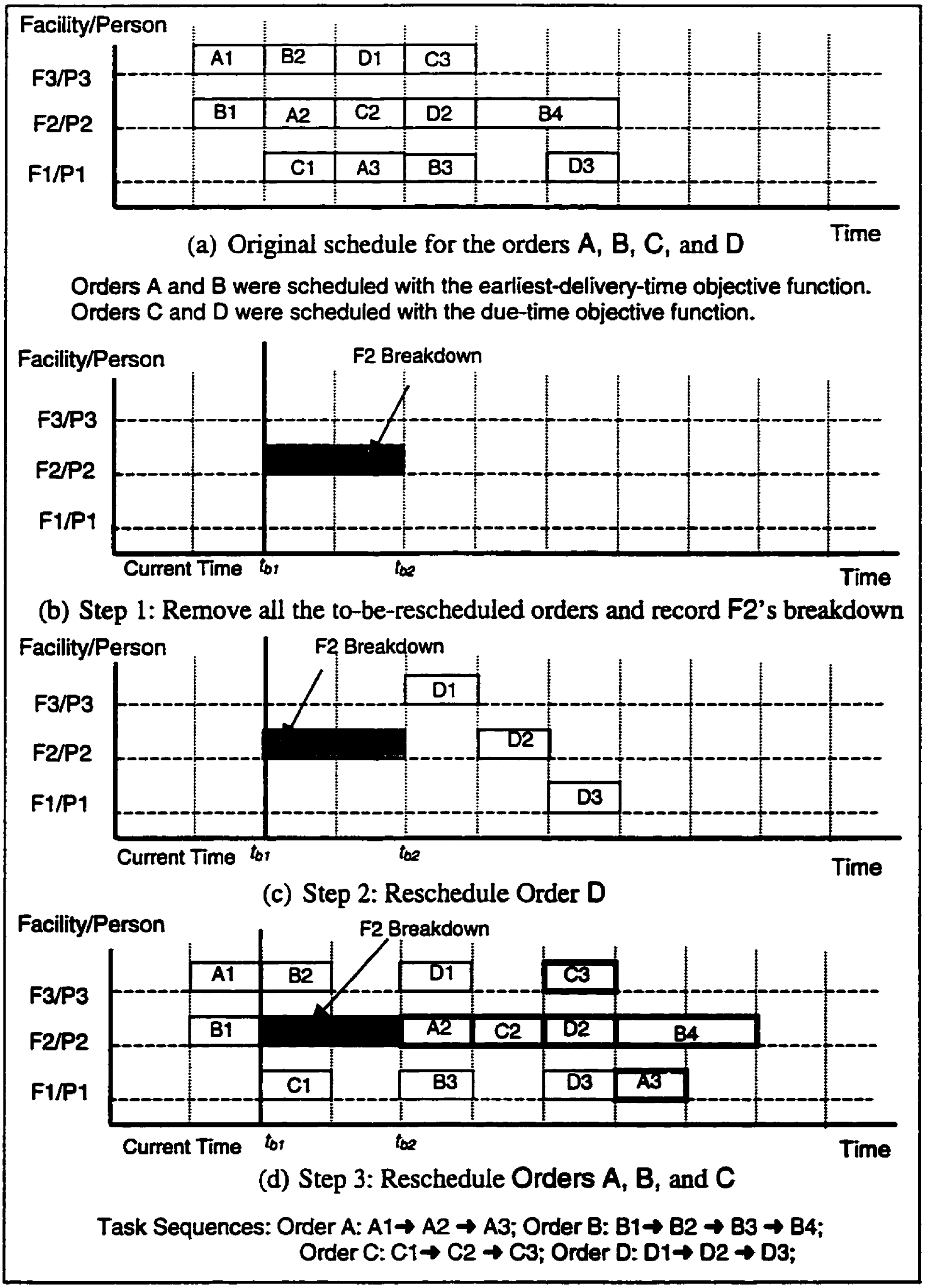
Figure 5.21: An example of match-up rescheduling for resource changes using the first strategy

Another example is given in Figure 5.22 to illustrate the second rescheduling strategy for responding to resource changes. In this example, the facility *F2* breaks down from t_{b1} to t_{b2}. No substitute facilities are found to take over the disrupted tasks *A2* and *C2*. A feasible schedule for *Order C* can not be generated using the first rescheduling strategy, as illustrated in Figure 5.17.

In Step 1, as shown in Figure 5.22 (b), *Order A*, *Order B*, *Order C*, and *Order D* are identified as to-be-rescheduled orders and their schedules are removed temporally. The time slot $(t_{b1}$— $t_{b2})$ is recorded as unavailable period for the facility *F2*.

In Step 2, as shown in Figure 5.22 (c), the order scheduled with the due-time objective function, *Order D*, is rescheduled. *Order C* will be rescheduled in Step 3 as its original schedule is directly affected by the facility *F2's* breakdown. Since no task is in conflict with the facility's breakdown, no task is identified as the to be-rescheduled task at this step. All tasks in *Order D*, *D1*, *D2*, and *D3*, are recovered with their original schedules in the revised schedule.

In Step 3, as shown in Figure 5.22 (d), *Order C* and the orders scheduled with the earliest-delivery-time objective function, *Order A* and *Order B*, are rescheduled. Since the original schedules for tasks *A2* and *C2* have to be changed due to the facility's breakdown, these two tasks are identified as the to-be-rescheduled tasks at the beginning of this step. Then, *Task A2* is selected as the first current to-be-rescheduled task and tasks *A1* and *B1* are recovered with their original schedules in the revised schedules since their finish time values are smaller than the start time of *Task A2*. *A2* is then reassigned with new timing parameters and *Task A3* is identified as a new to-be-rescheduled task since it follows *Task A2* in task sequence. Similarly, the rescheduling process continues until *Task B4* is rescheduled.



(a) Original schedule for the orders A, B, C, and D

Orders A and B were scheduled with the earliest-delivery-time objective function.
Orders C and D were scheduled with the due-time objective function.

(b) Step 1: Remove all the to-be-rescheduled orders and record F2's breakdown

(c) Step 2: Reschedule Order D

(d) Step 3: Reschedule Orders A, B, and C

Task Sequences: Order A: A1→ A2 → A3; Order B: B1→ B2 → B3 → B4;
Order C: C1→ C2 → C3; Order D: D1→ D2 → D3;

Figure 5.22: An example of match-up rescheduling for resource changes using the second strategy

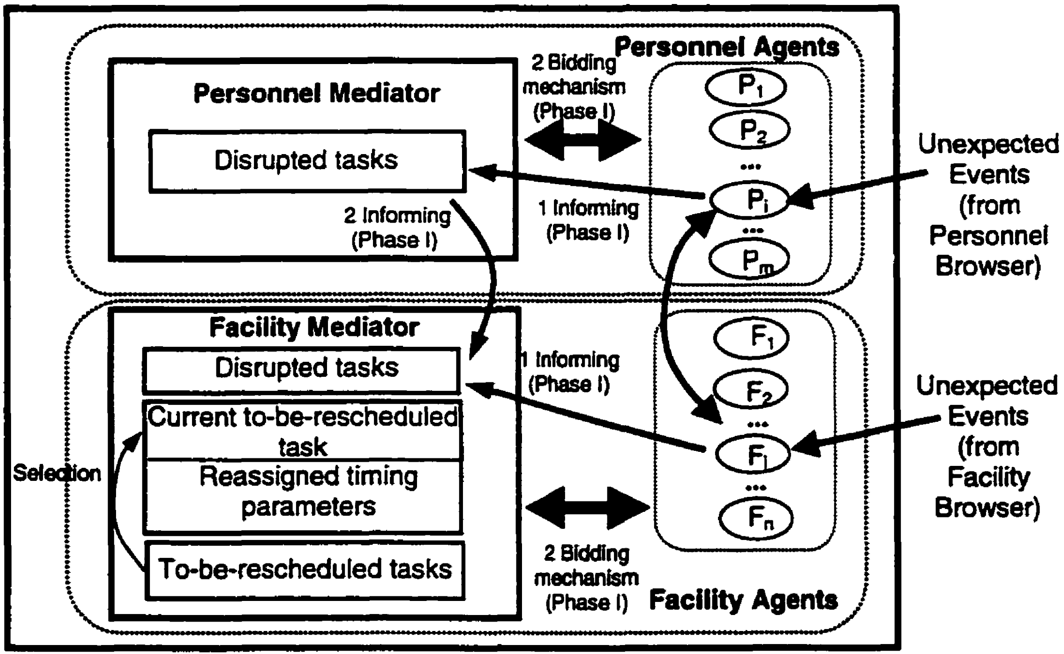
5.3.2 Reallocating Resources and Reassigning Task Timing Parameters Using Agent-based Negotiation

During the rescheduling process, both reallocating resources to the disrupted tasks in the first phase and reassigning timing parameters of these tasks in the second phase are conducted based upon the agent-based negotiation among resource agents and mediators. The agent-based negotiation mechanism is proposed as shown in Figure 5.23.

In the first phase of reactive scheduling, the personnel mediator and the facility mediator reassign the disrupted tasks to substitute persons and facilities using the bidding mechanism introduced in Section 4.4.2. In the second phase of reactive scheduling, the facility mediator conducts the match-up rescheduling for the disrupted tasks that cannot be taken over by substitute persons and facilities without changing the original timing parameters of the disrupted tasks. The agent-based negotiation mechanism in the second phase is the same as the one in the reactive scheduling for responding to production demand changes, which was introduced in Section 5.2.2.

An example is presented in Figure 5.24 to illustrate the agent-based negotiation mechanism used in the first phase of reactive scheduling for responding to resource changes. In this example, the two tasks *A* and *B* are disrupted by absence of the person P_i and the two tasks *C* and *D* are disrupted by break down of the facility F_m. First, the personnel agent P_i and facility agent F_m inform the personnel mediator and the facility mediator about the disrupted tasks. Then the personnel mediator and the facility mediator try to reassign the disrupted tasks to other personnel agents and facility agents using the bidding mechanism. After the bidding process, *Task A* is taken over by the personnel agent P_j and *Task C* is moved to the facility agent F_n. Finally, the personnel mediator informs the facility mediator about the disrupted task *B* that cannot be taken over by other personnel agents without changing the original timing parameters of task *B*. The facility mediator identifies *Task B* and *Task D* as the to-be-rescheduled tasks, which will be reassigned with new timing parameters in the second phase of reactive scheduling for resource changes.

The agent-based negotiation mechanism used in the second phase of reactive scheduling for resource changes can be illustrated using the same example as described in Section 5.2.2.



<In Phase I>

Step 1. **The personnel agents and the facility agents identify the disrupted tasks and inform the relevant mediators**

Step 2. **The mediators select substitute resource agents that take over the disrupted tasks without changing their original timing parameters using the bidding mechanism introduced in Section 4.4.2.**

If a substitute personnel agent for a disrupted task cannot be found, the personnel mediator should inform the facility mediator. If a substitute facility agent for the disrupted task cannot be found, the facility mediator records this task for rescheduling at Phase II.

<In Phase II> (The same as the negotiation mechanism introduced in Figure 5.5)

Step 1. **The facility mediator reassigns the current to-be-rescheduled task to the facility agent that was originally allocated to this task.**

Step 2. **The facility agent negotiates with its related personnel agent to identify new timing parameters to the current to-be-rescheduled task.**

2.1 The facility agent asks the personnel mediator to identify the personnel agent related to the facility agent.

2.2 The personnel mediator finds out the relevant personnel agent

2.3 The facility agent negotiates with the relevant personnel agent to identify new timing parameters to the current to-be-rescheduled task.

Step 3. **The facility agent informs the facility mediator about the new timing parameters of the current to-be-rescheduled task.**

Figure 5.23: The negotiation mechanism for reassigning task timing parameters

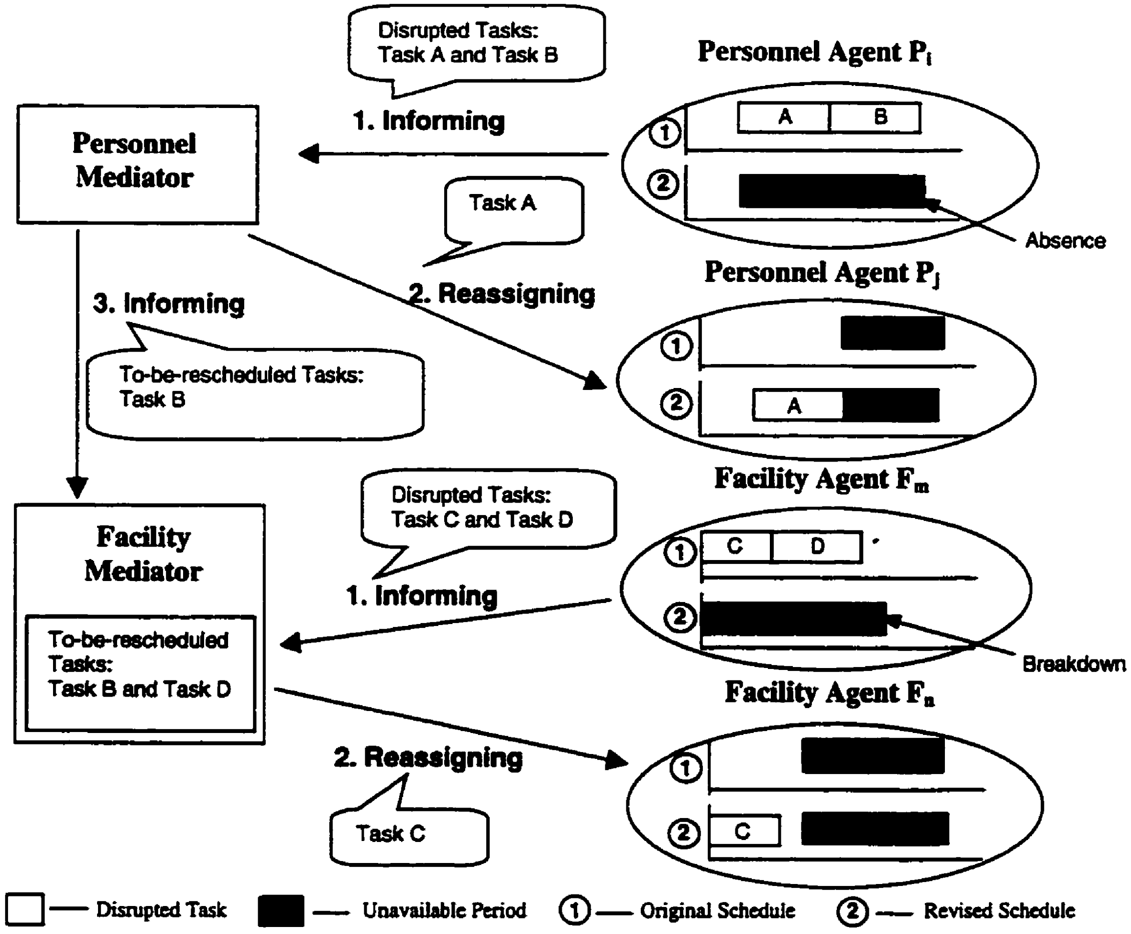


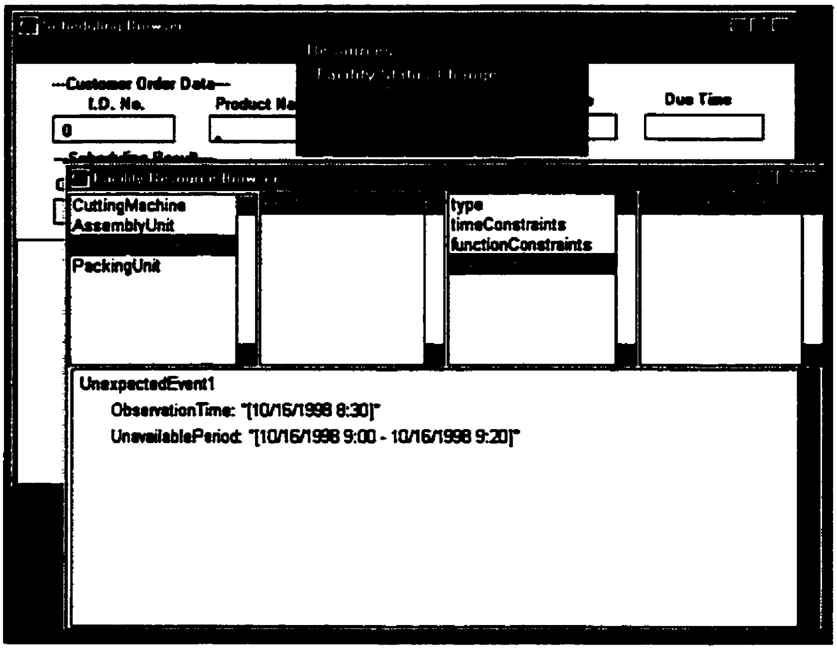
Figure 5.24: An example to illustrate agent-based negotiation mechanism

5.3.3 Case Study I: Facility Breakdowns

In this section, an extended example is used to illustrate the reactive scheduling function of this system for responding to resource changes — facility breakdowns.

The resource management module used in this example is the same as that described in Section 4.3. Four production orders, *Order 1*, *Order 2*, *Order 3*, and *Order 4*, have been scheduled previously. A window product represented by an instance feature *c* of class feature *WindowCenter*, as introduced in Section 4.2.2, needs to be produced in each order. *Order 1* and *Order 2* were scheduled using the due-time based scheduling, while *Order 3* and *Order 4* were scheduled using the earliest-delivery-time based scheduling.

At the current time, *10/16/1998 8:30*, it is observed that the facility *FF01* will not be available from *10/16/1998 9:00* to *10/16/1998 9:20*. ***Scheduling Browser*** and ***Facility Resource Browser*** are used as the user interfaces to conduct the reactive scheduling for facility breakdown, as shown in Figure 5.25. Upon selecting the ***Facility Status Change*** function from the ***Resources*** menu, the system opens a ***Facility Resource Browser***. The user specifies the *Observation Time* and *Unavailable Period* using the ***Facility Resource Browser***. The observation time is the time at which the unexpected events are identified. The system conducts reactive scheduling to revise the original schedules starting from the observation time. The unavailable period specifies when the facility is not available due to maintenance or repair of the facility. An example of facility breakdown event is described as follows:



UnexpectedEvent1
ObservationTime: "[10/16/1998 8:30]"
UnavailablePeriod: "[10/16/1998 9:00 - 10/16/1998 9:20]"

Figure 5.25: The scheduling browser and the facility resource browser as used for reactive scheduling for facility breakdown

UnexpectedEvent1

ObservationTime: "[10/16/1998 8:30]"

UnavailablePeriod: "[10/16/1998 9:00 – 10/16/1998 9:20]"

The time is described by the scheme of *month/day/year hour:minute*. After the unexpected events are described in the facility resource browser, the system can conduct reactive scheduling by selecting *First Rescheduling Strategy* or *Second Rescheduling Strategy* from the *Resources* menu.

Figure 5.26 shows the original schedules for *Order 1*, *Order 2*, *Order 3*, and *Order 4*. Figure 5.27 shows the revised schedule generated using the *First Rescheduling*

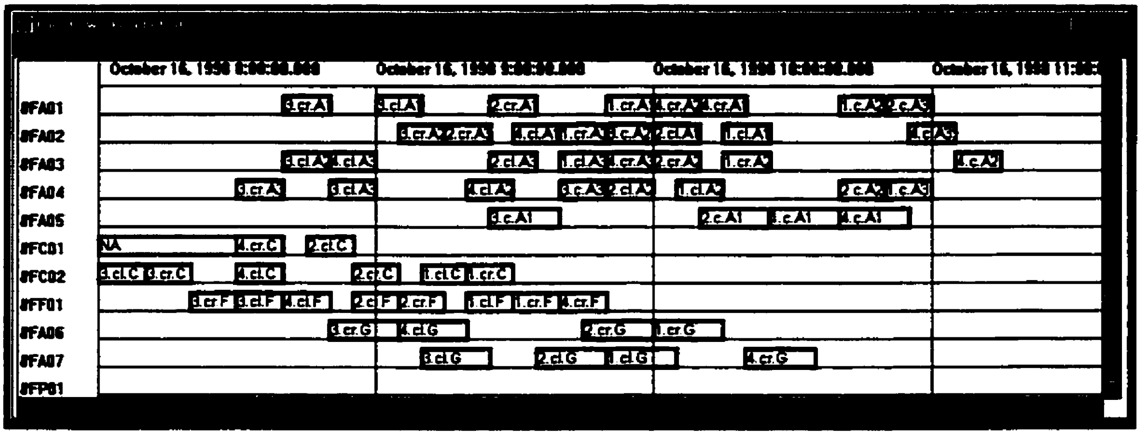
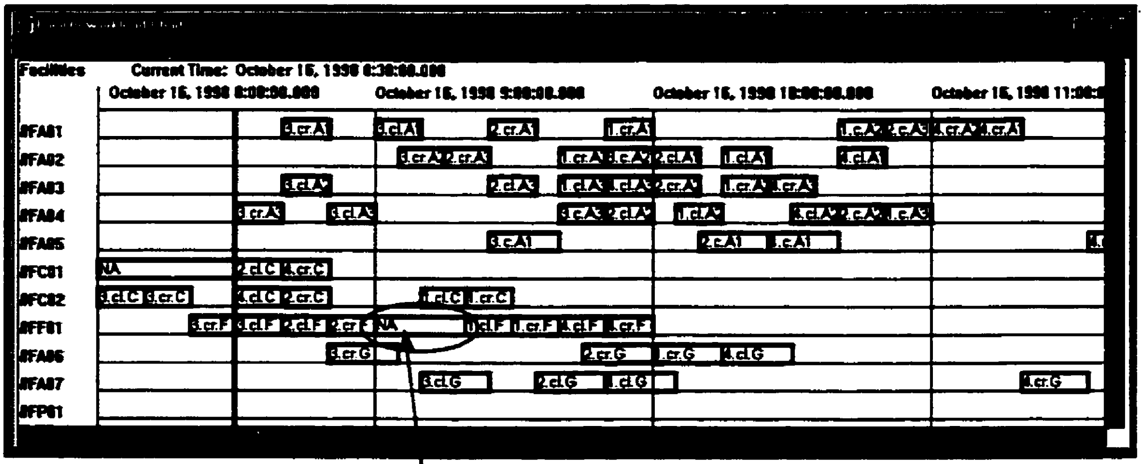

October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00
#FA01
#FA02
#FA03
#FA04
#FA05
#FC01
#FC02
#FF01
#FA06
#FA07
#FP01
NA

Figure 5.26: The original schedules for *Orders 1, 2, 3,* and *4*


Facilities
Current Time: October 16, 1998 8:30:00.000
October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00
#FA01
#FA02
#FA03
#FA04
#FA05
#FC01
#FC02
#FF01
#FA06
#FA07
#FP01
NA
NA

FF01 breakdown

Figure 5.27: The revised schedules for the facility *FF01* breakdown

Strategy for responding to breakdown of the facility *FF01*. Comparing the original schedules and the revised schedules, it is found that only partial schedules are changed in this case. The task schedules for *Order 2* are partially shifted forward to new time slots, and task schedules for *Order 1* still keep their original time slots. The task schedules for *Order 3* and *Order 4* are partially shifted backward to new time slots. The comparison between the original schedules and the revised schedules is shown in Table 5.3.

Table 5.3: The comparison between the original schedules and the revised schedules

Order	Arrival Time	Due Time	Original Schedule		Revised Schedule	
			Release	Completion	Release	Completion
1	10/15 8:00	10/16 11:00	10/16 9:10	10/16 11:00	10/16 8:00	10/16 11:00
2	10/15 8:05	10/16 11:00	10/16 8:45	10/16 11:00	10/16 8:20	10/16 11:00
3	10/15 8:10		10/16 8:00	10/16 10:00	10/16 9:10	10/16 10:00
4	10/15 8:15		10/16 8:30	10/16 11:15	10/16 8:30	10/16 11:40

5.3.4 Case Study II: Personnel Absence

In this section, an extended example is used to illustrate the reactive scheduling function of this system for responding to resource change — personnel absence.

The resource management module used in this example is the same as that described in Section 4.3. Four production orders, *Order 1*, *Order 2*, *Order 3*, and *Order 4*, have been scheduled previously. A window product represented by an instance feature *c* of class feature *WindowCenter*, as introduced in Section 4.2.2, needs to be produced in each order. *Order 1* and *Order 2* were scheduled using the due-time based scheduling, while *Order 3* and *Order 4* were scheduled using the earliest-delivery-time based scheduling.

At the current time, *10/16/1998 8:30*, it is observed that the person *PM03* will not be available from *10/16/1998 8:30* to *10/16/1998 9:00*. *Scheduling Browser* and *Personnel Resource Browser* are used as user interfaces to conduct the reactive scheduling for personnel absence, as shown in Figure 5.28. Upon selecting the *Person*

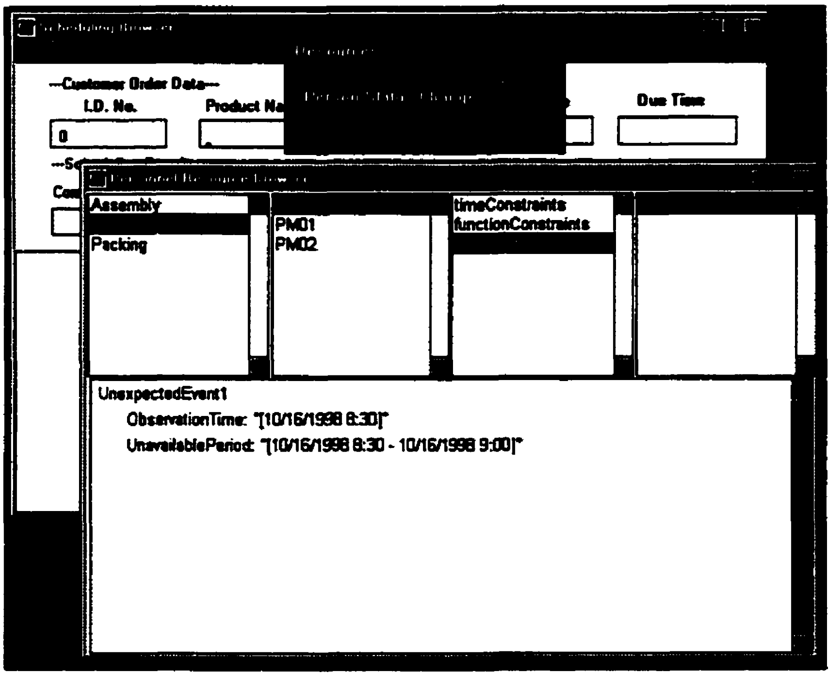
Status Change function from the *Resources* menu, the system opens the *Personnel Resource Browser*. The user specifies the *Observation Time* and *Unavailable Period* using the *Personnel Resource Browser*. The observation time is the time at which the unexpected events are found. The system conducts reactive scheduling to revise the original schedules starting from the observation time. The unavailable period specifies when the person is not available. An example of personnel absence description is shown as follows:

UnexpectedEvent1

 ObesrvationTime: "[10/16/1998 8:30]"

 UnavailablePeriod: "[10/16/1998 8:30 – 10/16/1998 9:00]"

The time is described by the scheme of *month/day/year hour:minute*. After the unexpected events are described in the personnel resource browser, the system can conduct reactive scheduling by selecting *First Rescheduling Strategy* or *Second Rescheduling Strategy* from *Resources* menu.



UnexpectedEvent1
 ObservationTime: "[10/16/1998 8:30]"
 UnavailablePeriod: "[10/16/1998 8:30 - 10/16/1998 9:00]"

Figure 5.28: The scheduling browser and the personnel resource browser as used for reactive scheduling for personnel absence

Figure 5.29 and Figure 5.30 show the original schedules and personnel workload chart respectively for *Order 1*, *Order 2*, *Order 3*, and *Order 4*. Figure 5.31 and Figure 5.32 show the revised schedules and personnel workload chart respectively for the absence of the person *PM03*. The *Second Rescheduling Strategy* is used in the case. Comparing the original schedules and the revised schedules, it is found that only partial schedules are changed in this case. The schedules for *Order 1* still keep their original time slots after rescheduling. The task schedules for *Order 3* and *Order 4* are partially shifted backward to new time slots. *Order 2* is treated the same as *Order 3* and *Order 4* in the rescheduling process and its schedule is shifted backward to new time slots. The comparison between the original schedules and the revised schedules is shown in Table 5.4.

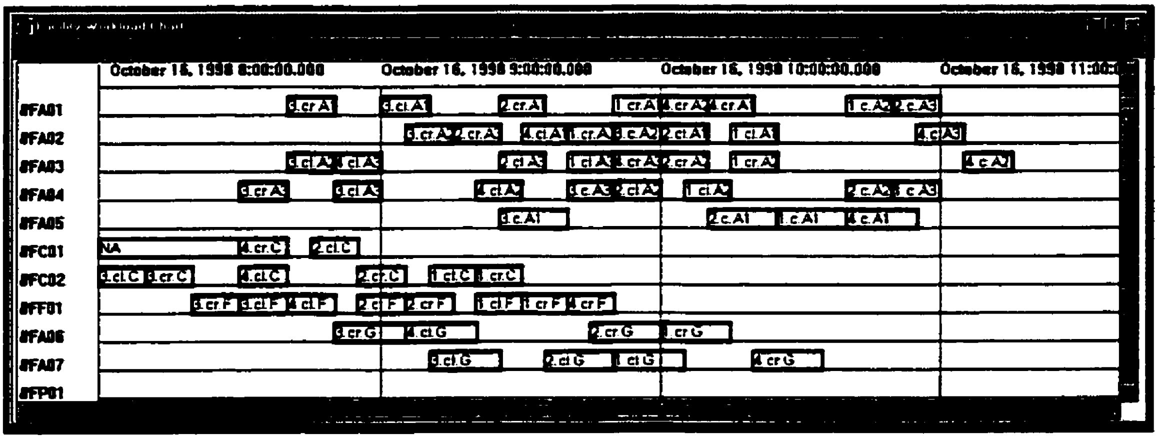

October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00:0
#FA01
#FA02
#FA03
#FA04
#FA05
#FC01
#FC02
#FF01
#FA06
#FA07
#FP01
NA

Figure 5.29: The original schedules for *Orders 1, 2, 3,* and *4*

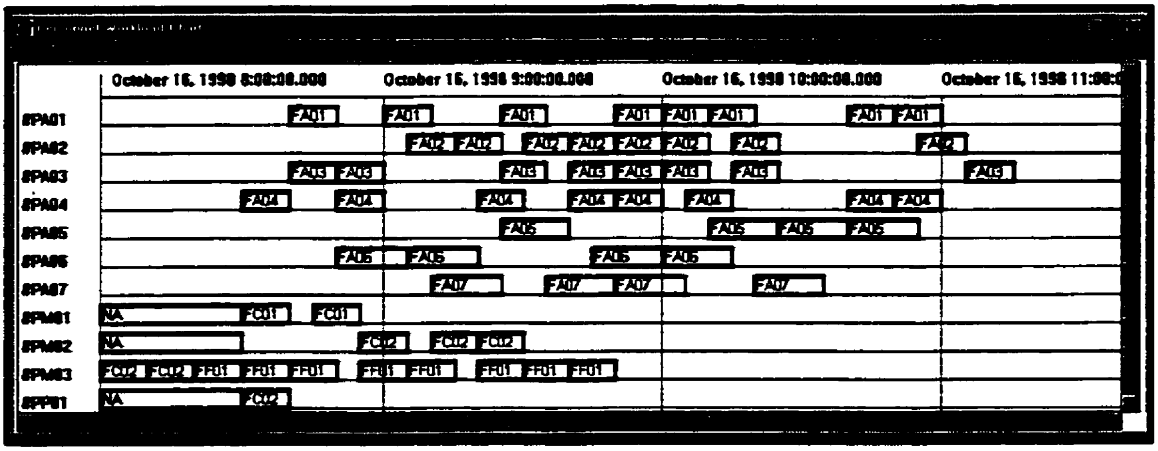
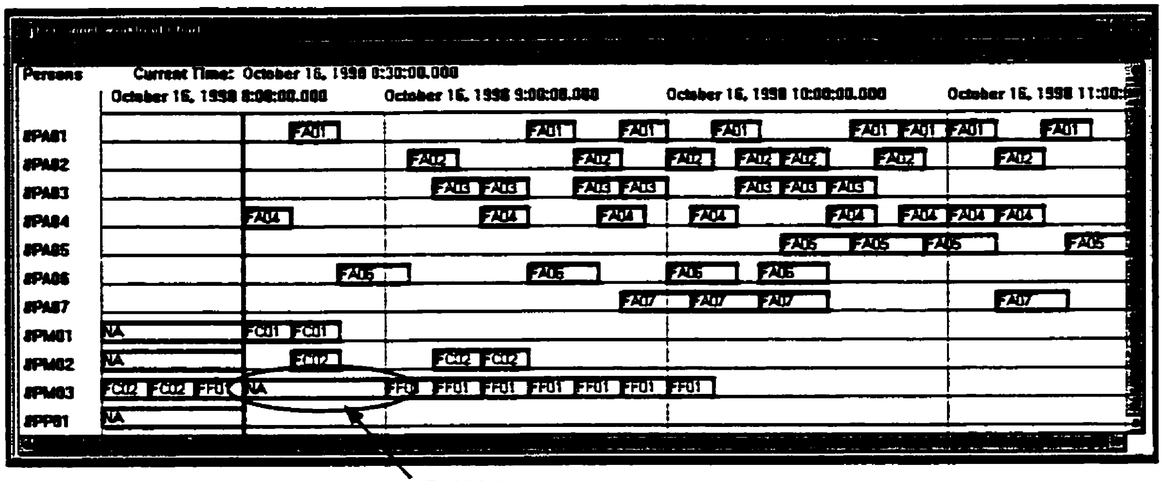
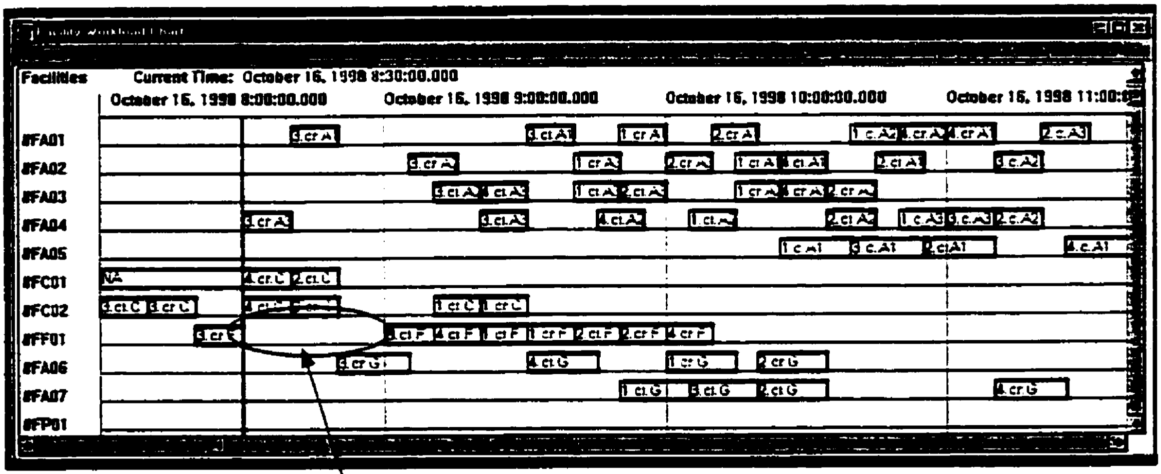


Figure 5.30: The personnel workload chart for the original schedules


Persons
Current Time: October 16, 1998 8:30:00.000
October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00:0
#PA01
#PA02
#PA03
#PA04
#PA05
#PA06
#PA07
#PM01
#PM02
#PM03
#PP01
NA
FC01
FC02
FF01

PM03 Absence

Figure 5.31: The personnel workload chart for the revised schedules


Facilities
Current Time: October 16, 1998 8:30:00.000
October 16, 1998 8:00:00.000
October 16, 1998 9:00:00.000
October 16, 1998 10:00:00.000
October 16, 1998 11:00:0
#FA01
#FA02
#FA03
#FA04
#FA05
#FC01
#FC02
#FF01
#FA06
#FA07
#FP01
NA

PM03 Absence

Figure 5.32: The revised schedules for the person *PM03* absence

Table 5.4: The comparison between the original schedules and the revised schedules

Order	Arrival Time	Due Time	Original Schedule		Revised Schedule	
			Release	Completion	Release	Completion
1	10/15 8:00	10/16 11:00	10/16 9:10	10/16 11:00	10/16 9:10	10/16 11:00
2	10/15 8:05	10/16 11:00	10/16 8:45	10/16 11:00	10/16 8:45	10/16 11:30
3	10/15 8:10		10/16 8:00	10/16 10:00	10/16 9:10	10/16 11:20
4	10/15 8:15		10/16 8:30	10/16 11:15	10/16 8:15	10/16 12:00

5.4 Summary

This chapter presented the reactive scheduling function of the intelligent production scheduling system. Two intelligent reactive scheduling mechanisms have been proposed in this research for schedule revision to respond to production demand changes and unexpected resource changes. The production demand changes include canceling old orders and inserting urgent orders. The resource changes include facility breakdowns and personnel absence.

The intelligent reactive scheduling is conducted based on the match-up rescheduling approach and the agent-based negotiation. With application of the match-up rescheduling approach in the multi-agent environment, the original schedule is only partially changed for responding to production demand changes and resource changes. During the reactive scheduling process, agent-based negotiation is used to reassign timing parameters and reallocate resources to the tasks that need to be rescheduled.

Chapter 6

System Implementation and Evaluation

The implementation and evaluation of the intelligent production scheduling system are presented in this chapter. Section 6.1 describes the system architecture. Section 6.2 introduces the system implementation using VisualWorks 2.5. Section 6.3 presents simulation results to evaluate the performance of the intelligent production scheduling system.

6.1 System Architecture and User Interfaces

The architecture of the intelligent production scheduling system is illustrated in Figure 6.1. The system has been implemented using VisualWorks 2.5 — a window-based environment for programming in Smalltalk language [Hopkins 95]. The system consists of three modules: *product modeling module, resource management module*, and *scheduling module*. The various user interfaces of this system related to production scheduling and their interactions are shown in Figure 6.2.

The intelligent production scheduling system and the feature-based product design system are integrated together as a software package, called *Integrated Design and Manufacturing System*. The user interfaces of the intelligent production scheduling system, including the *Class Feature Browser, Instance Feature Browser, Facility Resource Browser, Personnel Resource Browser, Scheduling Browser*, can be opened

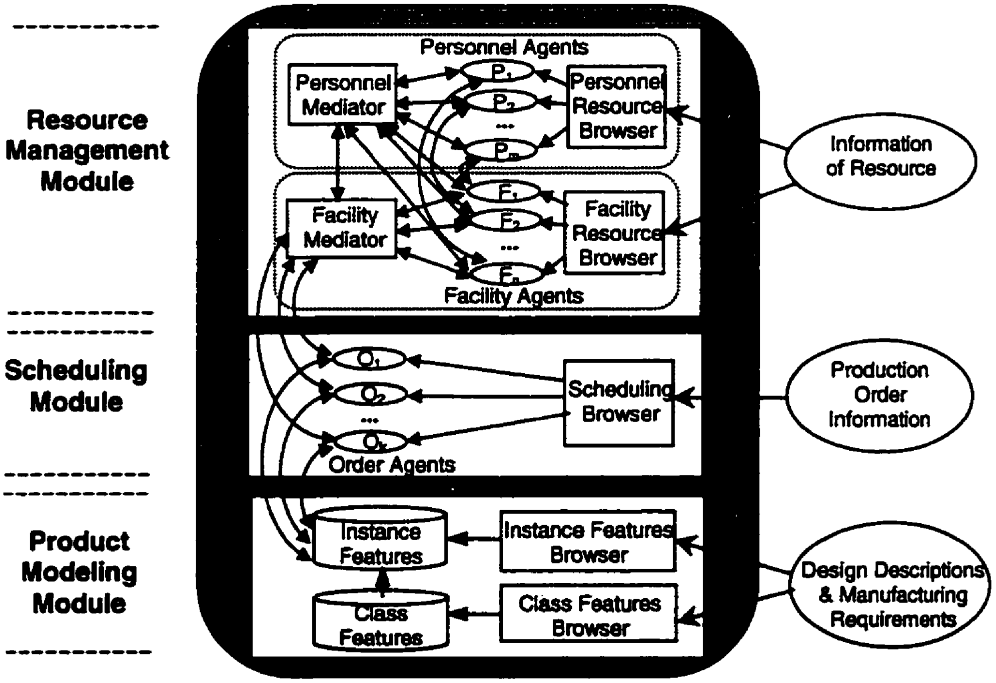


Figure 6.1: The architecture of intelligent production scheduling system

using an user interface, called *Launcher* as shown in Figure 6.3. The user interfaces *Facility Workload Chart* and *Personnel Workload Chart* are opened from the *Scheduling Browser*. These interfaces are used for defining class features and their manufacturing requirements, generating instance features and modifying their manufacturing requirements, modeling and managing resource databases (including facilities and persons), generating production orders associated with designed products, conducting predictive and reactive scheduling, and displaying scheduling results.

In the product modeling module, a product is modeled by its composing primitives — features. Features are defined at two different levels: class level corresponding to standard product data and instance level corresponding to special product data. The manufacturing requirements to produce each feature are also defined as part of feature descriptions at class level and instance level. *Class Feature Browser* and *Instance Feature Browser* are used for defining class features and generating instances.

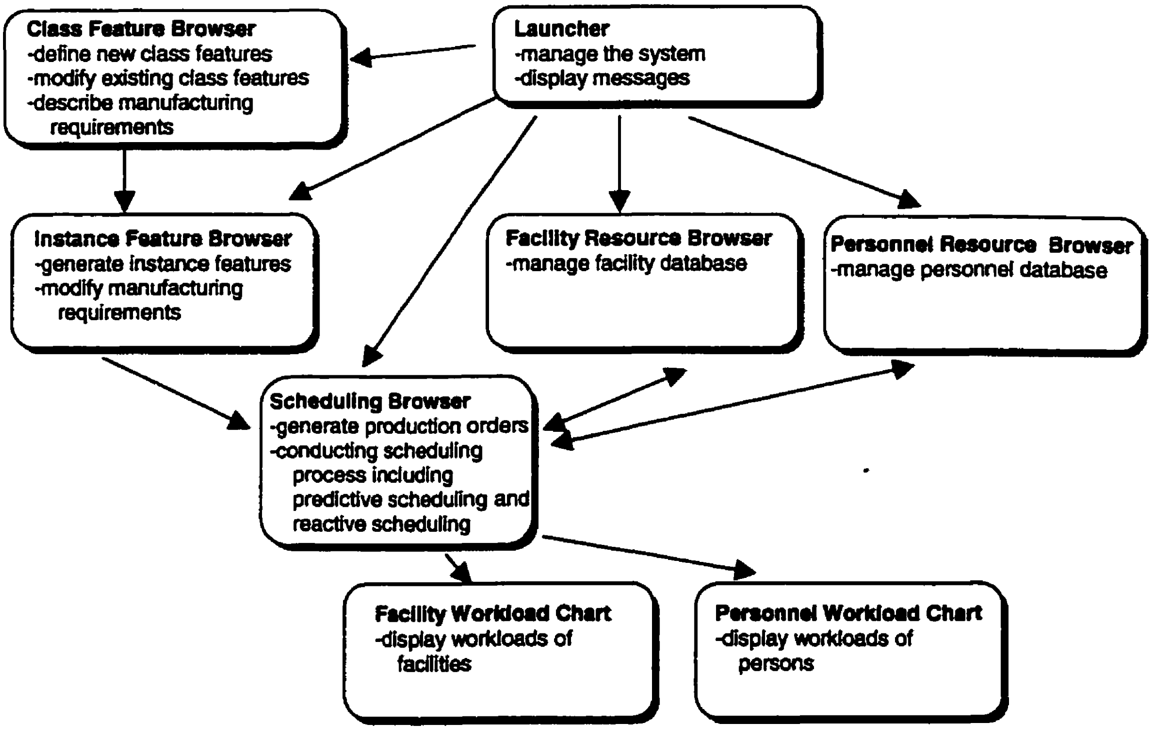

Class Feature Browser
-define new class features
-modify existing class features
-describe manufacturing requirements
Launcher
-manage the system
-display messages
Instance Feature Browser
-generate instance features
-modify manufacturing requirements
Facility Resource Browser
-manage facility database
Personnel Resource Browser
-manage personnel database
Scheduling Browser
-generate production orders
-conducting scheduling process including predictive scheduling and reactive scheduling
Facility Workload Chart
-display workloads of facilities
Personnel Workload Chart
-display workloads of persons

Figure 6.2: User interface environment of the system

In the resource management module, the resources are organized in a multi-agent system. In this system, two types of resources, facilities and persons, are considered. Thus, two kinds of resource browsers, *Facility Resource Browser* and *Personnel Resource Browser*, are developed for modeling and managing the relevant resource agents.

The scheduling module aims at conducting the predictive scheduling function and the reactive scheduling function. Upon receiving the customer requirements, the system

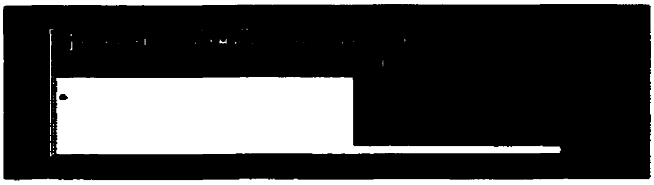


Figure 6.3: A snapshot of system launcher

generates a production order to manufacturing a product. The order is modeled as an order agent to represent the customer requirements. The system identifies the optimal production schedule considering the design constraints and manufacturing constraints. During production process, the scheduling module conducts reactive scheduling for responding to production demand changes and resource changes. ***Scheduling Browse*** is used for generating production orders and associating them with designed products, and conducting predictive scheduling and reactive scheduling. The scheduling results are displayed on ***Facility Workload Chart*** and ***Personnel Workload Chart***.

6.2 System Implementation Using VisualWorks 2.5

VisualWorks 2.5 provides a class library of around one thousand classes [Sharp 97]. The system classes that are commonly used in this system include ***Object, ApplicationModel, Dictionary, OrderedCollection, SortedCollection, String, View, Timestamp,*** etc. Explanations to these classes are given in Table 6.1.

Table 6.1: The system classes used for system implementation

Class Names	**Explanations**
Object	A root super-class of Smalltalk from which all other classes are descended
ApplicationModel	A class to provide frameworks to help manage user interface applications with MVC (Model-View-Controller) components
Dictionary	A class for storing data in form of a key-and-value pair where key is described by a symbol and value that can be any Smalltalk object
OrderedCollection	A class that stores Smalltalk objects in the order they were added, and will grow and shrink as adding and removing objects

Table 6.1: The system classes used for system implementation (continued)

Class Names	Explanations
SortedCollection	A class that sorts the stored Smalltalk objects according to some collating sequences A sub-class of *OrderedCollection*
String	A class for representing an array of characters
View	A class that is used to present to user the information contained in a model
Timestamp	A class for representing and operating time in form of year/month/day/hour/minute/second

In addition to the system classes, a number of new classes have been created for implementation of the intelligent production scheduling system. Those new classes were defined as sub-classes of the system classes. All the variables and methods of super-classes are inherited by the sub-classes automatically. The newly created classes are organized in different categories, as presented in Table 6.2. Among the classes listed in this table, the classes in the categories *Feature-Design* and *Feature-Design-Instance* were developed for the intelligent product design system [Yadav 98].

Table 6.2: New created classes and categories used for system implementation

Categories	Classes	Super-Classes
Feature-Design	• ClassFeatureBrowser • FeatureClass • FeatureInitialize • MainBrowser	ApplicationModel
Feature-Design-Instance	• InstanceBrowser • FeatureInstance	ApplicationModel

Table 6.2: New created classes and categories used for system implementation (continued)

Categories	Classes	Super-Classes
Resource	• FacilityResourceBrowser • FacilityAgent • FacilityMediator • PersonnelResourceBrowser • PersonnelAgent • PersonnelMediator	ApplicationModel
Scheduling	• SchedulingBrowser • OrderAgent • RootNode • SearchNode	ApplicationModel
	• FacilityWorkloadChart • PersonnelWorkloadChart	View

A number of global variables have been defined in the system to preserve the knowledge and data. For instance, the global variables *FacilityCategoryDic* and *PersonnelCategoryDic* are used to store all the facility agents and personnel agents respectively. When a new facility agent or personnel agent is defined, its description is stored in the global variable *FacilityCategoryDic* or *PersonnelCategoryDic*. The major global variables used in this system are listed in Table 6.3.

A Smalltalk class called *SystemInitialize* has been defined in the system for initializing all the global variables. When a new copy of Smalltalk system is used, the user should execute the instance method *initialize* of the class *SystemInitialize*.

Table 6.3: Major global variables used in this system

Variables	Stored Data	Type
FeatureCategoryDic	Class features defined in the system	Dictionary
FeatureInstanceDic	Instance features generated in the system	Dictionary
FeatureAspectsList	List of aspects shown in the class feature browser	OrderedCollection
FeatureInstanceAspectList	List of aspects shown in the instance feature browser	OrderedCollection
FacilityCategoryDic	Facility agents defined in the system	Dictionary
FacilityAspectList	List of aspects shown in the facility resource browser	OrderedCollection
PersonnelCategoryDic	Personnel agents defined in the system	Dictionary
PersonnelAspectList	List of aspects shown in the personnel resource browser	OrderedCollection
OrderDic	Orders defined in the system	Dictionary

6.2.1 New Classes for Implementing the Resource Management Module

All facility agents and personnel agents are translated into Smalltalk classes. Two Smalltalk classes, called *FacilityAgent* and *PersonnelAgent*, are used to represent the facility agent and personnel agent respectively.

FacilityResourceBrowser and *PersonnelResourceBrowser* are the two Smalltalk classes for implementing the facility resource browser and the personnel resource browser respectively. The methods in these classes are used to define the interface and implement the actions of menu items. When a facility agent or a personnel agent is being defined, the actions to be performed include saving and removing the descriptions of facility agents or personnel agents, including types, function-constraints, and time-constraints. These descriptions are stored in Smalltalk variables. For instance, the data structure to store the facility agents is shown in Figure 6.4.

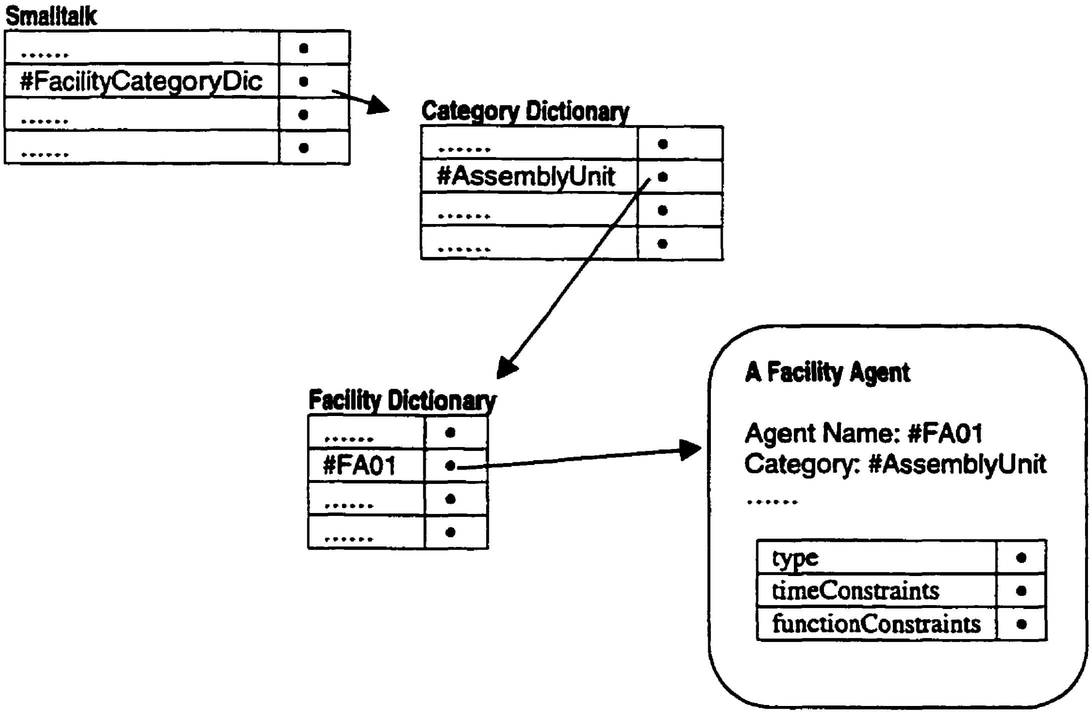


Figure 6.4: The data structure in which a facility agent is stored

6.2.2 New Classes for Implementing the Production Scheduling Mechanism

In this system, the following Smalltalk classes are related with the implementations of predictive scheduling and reactive scheduling:

- *SchedulingBrowser*
- *FacilityWorkloadChart*
- *PersonnelWorkloadChart*
- *OrderAgent*

- *FacilityAgent*
- *PersonnelAgent*
- *FacilityMediator*
- *PersonnelMediator*
- *SearchNode*
- *RootNode.*

The Smalltalk class *SchedulingBrowser* is used for implementing the interface for generating the production order and conducting the scheduling process. The major methods defined in this class mainly include:

- *GenerateNewOrder*
- *DoEarliestDeliveryTimeBasedScheduling*
- *DoDueTimeBasedScheduling*
- *CancelAnOldOrder*
- *InsertAnUrgentOrder*
- *ChangeFacilityStatus*
- *ChangePersonnelStatus*
- *DoFirstReschedulingStrategy*
- *DoSecondReschedulingStrategy*

The scheduling result is presented using the two Smalltalk classes *FacilityWorkloadChart* and *PersonnelWorkloadChart*, which are sub-classes of Smalltalk class *View.*

The descriptions of a production order are preserved in the Smalltalk class *OrderAgent.* Each of order agents has a number of instance variables to describe the different types of descriptions. The instance variables of the class *OrderAgent* include:

- *arrivalTime*
- *dueTime*
- *iDNumber*
- *amount*

- *schedulingObjective*
- *releaseTime*
- *completionTime*
- *schedule*

The two Smalltalk classes *SearchNode* and *RootNode* are used for heuristic search in the predictive scheduling process. These classes represent search states using a number of variables as follows:

- *currentSearchTask*
- *evaluationValue*
- *toBeSchedulingFeatures*
- *toBeSchedulingTasks*
- *schedulingResult*
- *currentVersionResourceManagementModule*

The reactive scheduling is conducted based on the match-up rescheduling approach and agent-based negotiation. To execute the match-up rescheduling for responding to production demand changes and resource changes, the Smalltalk class *FacilityMediator* has the following methods:

- *cancelAnOldOrder*
- *insertAnUrgentOrder*
- *firstReschedulingStrategyForResourceChanges*
- *secondReschedulingStrategyForResourceChanges*

To realize agent-based negotiation in scheduling process, the Smalltalk classes *FacilityAgent, PersonnelAgent, FacilityAgent,* and *PersonnelAgent* have their methods such as:

In *FacilityAgent*

- *getRequest:*
- *submitBid:with:*
- *proposeFinishTime:with:with:*

- *proposeReleaseTime:with:with:*

In *PersonnelAgent*

- *getRequest:*
- *submitBid:with:*
- *proposeFinishTime:with:with:*
- *proposeReleaseTime:with:with:*

In *FacilityMediator*

- *getRequest:*
- *sendAnnouncement*

In *PersonnelMediator*

- *getRequestFromFacilityAgent:with:with:*

6.3 System Evaluation

We evaluate performance of the intelligent production scheduling system based on a simulation model for window production at Gienow Building Products Ltd. Gienow Building Products Ltd. is a manufacturing company for producing windows and doors. The production management, including design and manufacturing activities, at this company is based upon the orders from the customers.

In the simulation model, the manufacturing resources in job-shop, including facilities and persons, are described the same as those in Section 4.3.3. Eleven facilities are 2 cutting machines, 1 framing machine, 7 assembly units, and 1 packing unit, while eleven persons are assigned to work for the relevant facilities.

The window products to be ordered in the simulation are instance features of class features *Window* and *WindowCenter*, as described in Section 4.2.1. Manufacturing requirements for these products are defined in Section 4.2.2.

The evaluation of the system performance will be presented in Section 6.3.1 and Section 6.3.2.

Worth to be mentioned, a qualitative comparison of the scheduling approaches proposed in this research with conventional centralized scheduling approaches would have been desirable, although the theoretical analysis on advantages of distributed scheduling approach has been discussed in Chapter 2. Unfortunately, due to the limitation of experimental facilities, we have not conducted the testing experiment for this comparison. In the future work, the testing experiment will be desirable and should focus on the following issues, as listed in Table 6.4.

Table 6.4: Comparison of the scheduling approaches proposed in this research with conventional centralized scheduling approaches [Shaw 87]

	Conventional Approaches (Centralized Approach)	**The Approaches Proposed in This Research (Distributed Approach)**
Execution of Scheduling (Efficiency of Scheduling)	only one master scheduler	a number of collaborative entities
Control Mechanism for Scheduling	master-slave control with unidirectional message-passing	coordination through exchanging messages
Vulnerability to Scheduler's Failure	entire system would stop	only the particular entity would be disrupted
Manufacturing Database	a global database	A distributed database
Maintaining Dynamic System Information	constant updating through communication messages	local updating without communication activities

6.3.1 Performance of Predictive Scheduling

The performance of predictive scheduling for this system is evaluated in the following two aspects: scheduling quality and efficiency. In the predictive scheduling process, the optimal schedule is identified using heuristic search — the beam search method is employed to improve the search efficiency. Thus, the scheduling quality is measured by the objective function selected in the scheduling process: the earliest-delivery-time objective function or the due-time objective function. The scheduling efficiency is measured by the number of the search nodes that are generated in heuristic search.

In this simulation, two orders, *Order 1* and *Order 2*, are assumed to have been scheduled with the earliest-delivery-time objective function and the due-time objective function respectively.

Order 1: Release Time: 10/16/98 8:30 AM, Completion Time: 10/16/98 11:25 AM.

Order 2: Release Time: 10/16/98 10:10 AM, Completion Time: 10/16/98 1:00 PM.

Currently, two new orders, *Order 3* and *Order 4*, need to be scheduled. The system performance is evaluated using the following scheduling results generated for the two new orders:

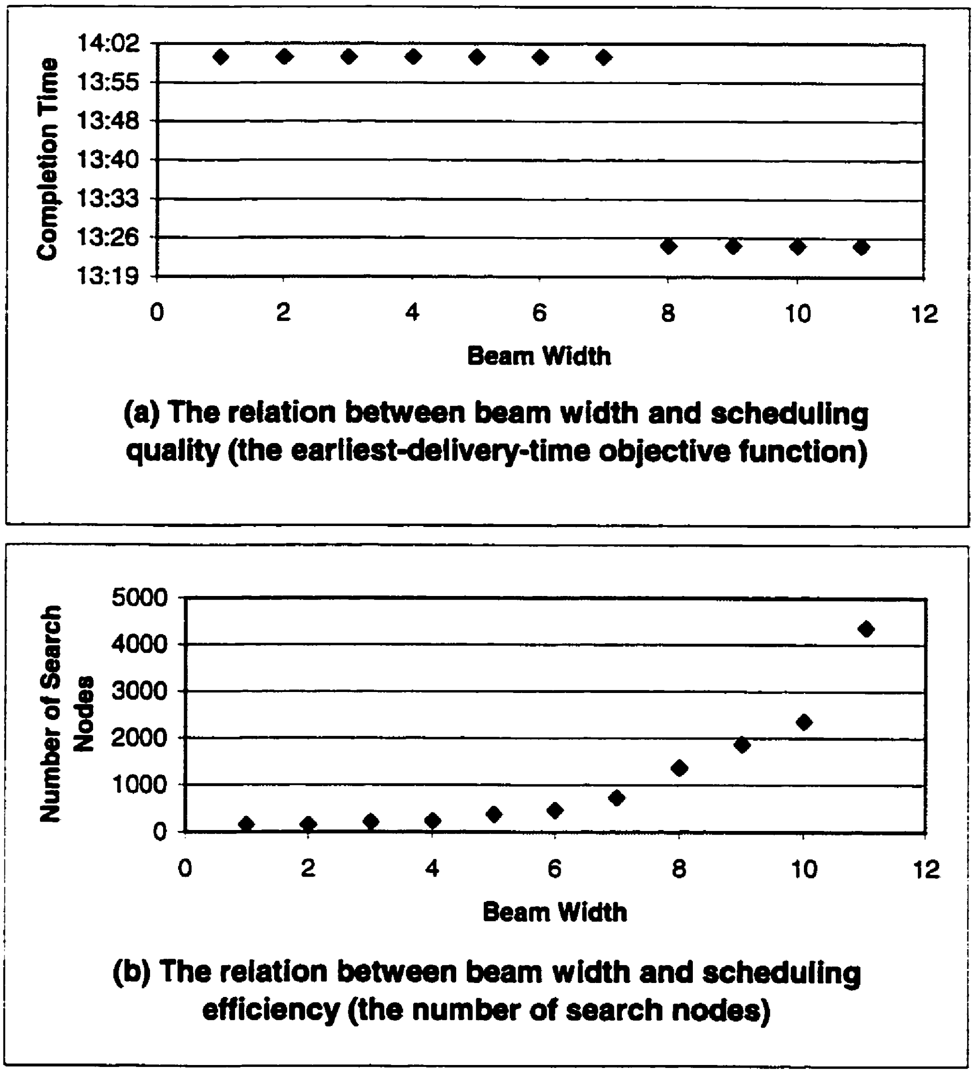
Order 3: Arrival Time: 10/15/98 8:10 AM, Objective function: the earliest-delivery-time objective function.

Order 4: Due Time: 10/16/98 2:30 PM, Objective function: the due-time objective function.

The relation between the beam width used in search process and the scheduling quality/efficiency are shown in Figure 6.5 and Figure 6.6. These figures show that (1) when the beam width is increased, the objective measure is improved as follows: the completion time for *Order 3* is shifted to earlier time and the release time for *Order 4* is moved to later time close to due time; (2) when the beam width is increased, the number of search nodes is also increased, meaning the scheduling efficiency is decreased.

6.3.2 Performance of Reactive Scheduling

In the intelligent production scheduling system, reactive scheduling is conducted to revise only part of the original schedule for responding to the production demand changes and resource changes without rescheduling all the required tasks. The reactive scheduling quality is also measured by the objective function, the earliest-delivery-time objective function and the due-time objective function. The scheduling efficiency is measured by the number of the revised tasks in the reactive scheduling process.

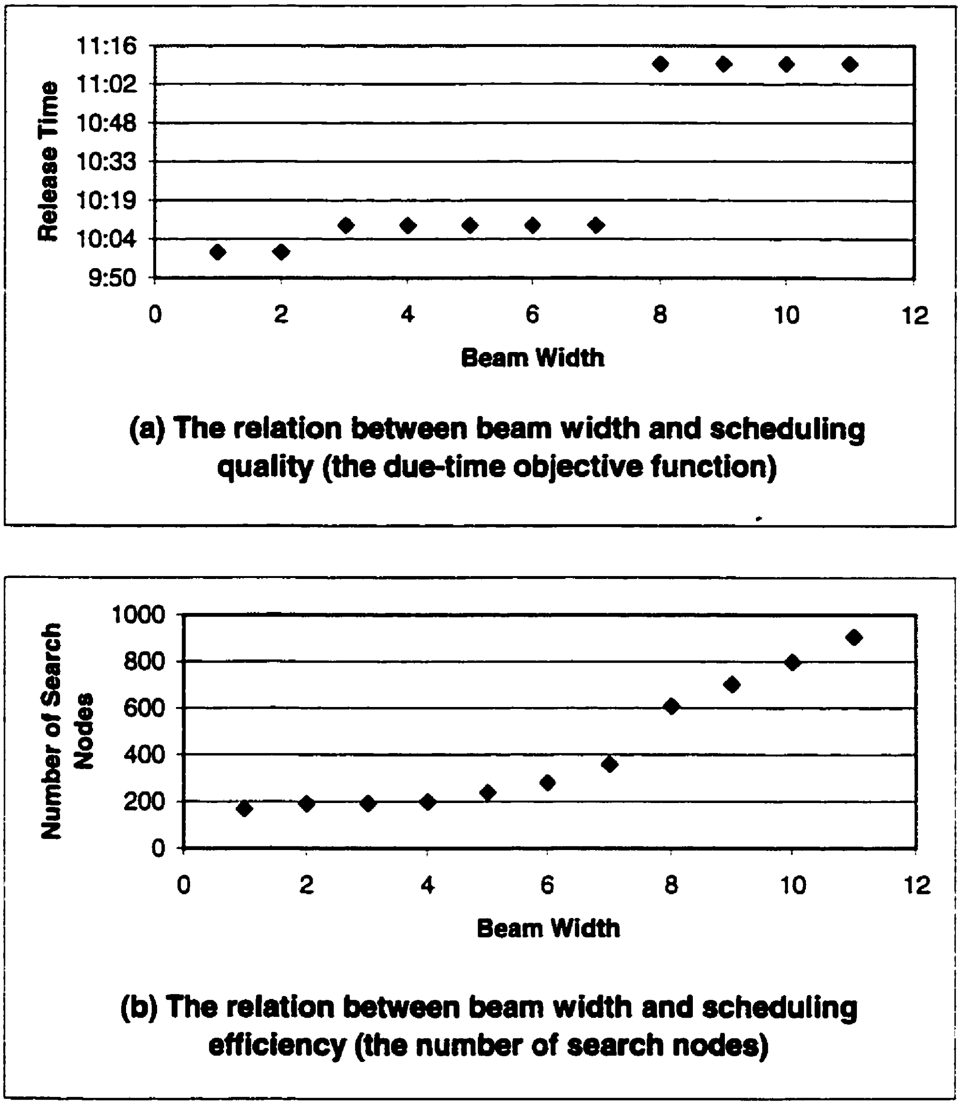


(a) The relation between beam width and scheduling quality (the earliest-delivery-time objective function)

(b) The relation between beam width and scheduling efficiency (the number of search nodes)

Figure 6.5: The system performance of predictive scheduling with the earliest-delivery-time objective function

In Chapter 5, we have provided four case studies to illustrate the reactive scheduling functions of this system. The four reactive scheduling results generated in those cases are used to evaluate the system performance of reactive scheduling.



(a) The relation between beam width and scheduling quality (the due-time objective function)

(b) The relation between beam width and scheduling efficiency (the number of search nodes)

Figure 6.6: The system performance of predictive scheduling with the due-time objective function

(1) Canceling an Old Order

In this case, as introduced in Section 5.2.3, five production orders, *Order 1*, *Order 2*, *Order 3*, *Order 4*, and *Order 5*, have been scheduled previously. Each order needs to produce a window product that is represented by *c*, an instance feature of class feature *WindowCenter*. *Order 1*, *Order 2*, *Order 3*, and *Order 4* were scheduled using the due-time based scheduling. *Order 5* was scheduled using the earliest-delivery-time based scheduling. At the current time, *10/16/98 8:00*, one of the scheduled orders, *Order 1*, is canceled based upon the customer's request. The system performance of reactive scheduling to respond to canceling an old order is illustrated in Table 6.4.

- The objective function measures for the remaining orders are improved after reschedule revision.

  The release time values for *Order 2, Order 3,* and *Order 4* are increased towards to the due time values, and the completion time value for *Order 2* is decreased backwards to the release time.
- The original schedule is partially revised for responding to canceling *Order 1*. The ratio of task revision (number of revised tasks/number of total tasks in revised schedule) is 48/60.

Table 6.5: The system performance of reactive scheduling to respond to canceling an old order

Order		1	2	3	4	5
Objective Function		D	D	D	D	E
Due Time		11:30	11:30	11:30	11:30	
Original Schedule	*Release*	9:40	9:15	8:40	8:25	8:00
	Completion	11:30	11:30	11:30	11:30	11:40
Revised Schedule	*Release*		9:40	9:10	8:40	8:00
	Completion		11:30	11:30	11:30	11:05
Number of Tasks		15	15	15	15	15
Number of Revised Tasks			13	13	13	9

D — Due-time objective function; E — Earliest-delivery-time objective function

(2) Inserting an Urgent Order

In this case, as introduced in Section 5.2.4, four production orders, *Order 1*, *Order 2*, *Order 3*, and *Order 4*, have been scheduled previously. Each order needs to produce a window product that is represented by *c*, an instance feature of class feature *WindowCenter*, as introduced in Section 4.2.2. *Order 1*, *Order 2*, and *Order 3* were scheduled by the due-time based scheduling. *Order 4* was scheduled by the earliest-delivery-time based scheduling. At the current time, *10/15/98 3:00 PM*, *Order 5* needs to be inserted with an urgent due-time requirement. The system performance of reactive scheduling for responding to inserting an urgent order is illustrated in Table 6.5.

- The objective function measures for the remaining orders are worse after schedule revision. However, the due-time requirements for the remaining orders are still satisfied.

  The release time values for *Order 1*, *Order 2*, *Order 3* are decreased, and the completion time value for *Order 4* is increased. The due-time requirements of *Order 1*, *Order 2*, and *Order 3* are still satisfied.
- The original schedule is partially revised for responding to inserting *Order 5*. The ratio of task revision (number of revised tasks/number of total tasks in revised schedule) is 47/75.

Table 6.6: The system performance of reactive scheduling to respond to inserting an urgent order

Order		1	2	3	4	5
Objective Function		*D*	*D*	*D*	*E*	*D*
Due Time		11:00	11:00	11:30		11:10
Original Schedule	*Release*	9:10	8:45	8:50	8:00	
	Completion	11:00	11:00	11:30	10:00	
Revised Schedule	*Release*	9:00	9:00	8:30	8:00	9:20
	Completion	11:00	11:00	11:30	11:40	11:10
Number of Tasks		15	15	15	15	15
Number of Revised Tasks		14	15	9	9	

D — Due-time objective function; E — Earliest-delivery-time objective function

(3) Facility Breakdowns

In this case, as introduced in Section 5.3.3, four production orders, *Order 1*, *Order 2*, *Order 3*, and *Order 4*, have been scheduled previously. A window product represented by *c*, an instance feature of class feature *WindowCenter*, as introduced in Section 4.2.2, needs to be produced in each order. *Order 1* and *Order 2* were scheduled using the due-time based scheduling, while *Order 3* and *Order 4* were scheduled using the earliest-delivery-time based scheduling. At the current time, *10/16/1998 8:30*, it is observed that the facility *FF01* will not be available from *10/16/1998 9:00* to *10/16/1998 9:20*. The system performance of reactive scheduling to respond to facility breaks is illustrated in Table 6.6.

- The objective function measures for some remaining orders are worse after schedule revision. However, the due-time requirements for the remaining orders are still satisfied.

  The release time values for *Order 1* and *Order 2* are decreased and the completion time value for *Order 4* is increased. However, the due-time requirements of *Order 1* and *Order 2* are still satisfied.

- The original schedule is partially revised for responding to facility *FF01* breakdown.

  The ratio of task revision (number of revised tasks/number of total tasks in revised schedule) is 17/60.

Table 6.7: The system performance of reactive scheduling to respond to facility breakdowns

Order		1	2	3	4
Objective Function		*D*	*D*	*E*	*E*
Due Time		11:00	11:00		
Original Schedule	*Release*	9:10	8:45	8:00	8:30
	Completion	11:00	11:00	10:00	11:15
Revised Schedule	*Release*	8:00	8:20	8:00	8:30
	Completion	11:00	11:00	10:00	11:40
Number of Tasks		15	15	15	15
Number of Revised Tasks		0	3	0	14

D — Due-time objective function; E — Earliest-delivery-time objective function

(4) Personnel Absence

In this case, as introduced in Section 5.3.4, four orders, *Order 1*, *Order 2*, *Order 3*, and *Order 4*, have been scheduled previously. A window product represented by *c*, an instance feature of class feature *WindowCenter*, as introduced in Section 4.2.2, needs to be produced in each order. *Order 1* and *Order 2* were scheduled using the due-time based scheduling, while *Order 3* and *Order 4* were scheduled using the earliest-delivery-time based scheduling. At the current time, *10/16/1998 8:30*, it is observed that the person *PM03* will not be available from *10/16/1998 8:30* to *10/16/1998 9:00*. The system performance of reactive scheduling to respond to personnel absence is illustrated in Table 6.7.

- The objective function measures for some remaining orders are worse after schedule reversion, and the due-time requirement for the order that is directly disrupted by the personnel absence cannot be satisfied in this case.

  The completion time values for *Order 3* and *Order 4* are increased. The due-time requirement of *Order 3*, which is disrupted directly by the personnel absence, cannot be satisfied. The schedule for *Order 1* is not changed in the revised schedule.

Table 6.8: The system performance of reactive scheduling to respond to personnel absence

Order		1	2	3	4
Objective Function		*D*	*D*	*E*	*E*
Due Time		11:00	11:00		
Original Schedule	*Release*	9:10	8:45	8:00	8:30
	Completion	11:00	11:00	10:00	11:15
Revised Schedule	*Release*	9:10	8:45	8:00	8:15
	Completion	11:00	11:30	11:20	12:00
Number of Tasks		15	15	15	15
Number of Revised Tasks		0	15	8	13

D — Due-time objective function; E — Earliest-delivery-time objective function

- The original schedule is partially revised for responding to facility *PM03* absence.

  The ratio of task revision (number of revised tasks/number of total tasks in revised schedule) is 38/60.

6.4 Summary

This chapter presented the implementation and evaluation of the intelligent production scheduling system. The system has been implemented using ViaualWorks/Smalltalk. In addition to the system classes of VisualWorks/Smalltalk, a number of new classes have been created for implementation of this system. The performance of the system, including scheduling quality and efficiency, was evaluated based on a simulation model for window production at Gienow Products Lit. The experimental testing results for both predictive scheduling and reactive scheduling were given in this Chapter.

A qualitative comparison of the scheduling approaches proposed in this research with conventional centralized scheduling approaches will be desirable in the future work.

Chapter 7
Conclusions and Future Work

This chapter gives conclusions of this work. The future directions of this research are also discussed in this chapter.

7.1 Conclusions

This research was devoted to the development of an intelligent production scheduling system. In this system, manufacturing requirements, including tasks and precedence constraints for accomplishing these tasks, are described using the feature-based product representation scheme. Manufacturing resources are organized using agent-based distributed system approach, in which manufacturing resources including facilities and persons are defined as agents that are coordinated by two mediators: facility mediator and personnel mediator. This system incorporates two scheduling functions: predictive scheduling and reactive scheduling. In the intelligent predictive scheduling mechanism, the optimal production task sequences and their timing parameters, and resource allocation for producing the products ordered by customers are identified based upon heuristic search and agent-based negotiation. Two intelligent reactive scheduling mechanisms are developed in this system to revise partial original schedules for responding to production demand changes and resource changes respectively. The intelligent reactive scheduling is conducted based on match-up rescheduling approach and agent-based negotiation in a multi-agent environment.

The contributions of this research are summarized below.

(1) Constraints in design and manufacturing aspects were considered in production scheduling.

In production scheduling, manufacturing constraints are defined as available manufacturing resources. In the conventional production scheduling systems, manufacturing requirements are given as production tasks, which specify their requirements for manufacturing resources including facilities and persons. These tasks are not associated with the design descriptions/constraints. In this research, an integrated scheme for representing product design and manufacturing requirements has been developed.

In this research, the manufacturing requirements for producing ordered products were described using the feature-based product modeling module. In this module, the product primitives are modeled as features. Features were described at two different levels, class level and instance level, corresponding to generic product library and special product data respectively. Instance features were generated using class features as their templates. Manufacturing requirements to produce the products, including tasks and precedence constraints for accomplishing these tasks, were modeled as part of the feature descriptions. Design constraints, such as relations among composing features of product, were described in this module and were considered in production scheduling.

Two types of manufacturing resources, facilities and persons, are considered as manufacturing constraints in this system.

(2) Intelligent predictive scheduling and reactive scheduling were implemented in a multi-agent environment.

Conventionally, the scheduling systems are developed using centralized approach, by which the manufacturing resource databases are described at the same place and accessed by a centralized control system. With the increase of the manufacturing system complexity, manufacturing activities often take place at different locations, thus requiring different knowledge bases and databases at different places. Therefore, distributed resource modeling approach is needed.

In this research, the agent-based distributed approach was used to model the manufacturing resource management module. The manufacturing resources were described as agents, including facility agents and personnel agents. The activities of agents were coordinated by two mediators, facility mediator and personnel mediator. The facility agents and personnel agents communicated and negotiated with each other through the two mediators during the scheduling process.

Among the existing intelligent scheduling approach, the traditional heuristic search based scheduling is effective to identify the optimal predictive schedule considering constraints. However, due to the large search space, improvement in scheduling efficiency, while maintaining the scheduling quality, has to be conducted. In this research, the heuristic search based scheduling was implemented in a multi-agent environment to improve the scheduling efficiency.

The reactive scheduling usually attempts to revise only partial original schedules for responding to the production environment changes without rescheduling all the scheduled tasks. To this end, two reactive scheduling mechanisms were developed in a multi-agent environment for responding to production demand changes and resource changes in order to minimize the disruption to original schedule, while to satisfy the design and manufacturing constraints.

This research can be served as a basis for associating computer-based design and manufacturing systems in an integrated environment.

7.2 Future Work

The effectiveness of production scheduling using the developed intelligent production scheduling system has been proved by the system evaluation examples. To further improve this system, the research should focus on the following issues:

(1) Integrating manufacturing requirements with more design descriptions/constraints

In this system, a feature-based integrated scheme for representing product design descriptions/constraints and manufacturing requirements has been developed. However, only the relations among composing features of the product are viewed as the design constraints in scheduling process. Since many design descriptions and constraints, such as dimensions, tolerances, and material requirements, also influence the requirements of manufacturing tasks, the system should integrate manufacturing requirements with more design descriptions and constraints.

(2) Considering more manufacturing constraints in production scheduling

Manufacturing resources are the manufacturing constraints to be considered in production scheduling. Each resource has its own characteristics and knowledge and is described as an agent, such as a facility agent or a personnel agent. Currently, the type, functions, and time constraints of each resource are considered during the scheduling process. To improve this system, more manufacturing constraints, such as manufacturing capability, manufacturing cost [Maturana 96], and manufacturing accuracy of facilities (machines) should be considered in production scheduling.

(3) Improving reactive scheduling capability

In this research, the match-up rescheduling approach was implemented in a multi-agent environment for responding to production demand changes and resource changes, including canceling an old order, inserting an urgent order, facility breakdown, and personnel absence. However, if this system is expected to be applied in actual job-shops and plants, the reactive scheduling capability should be improved for responding to more complex schedule disturbances, such as a facility breakdown with the processing part destroyed, increasing new facilities or persons in job-shop, and so on.

(4) Evaluating the proposed predictive scheduling and reactive scheduling mechanisms using the MetaMorph prototype

MetaMorph is a multi-agent architecture for development of intelligent manufacturing control system proposed by Maturana and Norrie [Maturana 96]. A prototype of MetaMorph architecture has been developed at the University of Calgary based on a computer network that are associated with the physical manufacturing system. This prototype can be used as the experimental test-bed to evaluate the efficiencies of the scheduling mechanisms proposed in this research. The qualitative comparison of the distributed scheduling approach with the centralized scheduling approach will be investigated based on the results of the testing experiments.

Bibliography

[Artiba 97]
Artiba, A. and Elmaghraby, S. E. (1997). *The Planning and Scheduling of Production Systems: Methodologies and Applications.* Chapman & Hall.

[Baker 74]
Baker, K. R. (1974). *Introduction to Sequencing and Scheduling.* John Wiley and Sons.

[Baker 95]
Baker, K. R. (1995). *Elements of Sequencing and Scheduling*, Amos Tuck School of Business Administration, Dartmouth College, Hanover.

[Bean 91]
Bean, J. C., Birge, J. R., Mittenehal, J., Noon, C. E. (1991). Match-up Scheduling with Multiple Resource, Release Dates, and Disruption. *Operation Research*, 34(8), pp. 2299-2315.

[Beck 98]
Beck, J. C. and Fox, M. S. (1998). A Generic Framework for Constraint-Directed Search and Scheduling. *AI Magazine*, 19(4), pp. 101-130.

[Bedworth 87]
Bedworth, D. D. and Bailey, J. E. (1987). *Integrated Production Control System.* John Wiley and Sons.

[Bellmore 87]
Bellmore, M. and Nemhauser, G. L. (1987). *Integrated Production Control Systems.* John Wiley and Sons.

[Booch 94]
Booch, G. (1994). *Object-Oriented Analysis and Design with Applications*, 2nd edition. The Benjamin/Cummings Publishing Company, Inc.

[Butler 95]
Butler, J. and Ohtsubo, H. D. (1995). ADDYMS: Architecture for Distributed Dynamic Manufacturing Scheduling. *Artificial Intelligence Applications in Manufacturing*, edited by Famili, A., Nau, D. S., and Kim, S. H., AAAI Press/The MIT Press, pp. 199-214.

[Camden 90]
Camden, R., Dunmire, C., Goyal, R., Sathi, H., Elm, B., and Fox, M. S. (1990). Distribution Planning: An Integration of Constraint Satisfaction and Heuristic Search Techniques. *Proceedings of the Conference on AI Applications in Military Logistics.*

[Collinot 88]
Collinot, A., Lepape, C., and Pinoteau, C. (1988). SONIA, A Knowledge-Based Scheduling System. *Artificial Intelligence Engineering*, 3(2), pp. 86-98.

[Dauzere-Peres 94]
Dauzere-Peres, S. and Lasserre, J. B. (1994). *An Integrated Approach in Production Planning and Scheduling*. Springer-Verlag.

[Dechter 91]
Dechter, R., Meiri, I., and Pearl, J. (1991). Temporal Constraint Networks. *Artificial Intelligence*, 49, pp. 61-95.

[Dean 95]
Dean, T., Allen, J., and Aloimonos, Y. (1995). *Artificial Intelligence: Theory and Practice*, Addison-Wesley Publishing Company.

[Detand 93]
Detand, J. (1993). A Computer Aided Process Planning System Generating Non-Liner Process Plans, Ph.D. Thesis, K. U. Leuven, Belgium.

[Dond 92]
Dong, J., Jo, H. H., and Parsaei, H. R. (1992). A Feature-Based Dynamic Process Planning and Scheduling. *Computers and Industrial Engineering*, 23(1-4), pp. 141-144.

[Duffie 88]
Duffie, E. H. (1988). Fault-Tolerant Heterarchical Control of Heterogeneous Manufacturing System Entities. *Journal of Manufacturing System*, 7(4), pp. 315-328.

[French 82]
French, S. (1982). *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*. John Wiley and Sons.

[Fox 83]
Fox, M. S. (1983). Constraint-Directed Search: A Case Study of Job-Shop Scheduling. Ph.D. Thesis, CMU-RI-TR-85-7, Intelligent Systems Laboratory, The Robotics Institute, Carnegie Melon University.

[Fox 94a]
Fox, M. S., Sadeh, N., and Baycan, C. (1989). Constrained Heuristic Search. *Proceedings of International Joint Conference on Artificial Intelligence*, Menlo Park, California, pp. 309-316.

[Fox 94b]
Fox, M. S. (1994). ISIS: A Retrospective. *Intelligent Scheduling*, edited by M. Zweben and M. S. Fox, Mogan Kaufmann Publishers, pp. 1-28.

[Gantt 19]
Gantt, H. L. (1919). *Organising for Work*, Harcourt, Brace and Howe (N. Y.).

[Gasser 91]
Gasser, L. (1991). Social Conceptions of Knowledge and Action: DAI Foundations and Open Systems Semantics. *Artificial Intelligence*, 47, pp. 107-138.

[Goldberg 80]
Goldberg, J. and Robson, D. (1980). *Smalltalk-80: The Language and Its Implementation.* Addison Wesley.

[Goldratt 84]
Goldratt, E. M. and Cox, J. (1984). *The Goal, A Process of Ongoing Improvement.* North River Press, Croton-on-Hudson (N.Y.).

[Graves 81]
Graves, S. C. (1981). A Review of Production Scheduling. *Operation Research*, 29(4), pp. 646-675.

[Grayer 76]
Grayer, A. R. (1976). A Computer Link between Design and Manufacture. Ph.D. Thesis, University of Cambridge, UK.

[Hopkins 95]
Hopkins, T. and Horan, B. (1995). *Smaltalk: An Introduaction to Application Development Using Visualworks*, Prentice Hall.

[Jawahar 98]
Jawahar, N., Aravindan, P., and Ponnambalam, S. G. (1998). A Genetic Algorithm for Scheduling Flexible Manufacturing System. *International Journal of Advanced Manufacturing Technology*, 14(8), pp. 588-607.

[Kott 98]
Kott, A., Saks, V., and Mercer, A. (1998). A New Technique Enables Dynamic Replanning and Rescheduling of Aeromedical Evacuation. *Proceedings of 15th National Conference on Artificial Intelligence*, Madison, WI, USA, pp. 1063-1070.

[Kumar 92]
Kumar, V. (1992). Algorithms for Constraint Satisfaction: A Survey. *AI Magazine*, 13, pp. 32-44.

[Kusiak 92]
Kusiak, A. (ed.) (1992). *Intelligent Design and Manufacturing.* John Wiley and Sons.

[Le Pape 94]
Le Pape, C. (1994). Scheduling as Intelligent Control of Decision-Making and Constraint Propagation. *Intelligent Scheduling*, edited by M. Zweben and M. S. Fox, Mogan Kaufmann Publishers, pp. 67-98.

[Maturana 96]
Maturana, F. and Norrie, D. (1996). Multi-Agent Mediator Architecture for Distributed Manufacturing. *Journal of Intelligent Manufacturing*, 7, pp. 257-270.

[Morris 94]
Morris, P. (1994). Solutions without Exhaustive Search: An Iterative Descent method for Binary constraint Satisfaction Problem. *Proceedings of the AAA-90 Workshop on Constraint-Directed Reasoning*, Boston, MA, USA.

[Moulin 96]
Moulin, B. and Chaib-Draa, B. (1996). An Overview of Distributed Artificial Intelligence. *Foundations of Distributed Artificial Intelligence*, edited by G. M. P. O'Hare and N. R. Jennings, John Wiley and Sons, pp. 3-55.

[O'Hare 96]
O'Hare, G. M. P. and Jennings, N. R. (1996). *Foundations of Distributed Artificial Intelligence*, John Wiley and Sons.

[Parunak 87]
Parunak, H. V. D. (1987). Manufacturing Experience with the Contract Net. *Distributed Artificial Intelligence*, edited by M. N. Huhns, Morgan Kaufmann, pp. 285-301.

[Parunak 91]
Parunak, H. V. D. (1991). Characterising the Manufacturing Scheduling Problem. *Journal of Manufacturing System*, 10(3), pp. 241-159.

[Parunak 96]
Parunak, H. V. D. (1996). Application of Distributed Artificial Intelligence in Industry. *Foundations of Distributed Artificial Intelligence*, edited by G. M. P. O'Hare and N. R. Jennings, John Wiley and Sons, pp. 139-164.

[Parunak 97]
Parunak, Van, Baker, A., and Clark, S. J. (1997). The AARIA Agent Architecture: An Example of Requirements-Driven Agent-Based System Design. *Proceedings of the First International Conference on Autonomous Agents (Autonomous Agent'99)*, pp. 482-483.

[Rich 91]
Rich, E. and Knight, K. (1991). *Artificial Intelligence*, McGraw-Hill, Inc.

[Rodammer 88]
Rodammer, F. A. and White, K. P. (1988). A Recent Survey of Production Scheduling. *IEEE Transaction on System, Man, and Cybernetics*, 18(6), pp. 841-851.

[Saad 95]
Saad, K., Kawamura, K., and Biswas, G. (1995). Performance Evaluation of Contract Net-Based Heterarchical Scheduling for Flexible Manufacturing System. *Proceedings of the 1995 International Joint Conference on Artificial Intelligence (IJCAI'95), Workshop on Intelligent Manufacturing*, Montreal, Canada.

[Shah 95]
Shah, J. J. and Mantyla, M. (1995). *Parametric and Feature-Based CAD/CAM*. John Wiley and Sons.

[Sharp 97]
Sharp, A. (1997). *Smalltalk by Example: The Developer's Guide*, McGraw-Hill.

[Shaw 85]
Shaw, M. J. and Winston, A. B. (1985). Task Bidding and Distributed Planning in Flexible Manufacturing. *Proceedings of IEEE International Conference on Artificial Intelligence*, pp. 184-189.

[Shaw 87]
Shaw, M. J. (1987). A Distributed Scheduling Method for Computer Integrated Manufacturing: The Use of Local Area Networks in Cellular Systems. *International Journal Production Research*, 25(9), pp. 1285-1303.

[Singh 96]
Singh, N. (1996). *Systems Approach to Computer-Integrated Design and Manufacturing*. John Wiley & Sons.

[Smith 94]
Smith, J. S. (1994). OPIS, A Methodology and Architecture for Reactive Scheduling. *Intelligent Scheduling*, edited by Zweben, M. and Fox, M. S., Mogan Kaufmann Publishers, pp. 29-66.

[Smith 80]
Smith, R. G. (1980). The Contract Net Protocol: High-Level Communication and Contract in A Distributed Problem Solver. *IEEE Transactions on Computers*, 29(12), pp. 1104-1113.

[Sprecher 94]
Sprecher, A. (1994). *Resource-Constrained Project Scheduling*. Springer-Verlag.

[Sun 99a]
Sun, J., Xue, D., and Norrie, D. H. (1999). An Intelligent Production Scheduling Mechanism Considering Design and Manufacturing Constraints. *Proceedings of Third International Conference on Industry Automation*, Montreal, pp. 24.11-24.14.

[Sun 99b]
Sun, J., Kalenchuk, D. K., Xue, D., and Gu, P. (1999). Neural Network-based Fuzzy Reasoning for Conceptual Design Evaluation. *CD-ROM Proceedings of ASME 1999 Design Engineering Technical Conferences and Computers in Engineering Conference*, Las Vegas, Nevada.

[Sycara 91]
Sycara, K. P., Roth, S., Sadeh, N., and Fox, M. (1991). Distributed Constrained Heuristic Search. *IEEE Transactions on System, Man and Cybernetics*, 21(6), pp. 1446-1460.

[Szelke 94]
Szelke, E. and Kerr, R. M. (1994). Knowledge Based Reactive Scheduling – State of Art. *International Journal of Production Planing and Control*, 5, pp. 124-145.

[Szelke 95]
Szelke, E. and Monostori, L. (1995). Reactive and Proactive Scheduling with Learning in Reactive Operation Management. *Proceedings of IFIP International Working Conference on Managing Concurrent Manufacturing to Improve Industrial Performance*, Seattle, Washington, USA, pp. 456-483.

[Tharumarajah 97a]
Tharumarajah, A. and Bemelman, R. (1997). Approaches and Issues in Scheduling a Distributed Shop-floor Environment. *Computer in Industry*, 34, pp. 95-109.

[Tharumarajah 97b]
Tharumarajah, A. and Wells, A. J. (1997). A Behaviour-Based Application to Scheduling in Distributed Manufacturing Systems. *Integrated Computer Aided Engineering*, 4, pp. 235-249.

[Tsukada 96]
Tsukada, T. K. and Shin, K. G. (1996). PRIAM: Polite Rescheduling for Intelligent Automated Manufacturing. *IEEE Transactions of Robotics and Automation*, 12(2), pp. 235-245.

[Valckenaers 93]
Valckenaers, P. (1993). Flexibility for Integrated Production Automation. Ph.D. Thesis, K. U. Leuven, Belgium.

[Van Hentenryck 89]
Van Hentenryck, P. (1989). *Constraint Satisfaction in Logic Programming,* MIT Press.

[Vickers 88]
Vickers, D. L. and Swanson, K. A. (1988). A Form Feature-Centered Architecture for Product Definition Exchange. *AUTOFACT'88 Conference Proceedings*, pp. 25-37.

[Winston 92]
Winston, P. H. (1992). *Artificial Intelligence*, 3rd edition, Addison-Wesley.

[Xue 92]
Xue, D., Takeda, H., Kiriyama, T., Tomiyama, T., and Yoshikawa, H. (1992). An Intelligent Integrated Interactive CAD. *Intelligent Computer-Aided Design,* edited by Waldron, M. B., Brown, D, and Yoshikawa, H., North-Holland, Amsterdam, pp. 163-192.

[Xue 93]
Xue, D. and Dong, Z. (1993). Feature Modeling Incorporating Tolerance and Production Process for Concurrent Design. *Concurrent Engineering: Research and Applications*, 1, pp. 107-116.

[Xue 94]
Xue, D. and Dong, Z. (1994). Developing a Quantitative Intelligent System for Implementing Concurrent Engineering Design. *Journal of Intelligent Manufacturing*, 5, pp. 251-267.

[Xue 99a]
Xue, D., Yadav, S., and Norrie, D. H. (1999). Development of an Intelligent System for Building Product Design and Manufacturing - Part I: Product Modeling and Design. *Proceedings of the Second International Workshop on Intelligent Manufacturing System*, Leuven, Belgium, pp. 367-376.

[Xue 99b]
Xue, D., Sun, J., and Norrie, D. H. (1999). Development of an Intelligent System for Building Product Design and Manufacturing – Part II: Production Process Planning and Scheduling. *Proceedings of the Second International Workshop on Intelligent Manufacturing System*, Leuven, Belgium, pp. 57-66.

[Xue 99c]
Xue, D., Yadav, S., and Norrie, D. H. (1999). Knowledge Base and Database Representation for Intelligent Concurrent Design. *Computer-Aided Design*, 31, pp. 131-145.

[Yadav 98]
Yadav, S. (1998). Development of a Feature-based Intelligent Design System. M.Sc Thesis, The University of Calgary.

[Yadav 99]
Yadav, S., Xu, Y., and Xue, D. (1999). A Multi-Level Heuristic Search Algorithm for Production Scheduling. *CD-ROM Proceedings of ASME 1999 Design Engineering Technical Conferences and Computers in Engineering Conference,* Las Vegas, Nevada.

[Zweben 92]
Zweben, M., Davis, E., Daun, B., Drascher, E., Deal, M., and Eskey, M. (1992). Learning to Improve Constraint-based Scheduling. *Artificial Intelligence,* 58, pp. 271-296.

[Zweben 94]
Zweben, M. and Daun, B., and Deale, M. (1994). Scheduling and Rescheduling with Iterative Repair. *Intelligent Scheduling*, edited by Zweben, M and Fox, M. S., Mogan Kaufmann Publishers, pp. 241-256.