CHAPTER 65
Monitoring and Controlling Operations

ALBERT JONES
National Institute of Standards and Technology

YUEHWERN YIH
Purdue University

EVAN WALLACE
National Institute of Standards and Technology

1. INTRODUCTION

On the surface, companies from industry sectors such as mining, construction, manufacturing, and service would appear to be very different. Certainly, the physical activities involved and the resulting products are different. At a conceptual level, however, each of these companies can be viewed as a

 Handbook of Industrial Engineering: Technology and Operations Management, Third Edition.
Edited by Gavriel Salvendy

complex system trying to utilize its resources to maximize its performance. The ability of managers and workers to control and monitor the operations within such a system has a great impact on its performance.

This chapter addresses three issues related to controlling and monitoring operations in such a system: architectures to organize those operations; artificial intelligence techniques for scheduling those operations; and commercial software to implement monitoring and control. The principal application focus of this chapter is manufacturing, but the ideas can be applied to a wide range of complex systems.

2. CONTROL ARCHITECTURES

Decisions, decisions, decisions—factory managers make them and factory workers implement them everyday. Some decisions impact events immediately; others impact events months or years into the future. Industry, academia, government agencies, and standards bodies have expended considerable effort to develop architectures that (1) organize and integrate these decisions in some meaningful way and (2) specify the information required to make those decisions monitor their execution, and control their implementation. This section describes two such architectures.

2.1. The Purdue Enterprise Reference Architecture (PERA)

A tree structure, hierarchy, is one of the most common ways of organizing functions and activities. Many such hierarchies have been proposed for decision-making and control within manufacturing systems (Jones 1990). The Purdue Enterprise Reference Architecture (PERA), which was developed by a collection of industrial and academic representatives, is one such organization of the factory that includes the control functions and information requirements (Williams 1992). Originally aimed at the process industry, it has been developed so that it can be used across all types of manufacturing. The material in the following sections is taken from Annex D and Section 5.1 of ANSI/ISA-S95.00.01-2000, *Enterprise-Control System Integration Part 1: Models and Terminology* (ANSI/ISA 2000).

2.1.1. Control Hierarchy

Figure 1 shows three levels of the PERA functional hierarchy model at which decisions are made: business planning and logistics, manufacturing operation, and control. Level 4 and level 3 deal with plant production scheduling, operation management, and plant floor coordination. Levels 2, 1, and 0 decompose control functions for three types of manufacturing: batch, continuous, and discrete. This decomposition defines the cell or line supervision functions, operations functions, and process control functions. There are several different execution methods for these functions, which are based on the actual production strategy used.

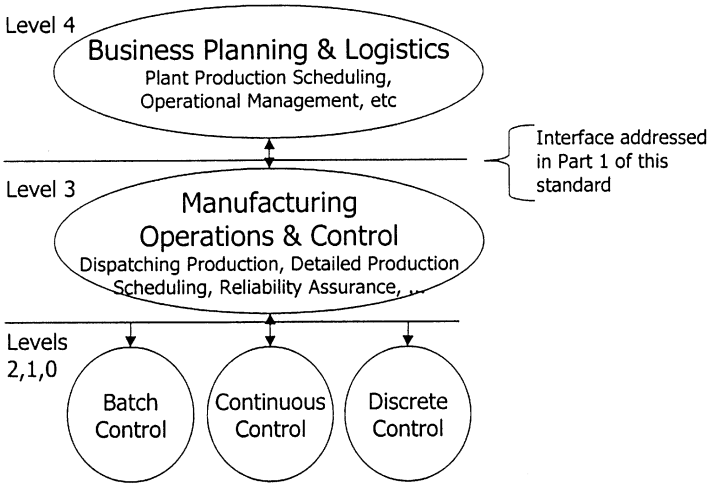


Figure 1 Decision-Making and Control Hierarchy.

Level 4 performs the following major functions: capacity planning, plant production scheduling, materials requirements planning, and all manufacturing-related purchasing. It establishes and modifies the basic plant production schedule for orders received, based on resource availability changes, energy sources available, power demand levels, and maintenance requirements. It develops preventive maintenance and equipment renovation schedules in coordination with the basic production schedule. It determines the optimum inventory levels of raw materials, energy sources, spare parts, and in-process goods. Finally, it collects, maintains, and provides data on a large number of items. These items include raw material and spare parts usage, overall energy use, goods in process, inventory, control files as they relate to customer requirements, machinery and equipment utilization and history files, and manpower use data for transmittal to personnel and accounting.

Level 3 is the principal level of interest in this chapter. Therefore, we provide detailed descriptions of the major activities performed at this level.

- *Resource allocation and control:* Manage those resources directly associated with control and manufacturing. These resources include machines, tools, labor skills, materials, and other equipment, documents, and entities that must be available for work to start and be completed. The management of these resources may include local resource reservation to meet production-scheduling objectives, the assurance that equipment is properly set up for processing, the responsibility for providing real-time, resource status and a history of resource use.
- *Dispatching:* Manage the flow of production in the form of jobs, orders, batches, lots, and work orders by dispatching production to specific equipment and personnel. The flow is governed by the sequence of operations, which determines the order the work is done, and the time that work starts and stops. It is possible to change the sequence or times in real time as events occur on the factory floor; however, those changes are made within agreed upon limits, based on local availability and current conditions. Dispatching of production includes the ability to control the amount of work in process through buffer management and management of rework and salvage processes.
- *Data collection and acquisition:* Manage the operational production and parametric data that is associated with the production equipment and production processes, provide real-time status of the equipment and processes, and keep a history of production and parametric data.
- *Quality management:* Provide real-time measurements collected from manufacturing and analysis in order to ensure proper product quality control and identify problems requiring attention. This includes SPC/SQC tracking and management of off-line inspection operations and analysis in laboratory information management system (LIMS). This activity may recommend actions to correct the problem, including correlating the symptoms, actions, and results to determine the cause.
- *Process management:* Monitor production and provide decision support to operators who correct and improve in-process functions. These functions may be intraoperational (focusing specifically on machines or equipment being monitored and controlled) or inter-operational (tracking the process from one operation to the next). It may include alarm management to alert factory personnel of process changes that are outside of acceptable tolerances.
- *Production planning and tracking:* Provide the status of production and the disposition of work. Status information may include personnel assigned to the work, component materials used in production, current production conditions, and any alarms, rework, or other exceptions related to the product.
- *Performance analysis:* Provide up-to-the-minute reporting of actual manufacturing operations results, along with comparisons to past history and expected results. Performance results include such measurements as resource utilization, resource availability, product unit cycle time, conformance to schedule, and performance to standards. Performance analysis may include SPC/SQC analysis, and may draw from information gathered by different control functions that measure operating parameters.
- *Operations and detailed scheduling:* Generate sequences that optimize some objective (such as minimize set-up time) based on priorities, attributes, characteristics, and production rules associated with specific production equipment and specific product characteristics. This activity is carried out using the current estimate of unused capacity and recognizing alternative and overlapping/parallel operations.
- *Document control:* Control records and forms that must be maintained with the production unit. The records and forms include work instructions, recipes, drawings, standard operation procedures, part programs, batch records, engineering change notices, shift-to-shift communication, as well as the ability to edit ''as-planned'' and ''as-built'' information. This activity is responsible for providing data to operators and recipes to device controls, and for maintaining the integrity of regulatory, environmental, health and safety regulations, and SOP information such as corrective action procedures.

- *Labor management:* Provide status of personnel in real time. This activity includes time and attendance reporting, certification tracking, as well as the ability to track indirect functions such as material preparation or tool room work as a basis for activity-based costing. It may interact with resource allocation to determine optimal personnel assignments.
- *Maintenance management:* Maintain equipment and tools. This activity ensures the availability of equipment and tools for manufacturing, and manages a history of past events or problems to aid in diagnosing problems.

2.1.2. Equipment Organization

The hierarchy described above deals with the decision making, control, and information needed to manage the physical assets of a manufacturing enterprise. Those assets are usually organized in a tree structure such as the one described in Figure 2. Lower-level groupings are combined to form higher-level entities. In some cases, a grouping within one level may be incorporated into another grouping at that same level. In the following section, we define the areas of responsibility for the different levels defined in the hierarchical model and some of the objects used in the information exchanged within and across those levels.

- *Enterprise:* A collection of one or more sites. The enterprise is responsible for determining the products to be manufactured, where they will be manufactured, and in general how they will be manufactured. Level 4 functions are generally dealing at the enterprise and site levels. However, enterprise planning and scheduling may involve areas, cells, lines, or units within an area.
- *Site:* A physical, geographical or logical grouping determined by the enterprise. It may contain areas, production lines, process cells, and production units that have well defined manufacturing capabilities. The level 4 functions at a site include local site management and optimization. Sites are often used for rough-cut planning and scheduling, which may involve cells, lines, or units within the areas.
- *Area:* A physical, geographical, or logical grouping, which may contain process cells, production units, and production lines. The main production capability and geographical location within a site usually identify areas. Areas generally have well defined manufacturing capabilities and capacities that are used for planning and scheduling at levels 3 and 4. An area is made up of lower-level elements that perform continuous manufacturing operations, discrete (repetitive and nonrepetitive) manufacturing operations, and batch manufacturing operations. An area may have several of these elements in varying combinations, depending upon the manufacturing requirements. For example, a beverage manufacturer may have an area with continuous mixing equipment that feeds a batch process cell for batch processing that feeds a bottling line for discrete bottling process. Depending on the planning and scheduling strategy selected, the level 4 func-

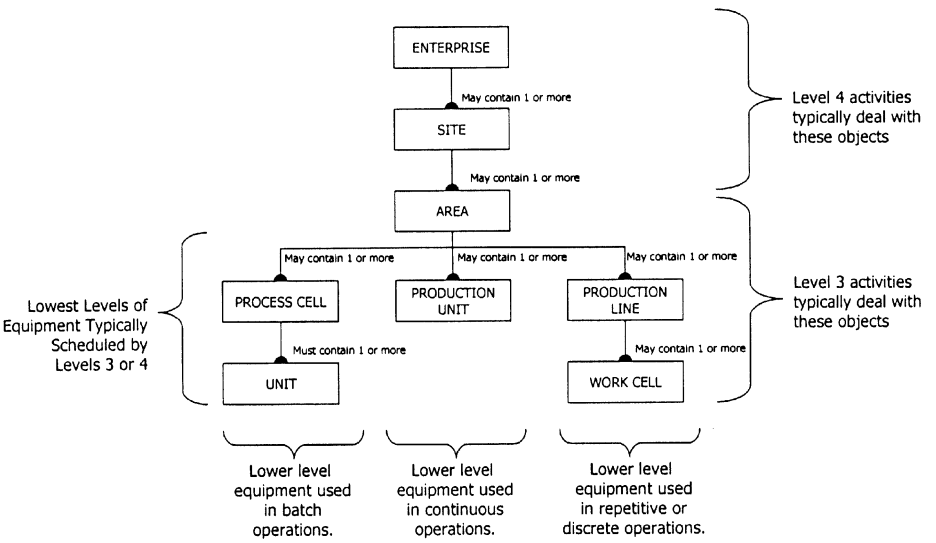


Figure 2 Typical Equipment Organization.

tions may stop at the area level, or they may schedule the functions of the lower-level elements within the areas.

- *Production units:* The lowest level of equipment typically scheduled by the level 4 or level 3 functions for continuous manufacturing processes. Production units are composed of lower level elements, such as equipment modules, sensors, and actuators. Production units have well-defined processing capabilities, and throughput capacities, and these are used for level 3 functions. The capacities and capabilities are also often used as input to level 4 scheduling, even if the production units are not scheduled by the level 4 functions.
- *Production lines and work cells:* The lowest levels of equipment typically scheduled by the level 4 or level 3 functions for discrete manufacturing processes. Work cells are usually only identified when there is flexibility in the routing of work within a production line. Production lines and work cells may be composed of lower level elements. Production line and work cells have well defined manufacturing capabilities and throughput capacities and these are used for level 3 functions. The capacities and capabilities are also often used as input to level 4 scheduling, even if the production lines and work cells are not scheduled by the level 4 functions.
- *Process cells and units:* The lowest level of equipment typically scheduled by the level 4 and level 3 functions for batch manufacturing processes. Units are usually only identified at levels 3 and 4 if there is flexibility in the routing of product within a process cell. The definitions for process cells and units are contained in the IEC 61512 and ISA S88.01 standards. Process cells and units have well-defined manufacturing capabilities and batch capacities, and these are used for level 3 functions. The capacities and capabilities may also be used as input data for level 4 scheduling, even if the process cells or units are not scheduled by the level 4 functions.

2.1.3. Status

The PERA plays a critical role in two standards, one being developed by SP95 of the ISA (Instrument Society of America) and the other by TC 184/WG 1 of the ISO (International Standards Organization). SP95 seeks to create standards for the interfaces between control functions and other enterprise functions (http://www.isa.org/sc/committee/1,1512,145,00.html) based upon the PERA. The interface initially considered is the interface between levels 3 and 4 of that model. Additional interfaces will be considered, as appropriate. WG 1 has published the PERA as an annex (http://www.nist.gov/sc5wg1/gera-std/15704fds.htm) to ISO 15704, *Requirements for Enterprise Reference Architectures and Methodologies,* which is a final draft international standard (http://www.nist.gov/sc5wg1/gera-std/15704AB.htm). From the WG1 perspective, PERA is an example of a generalized enterprise reference architecture, GERAM. A GERAM defines a toolkit of concepts for designing and maintaining enterprises for their entire life history and is meant to organize existing applications in all types of enterprises. An advantage is that previously published reference architectures can keep their own identity while identifying through GERAM their overlaps and complementing benefits compared to others.

2.2. SEMATECH CIM Framework

The CIM Framework, developed by SEMATECH, defines a standard component architecture and application component interfaces for manufacturing information and execution systems (MIES) software (Doscher 1998)—the following material is taken largely from (Hawker 1999). The CIM Framework is not hierarchical in nature; rather, it leverages distributed, object-oriented computing technology. Additionally, it uses middle-ware standards from the Object Management Group (OMG) (http://www.omg.org) to enable integration of the applications.

The CIM Framework software architecture was designed to enable the following capabilities:

- *Integration:* Applications can cooperate by exchanging data, providing services (client/server method invocation), publishing service exceptions, and publishing and subscribing to events.
- *Interoperability:* Applications from one supplier or source can be replaced easily with a functionally equivalent application (conformant to standard interface and behavior) from another source.
- *Flexibility:* Components and applications can be configured in a variety of ways that meet specific needs.
- *Reuse:* New systems can be implemented from standard components or applications more quickly, at lower cost, and with higher quality.

The major benefit of this framework is a significant reduction in the cost and time involved in building, modifying, and enhancing MIES software in response to changing business needs. Adherence to the framework allows semiconductor manufacturers to integrate applications from multiple suppliers with their legacy systems and to replace or upgrade these applications and systems over time.

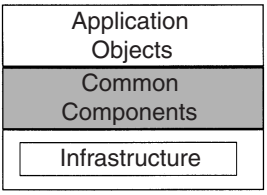


Figure 3 CIM Framework Architecture Layers.

2.2.1. CIM Framework Component Architecture

The CIM Framework architecture is a layered system that enables distributed, object-oriented applications assembled from common software components to interoperate as a single, integrated system.

The integration infrastructure is based on specifications in the Object Management Architecture (OMA) from the OMG (OMG 1995, 1996, 1997a). These specifications define standard services for distributed object communications, persistence, transactions, name services, and so forth. On top of the infrastructure, the CIM Framework architecture defines common application components. Components are software building blocks that implement some collection of functions. Typically, each MIES application will be composed of many such components. The CIM Framework common components layer defines standard models for application components that are common across MIES applications. Examples include a machine management component, a product management component, and a person management component. The machine management component includes machine resources, sensors, process capabilities, and relations to material and recipes. The product management component includes product material, lots, and relations to product and process specifications. The person management component includes persons, qualifications, and relations to skills and skill requirements. This common application model, defined in terms of common software components, is the *framework* for building integrated MIES applications.

The application objects layer of the CIM Framework architecture provides additional functionality, extending the common components to make a complete MIES. This layer, which is identified but not specified, enables MIES suppliers and users to define product-specific and site-specific application objects and components that use and extend the CIM Framework common components to implement MIES functions that meet business needs.

A given MIES application, as shown in Figure 4, implements some common components and application objects and interoperates (via the infrastructure) with other common components and

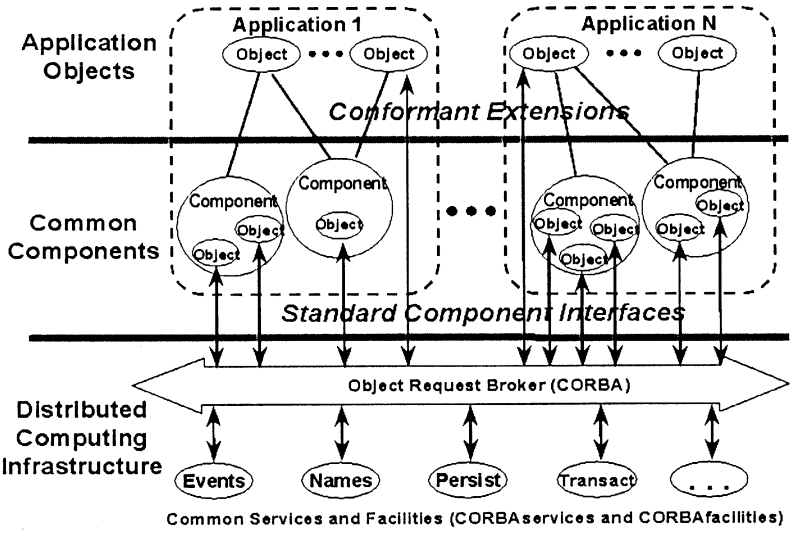


Figure 4 CIM Framework Component Architecture.

application objects implemented in other MIES applications. The collection of interoperating MIES applications provides a complete, integrated, MIES solution.

2.2.2. Component-Specification Methodology

The key to the CIM Framework is the specification of components. The CIM Framework uses the following modeling methods to specify components:

- Component relationship models showing interaction between "medium-grained" components (larger than an object, smaller than an application)
- Component information models showing object interfaces and relationships in the form of OMT (object modeling technique) diagrams (Rumbaugh et al. 1991)
- Object interface definitions using OMGs Interface Definition Language (IDL)
- Published and subscribed events using an extension to OMG IDL
- Component interaction diagrams showing scenarios that trace messages and events between components
- State transition diagrams as Harel state charts (Harel 1987) and state definition tables

These modeling methods go far toward specifying components that MIES implementers can "plug-and-play" into integrated systems. SEMATECH is also working in the OMG Business Objects Domain Task Force to define and standardize additional methods for even richer semantic models, including the specification of method preconditions and postconditions, roles, rules, and dependencies (http://www.omg.org/homepages/bodtf/).

2.2.3. Shop-Floor Application Modules

The CIM Framework specifies application components for manufacturing information and execution systems (MIES). MIES perform factory operations functions in the context of enterprise information and control systems and systems that automate material processing, storage and movement. Figure 5 shows the MIES functional groups in the CIM Framework.

Each functional group defines a collection of related application components. Table 1 lists the CIM Framework components in each functional group. The functional groups are a convenient mechanism to organize the CIM Framework components; they are not rigid partitions, and suppliers can deliver applications that span functional groups or that implement only some of the components of a group. In contrast, the component is the smallest-grained entity that suppliers can deliver. A supplier must implement all the interfaces and behaviors of a component in order to claim conformance to that component specification.

The value and power of the CIM Framework is the application model, which specifies medium-grained components common to MIES applications. The SEMATECH CIM Framework Specification Version 2.0 (Doscher 1998) has almost 300 pages of detailed component models specified using the

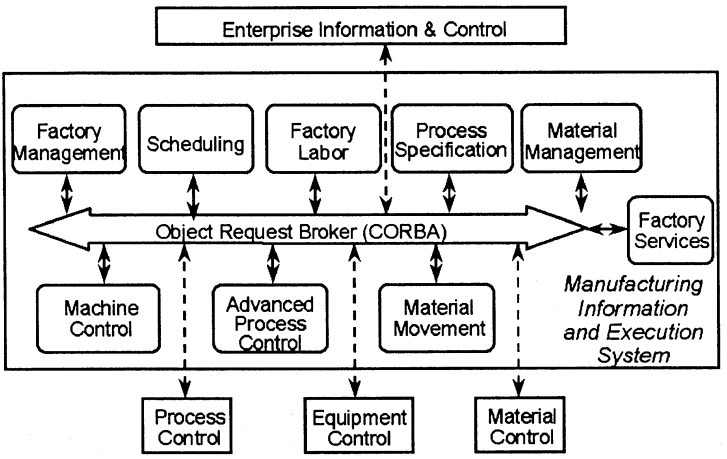


Figure 5 Functional Groups for MIES.

TABLE 1 CIM Framework Application Components

Factory Services	Material Management	Process Specification Management
Document management	Product management	Process specification
Version management	Durable management	Process capability
History management	Consumable management	
Event broker	Product specification	*Schedule management*
	Bill of material	Dispatching
Factory management		
Factory	*Advanced process control*	*Machine control*
Product release	Plug-in management	Machine management
Factory operations	Plug-in execution	Recipe management
	Control management	Resource tracking
Factory labor	Control execution	
Person management	Control database	*Material movement*
Skill management	Data collection	Material movement

methodology of Section 2.3. Section 4 presents a portion of the CIM Framework Product Management component to illustrate the style and detail of the specification. By developing industry-wide consensus on these component definitions, SEMATECH has enabled manufacturers quickly and cost effectively to build, modify and enhance MIES by assembling standards-conformant components from multiple suppliers.

2.2.4. Status

Six of the CIM Framework specifications have become SEMI (Semiconductor Equipment and Materials International) standards (http://www.semi.org/web/wstandards.nsf/). These standards were developed with the cooperative efforts of both users and suppliers of semiconductor MES software systems. Active efforts were underway to add another standard for the CIM Framework Scheduling Component by the end of 2000, and there is interest in working on factory operations and production machine components (Hodges 2000). While there may be few instances of fully compliant MES products being shipped, we believe that vendors are using the standards in their product-development process and plans. The customers of these MES products are also using the adopted standards in integration efforts that combine best-in-class components into working factory systems.

3. AI APPROACHES TO SHOP-FLOOR SCHEDULING AND CONTROL

In the preceding section, we described two architectural approaches for organizing functions related to shop-floor scheduling control. One of the most important of these functions is scheduling. Another chapter reported on two major approaches to solving these problems: mathematical programming and heuristics. In this chapter, we describe a number of AI (artificial intelligence) techniques.

3.1. Knowledge-Based Systems

Expert and knowledge-based systems were quite prevalent in the early and mid-1980s. They have four main advantages. First, and perhaps foremost, they can use both quantitative and qualitative knowledge in the decision-making process. Second, they are capable of generating heuristics that are more complex than the simple dispatching rules described in an earlier chapter. Third, the selection of the best heuristic can be based on information about the entire shop, including current jobs, expected new jobs, and the status of all resources. Fourth, they capture complex relationships in elegant, new data structures and contain special techniques for manipulation of these data structures. They are, however, time consuming to build and verify, difficult to maintain and change, and generate only feasible solutions that can be quite far from the optimum.

3.1.1. Generating the Required Knowledge Base

Formalizations of the knowledge that human experts use—rules, procedures, heuristics, and other types of abstractions—are captured in the knowledge base. Three types of knowledge, procedural, declarative, and meta, are usually included. Procedural knowledge is domain-specific, problem-solving knowledge. Declarative knowledge provides the input data that defines the problem domain. Metaknowledge is knowledge about how to use the other knowledge to solve the actual problem. Several data structures have been proposed to represent the knowledge, including semantic nets, frames, scripts, predicate calculus, and production rules. The inference engine implements a strategy

to apply to the knowledge to obtain a solution to the problem at hand. It can be forward chaining (data driven) or backward chaining (goal driven).

The first step in developing a knowledge base is knowledge acquisition, which is a two-step process: get the knowledge from knowledge sources and store that knowledge in digital form. Knowledge acquisition, such as protocol analysis, machine learning, and interactive editing (Shaw et al. 1992), has been an active area of research. Knowledge sources may be human experts, simulation data, experimental data, databases, and documents. In scheduling problems, the knowledge sources are likely to be human experts or simulation data. To extract knowledge from these two sources, any technique that learns from examples (data) becomes a promising tool. Inductive learning, which is a state classification process, is one such technique. If we view the state space as a hyperplane, training data (consisting of conditions and decisions) can be represented as points on that hyperplane. The inductive learning algorithm seeks to draw lines that divide the hyperplane into several areas within which the same decision (conclusion) will be made.

Quinlan (1986) developed an algorithm, which implements the inductive learning paradigm, called Iterative Dichotomister 3 (ID3). ID3 uses examples to induce production rules (e.g., if-then) that form a simple decision tree. Decision trees are one way to represent knowledge for the purpose of classification. The nodes in a decision tree correspond to attributes of the objects to be classified, and the arcs are alternative values for these attributes. The end nodes of the tree (leaves) indicate classes to which groups of objects belong. Each example is described by attributes and a resulting decision. To determine a good attribute, which is the basis for an object-class partitioning, entropy is employed. Entropy is a measure of the information content of each attribute. Rules are derived through a repetitive decomposition process that minimizes the overall entropy.

The attribute with the minimum entropy value will be selected as a node in the decision tree. The arcs out of this node represent different values of this attribute. If all the objects in an arc belong to one class, the partition process stops. Otherwise, another attribute will be identified using entropy values to partition further the objects that belong to this arc. This partition process continues until all the objects in an arc are in the same class. Before this algorithm is applied, all attributes that have continuous values need to be transformed to discrete values.

In the context of job shop scheduling, the attributes represent system status and the classes represent the dispatching rules. Very often the attribute values are continuous. Yih (1990) proposed a trace-driven knowledge-acquisition (TDKA) methodology to deal with continuous data and avoid the problems that occur interviewing human experts. TDKA learns scheduling knowledge from expert schedulers without resorting to an interview. There are three steps. In Step 1, an interactive simulator is developed to model the system of interest. The expert will interact with this simulator and make decisions. The entire decision-making process will be recorded in the simulator and can be repeated for later analysis. The pair, system information and scheduling decision, is called a trace. Step 2 analyzes the trace and proposes classification rules to partition the trace into groups. The partition process stops when most of the cases in each group use the same dispatching rule (error rate is below the threshold defined by the knowledge engineer). Then the decision rules are formed. The last step is to verify the generated rules. The resulting rule base is used to schedule jobs in the simulator. If it performs as well as or better than the expert, the process stops. Otherwise the threshold value is increased and the process returns to Step 2. This approach was applied in an electroplating process line and the rule base system outperforms the users. Later, Yih (1994) developed a prolog-based controller that handles the time-window problems in the same manufacturing environment.

3.1.2. Applications to Scheduling and Control

ISIS (Fox 1983) was the first expert system aimed specifically at job shop scheduling problems. ISIS used a constraint-directed-reasoning approach with three constraint categories: organizational goals, physical limitations, and causal restrictions. Organizational goals specified five objective functions based on due date and work-in-progress. Physical limitations specified the processing capability of each resource. Casual restrictions included all procedural constraints and resource requirements. Several issues related to these constraints were considered, such as conflicts among constraints, relative importance of constraints, and interactions of constraints. ISIS used a three-level, hierarchical, constraint-directed search. Orders were selected at level 1. Capacity analysis was performed at level 2 to determine the availability of the resources required by the order. Detailed scheduling was performed at level 3, to assign times to the resources identified at level 2. ISIS utilized its constraint knowledge to maintain the consistency of the schedule and identify scheduling decisions that would result in poorly satisfied constraints. It also included the capability to construct and alter schedules interactively. Chiu and Yih (1995) proposed a learning-based approach for dynamic scheduling in a distributed manufacturing system. An incremental approach to training a decision tree is proposed in this study. Each training sample consists of system attributes as inputs and a dispatching rule as its output. In their work, simulations are conducted first to collect some scenarios, and then the genetic algorithm

is performed to search a good dispatching rule for each scenario. The learning algorithm is then applied to obtain a decision tree for dynamic selection of scheduling rules.

Chang (1996) proposed a fuzzy-based methodology to control the number of kanbans in a generic kanban system. In this approach, the simulated annealing algorithm is employed to find the near-optimal number of kanbans for different system status, and thereafter a training instance is generated. Then the proposed fuzzy system will be generated for dynamic kanban control. Other work in developing a control system includes Huang and Chang (1992), Gupta et al. (1989), Chandra and Talavage (1991), and Talavage and Shodhan (1992).

Adachi et al. (1988) proposed a pattern-recognition-based method for controlling a multiloop production system. In their proposed approach, a state table is constructed for the control-decision support system (CDSS) based on the simulation results. After the user indicates the desired level and importance weight for each performance measure, the one with shortest distance to the desired pattern will be selected by the control system and the associated performance level will be displayed. These procedures are repeated until the user is satisfied with the expected performance levels. The authors further constructed a rule-based decision support system (RBDSS) to control the same production system and compared the performance of CDSS and RBDSS (Adachi et al. 1989).

Several researchers have attempted to use the knowledge-based approach to model the shop-floor control problem (Farhoodi 1990; Pluym 1990; Adachi et al. 1989). Under this approach, a central database with several production rules handles scheduling and monitors system status. Each production rule consists of a condition part and an action portion with a form of an if-then clause. Typically, these rules are based on the simulation results from different scenarios or the knowledge from the experience of schedulers. When a decision-making point is encountered, the database is scanned to find the condition that could match the current situation and the associated action is then executed. However, it is not easy to generate a database consisting of every possible situation for a system. Besides, if this database is large or the production rules are complex, it will take a long time to search the database and it is impractical for real-time implementation.

O'Grady and Lee (1988) proposed a cell control system, called PLATO-Z, by using a rule-based expert system and a multiblackboard/actor model. In the proposed control system, the major functions are performed by four blackboard subsystems: scheduling, operation dispatching, monitoring, and error handling. Adequate messages are passed between blackboard subsystems in order to achieve the control requirements. This control framework was further implemented by an object-oriented programming technique (O'Grady and Seshadri 1992).

Wu and Wysk (1988, 1989) also proposed a multipass, expert control system for flexible manufacturing cells. Under their proposed system, some candidate rules are selected by a knowledge-based system and then the performance of each candidate rule is evaluated through simulation. Weighted objective values are compared in order to achieve the multicriterion objective. Cho and Wysk (1993) then refined it by using a neural network instead of the knowledge-based system for selecting the candidate rules in the initial stage.

3.1.3. Agents to the Rescue?

It is difficult to use expert and knowledge-based systems to solve large, real-world scheduling problems because of their limited knowledge and problem solving abilities. To address this, AI researchers have used the "divide and conquer" approach to develop distributed scheduling approaches (Parunak et al. 1985). This requires a technique to decompose the scheduling problem and a collection of associated knowledge-based systems that cooperate to solve the overall problem (Zhang and Zhang 1995). Cooperation is handled through an agent paradigm. Each agent is a complete knowledge-based system with its own long-term knowledge, solution-evaluation criteria, languages, algorithms, and hardware requirements. A multiagent system is created by integrating agents selected from a "library" of agents.

For example, one such multiagent system could involve two types of agents: tasks and resources. Each task agent might schedule a certain class of tasks, such as material handling, machining, or inspection, on those resources capable of performing such tasks. The schedule is generated using any task-related performance measure, such as minimize tardiness. The schedules generated by task agents become goals for the resource agents (Daouas et al. 1995). Each resource agent schedules tasks for its assigned resource(s) using resource-related performance measures, such as maximize utilization. Each resource agent will use its schedule to decide whether it can meet the scheduling goals set by the task agents. Clearly, a situation can arise where no resource will accept a given task; coordination mechanisms must be developed to avoid this situation.

While there is promise for these types of agent-based approaches, there are no general guidelines for the design and implementation of such approaches.

3.2. Artificial Neural Networks

Neural networks, also called connectionist or distributed/parallel processing models, have been studied for many years in an attempt to mirror the learning and prediction abilities of human beings.

Neural network models are distinguished by network topology, node characteristics, and training or learning rules. They are important because they can match current shop status and the desired performance measures to near-optimal scheduling strategies and they can learn (Yih and Jones 1992).

Among the many network topologies and learning algorithms, the Hopfield network and the multilayer perceptron are preferred by several researchers for scheduling problems. Therefore, in the following sections, these two networks will be briefly discussed, and the related works on scheduling problems will be reviewed.

3.2.1. Hopfield Networks

A Hopfield network consists of nodes that are fully connected to each other bidirectionally. Instead of a continuous value, this network takes only the binary or bipolar value as its input. In addition, it is also regarded as a symmetrically weighted network because the weights on the links between nodes are the same in both directions. When an input pattern is applied, the Hopfield network will adjust the weights until it converges to a stable state. This happens when the output value of each node is no longer changed. In other words, the network will reduce its "energy" until it stabilizes in a hollow of the energy landscape.

Foo and Takefuji (1988a, b) used the Hopfield network to solve job shop scheduling problems. The scheduling problem was first mapped into a two-dimensional matrix representation. Feasibility constraints and performance measures were then formulated as the energy function, named cost function. The characteristic of this energy function is that it will result in very large value when the schedule is not feasible or the performance is far from expectations. The solution is obtained by reducing the energy in the network. The authors concluded that this approach could produce near-optimal solutions, though the optimality was not guaranteed. In addition, it was claimed that the proposed approach would not be feasible in a large-scale problem.

Zhou et al. (1991) modified this approach by using a linear cost function and concluded that this modification not only produced better results but also reduced network complexity. Other works related to using the Hopfield network for the scheduling problem include Zhang et al. (1991) and Arizono et al. (1992).

3.2.2. Supervised-Learning Neural Networks

Through exposure to historical data, supervised-learning neural networks attempt to capture desired relationships between the inputs and the outputs. Back-propagation is the most popular and widely used capture procedure. Back-propagation (Rumelhart et al. 1986, Werbos 1995) applies the gradient-descent technique to change a collection of weights so that some cost function can be minimized. The cost function, which is dependent on weights and training patterns only, is defined by:

$$C(W) = \tfrac{1}{2}\Sigma\ (T_{ij} - O_{ij}) \tag{1}$$

where the T is the target value, O is the output of the network, i represents the output nodes, and j represents the training patterns.

After the network propagates from the input layer to the output layer, the error between the desired output and actual output will be back-propagated to the previous layer. In the hidden layers, the error for each node is computed by the weighted sum of errors in the next layer's nodes. In a three-layered network, the next layer means the output layer. The activation function is usually a sigmoid function with the weights modified according to (2) or (3).

$$\Delta W_{ij} = \eta\ X_j\ (1 - X_j)(T_j - X_j)\ X_i \tag{2}$$

or

$$\Delta W_{ij} = \eta\ X_j\ (1 - X_j)\ (\Sigma\ \delta_k W_{jk})\ X_i \tag{3}$$

where W_{jk} is weight from node i to node (e.g., neuron) j, η is the learning rate, X_j is the output of node j, T_j is the target value of node j, and δ_k is the error function of node k. If j is in the output layer, (2) is used. If j is the hidden layers, (3) is used. The weights are updated to reduce the cost function at each step. The process continues until the error between the predicted and the actual outputs is smaller than some predetermined tolerance.

Rabelo (1990) was the first to use back-propagation neural nets to solve job shop scheduling problems. He allowed several job types, with different arrival patterns, process plans, precedence requirements, and batch sizes. Examples were generated to train the neural network to select those characterizations of the manufacturing environments suitable for various scheduling policies and applied to the target manufacturing system. The neural networks were trained for problems involving

3, 4, 5, 8, 10, and 20 machines. To carry out this training, a special input-feature space was developed. This space contained information on both job characteristics (such as job types, number of jobs in each type, routings, due dates, and processing times) and shop characteristics (such as number of machines and their capacities). Neural networks were tested on numerous scheduling problems with a variety of performance measures. For each test, the output of the neural network represented a relative ranking of the available dispatching rules. The one with the largest ranking was selected. Rabelo showed that the same rule did not always minimize a specified performance measure under all input conditions. For example, SPT does not always minimize mean flow time. In addition, he showed that the rule selected by the neural network never performed worse than the presumed optimum.

3.2.3. Multilayer Perceptrons

A multilayer perceptron is a fully connected feed-forward network consisting of an input layer, an output layer, and several hidden layers in between. Each layer is composed of nodes that are fully connected with those in the succeeding layer by weights. Each node computes a weighted sum of the elements in the preceding layer, subtracts a threshold, and then passes the result through a nonlinear function, called an activation function. Typically, the activation function is a sigmoid energy function and the learning algorithm employed to adjust weights is the Backpropagation algorithm (Rumelhart et al. 1986).

When an input pattern in training data is fed into the network, the error between the desired output and actual output values will be back-propagated to the previous layer and the weights will be adjusted accordingly. This procedure is called training and it is done to obtain the proper weight matrices so that the total is minimized. Yih et al. (1993) conducted a three-phased experiment to quantify the benefits of training. Schedules were generated by a human expert, an untrained neural network, and a neural network with training data refined by a semi-Markov decision model. The results indicated that the untrained neural network performed worse than the human expert did. However, the trained neural network outperformed both. This implies that good training data will significantly improve network performance.

Several works have used multilayer perceptrons with the Backpropagation training algorithm in scheduling or in candidate rules selection. Potvin et al. (1992) modified the network structure but still used the Backpropagation learning algorithm to build up the dispatcher for automated vehicles. Rabelo et al. (1993) used modular neural networks to serve as a candidate rule selector. In 1996, Chen and Yih discussed the impact of the network input attributes on the performance of the resulting control system.

As mentioned above, Cho and Wysk (1993) utilized the multilayer perceptron to take the place of the knowledge-based system in selecting candidate scheduling rules. In their proposed framework, the neural network will output a ''goodness'' index for each rule based on the system attributes and a performance measure. Sim et al. (1994) used an expert neural network for the job shop scheduling problem. In their approach, an expert system will activate one of 16 subnetworks based on whether the attribute corresponding to the node (scheduling rules, arrival rate factor, and criterion) is applicable to the job under consideration. Then the job with the smallest output value will be selected to process.

Yih and Jones (1992) proposed using multilayer perceptrons in selecting some candidate rules for further evaluation of their performance. In their proposed approach, a multilayer perceptron will take the attributes describing the system configuration and the performance measures and will output a proper matching score for each dispatching rule. They also used this approach for multiple-criterion objectives.

Sun and Yih (1996) adopted their idea to develop a neural network-based controller for manufacturing cells. In their approach, a neural network was trained to serve as decisionmaker that will select a proper dispatching rule for its associated machine to process the next job. Based on their results, the controller performs well under multiple criterion environments. In addition, when the production objectives change, the controller can respond to such change in a short time.

3.2.4. Unsupervised Neural Networks (Competition Based)

Competition-based neural networks, which are good at classifying or clustering input data, can also be applied to scheduling problems. Since the classes or clusters are not known in advance, the network must discover them by finding correlation in the input data. Multidimensional data sets are presented to the network, which adaptively adjusts its weights. This input presentation process is repeated until the network reaches stability—each output unit is activated only for a particular subset of the input patterns. Variations of these neural networks have been used to solve scheduling problems. For example, Bourret et al. (1989) applied some of these principles to develop a neural network that was able to schedule optimal time periods of low-level satellites to one or several antennas. The neural network was able to take into account that each satellite has a given priority and several other

operational constraints. Min et al. (1998) adopted this concept and developed a methodology in a multiobjective scheduling problem. Kim et al. (1998) integrated the network with inductive learning module to develop a real-time controller for flexible manufacturing systems.

3.2.5. Reinforcement Learning

We noted above that supervised learning neural networks attempt to capture desired relationships between inputs and outputs through exposure to training patterns. For some problems, the training period may be too short to find those relationships. When the desired response is obtained, changes to the neural network are performed by assessing penalties for the actions previously decided by the neural network. As summarized by Tesauro (1992), "In the simplest form of this paradigm, the learning system passively observes a temporal sequence of input states that eventually leads to a final reinforcement or reward signal (usually a scalar). The learning system's task in this case is to predict expected reward given an observation of an input state or sequence of input states. The system may also be set up so that it can generate control signals that influence the sequence of states." For scheduling, the learning task is to produce a schedule that minimizes (or maximizes) the performance measure. Several procedures have been developed to train neural networks in a variety of generic cases.

One of the popularly adopted reinforcement learning algorithms is called Q-learning. In this approach, an action-value function, which assigns an expected utility to take a given action in a given state, is defined. The output of this function is called Q-values. The relation between Q-values and the utility values is as follows:

$$U(s) = \operatorname*{Max}_{a} Q(a, s)$$

where $U(s)$ = the utility value at state s
$Q(a, s)$ = the Q-value of taking action a in state s

The learning process in Q-learning is to find an appropriate Q-value associated with each action in each state that the decision can be based on.

Rabelo et al. (1994) utilized a procedure developed by Watkins (1989), called Q-learning, to solve dynamic scheduling problems. The procedure followed trends in the shop floor and selected a dispatching rule that provided the maximum reward according to performance measures based on tardiness and flow time. Zhang and Dietterich (1996) utilized a procedure developed by Sutton (1988) called TD(λ) to schedule payload processing of NASA's space shuttle program. The scheduling system was able to outperform an iterative repair scheduler that combined heuristics with simulated annealing.

Kim and Lee (1995) formulated the machine-scheduling problem as a reinforcement learning problem and then developed a learning-based heuristic, called EVIS, for solving the scheduling problem. The EVIS, implementing reinforcement learning with the genetic algorithm, was then applied to a few deterministic scheduling problem instances. The results show that the proposed heuristic has good average-case performances for most of the problem instances.

Another alternative in reinforcement learning involves using the CMAC network. A CMAC network can be regarded as an associative memory system which stores the appropriate output in the associated memory cells. As mentioned by Miller et al. (1990), the CMAC network is an alternative to the back-propagated, multilayer, neural network because it has the advantages of local generalization, rapid training, and output superposition. Several researches have been involved in applying the CMAC network to develop a controller. Miller et al. (1990) demonstrated the application of CMAC networks in real-time robot control without providing any initial knowledge, in character recognition, and in signal processing. Lin and Kim (1991) constructed a CMAC-based controller and demonstrated its capability in the inverted pendulum problem. Moody (1989) proposed a multiresolution CMAC (MRC) that combines some CMAC networks with different resolution to increase the accuracy and generalization ability. The proposed approach shows good performance and online learning ability in prediction of a time series.

3.3. Genetic Algorithms

Genetic algorithms (GA) provide an optimization methodology based on a direct analogy to Darwinian natural selection and mutations in biological reproduction. In principle, genetic algorithms encode a parallel search through concept space, with each process attempting coarse-grain hill climbing (Goldberg 1988). Instances of a concept correspond to individuals of a species. Induced changes and recombinations of these concepts are tested against an evaluation function to see which ones will survive to the next generation. The use of genetic algorithms requires five components:

1. A way of encoding solutions to the problem—fixed length string of symbols
2. An evaluation function that returns a rating for each solution
3. A way of initializing the population of solutions
4. Operators that may be applied to parents when they reproduce to alter their genetic composition, such as crossover (i.e., exchanging a randomly selected segment between parents), mutation (i.e., gene modification), and other domain-specific operators
5. Parameter setting for the algorithm, the operators, and so forth

A number of approaches have been utilized in the application of genetic algorithms (GA) to job shop scheduling problems (Davis 1985; Goldberg and Lingle 1985; Starkweather et al. 1992):

1. Genetic algorithms with blind recombination operators have been utilized in job shop scheduling. Their emphasis on relative ordering schema, absolute ordering schema, cycles, and edges in the offsprings will lead to differences in such blind recombination operators.
2. Sequencing problems have been addressed by mapping their constraints to a Boolean satisfiability problem using partial payoff schemes. This scheme has produced good results for very simple problems.
3. Heuristic genetic algorithms have been applied to job shop scheduling. In these genetic schemes, problem specific heuristics are incorporated in the recombination operators (such as optimization operators based).

Starkweather et al. (1992, 1993) were the first to use genetic algorithms to solve a dual-criteria job shop scheduling problem in a real production facility, a beer plant. Those criteria were the minimization of average inventory in the plant and the minimization of the average waiting time for an order to be selected. These criteria are negatively correlated: as the inventory increases (decreases), the wait decreases (increases). To represent the production/shipping optimization problem, a symbolic coding was used for each member (chromosome) of the population. In this scheme, customer orders are represented by discrete integers. Therefore, each member of the population is a permutation of customer orders. The GA used to solve this problem was based on blind recombinant operators. This operator emphasizes information about the relative order of the elements in the permutation because this impacts both inventory and waiting time. A weighted sum of the two criteria was utilized to rank each member of the population. That ranking was based on an online simulation of the plant operations. This approach generated schedules that produced inventory levels and waiting times that were acceptable to the plant manager. In addition, the integration of the genetic algorithm with the online simulation made it possible to react to plant dynamics.

These applications have emphasized the utilization of genetic algorithms as a ''solo'' technique. This limits both the complexity of the problems solved and levels of success. Recent research has demonstrated the sensitivity of genetic algorithms to the initial population. When the initial population is generated randomly, genetic algorithms are shown to be less efficient that the annealing-type algorithms but better than the heuristic methods alone. However, if the initial population is generated by a heuristic, the genetic algorithms become as good as or better than the annealing-type algorithms. In addition, integration with other search procedures (e.g., taboo search) has enhanced the capabilities of both. This result is not surprising, as it is consistent with results from nonlinear optimization.

3.4. Fuzzy Logic

Fuzzy set theory has been utilized to develop hybrid-scheduling approaches. Fuzzy set theory can be useful in modeling and solving job shop scheduling problems with uncertain processing times, constraints, and set-up times. These uncertainties can be represented by fuzzy numbers, which are described by the concept called interval of confidence. These approaches usually are integrated with other methodologies (e.g., search procedures, constraint relaxation). For example, Slany (1994) stressed the imprecision of straightforward methods presented in the mathematical approaches and introduced a method known as fuzzy constraint relaxation, which is integrated with a knowledge-based scheduling system. Chang (1996) and Chang and Yih (1998, 1999) proposed a machine learning methodology to develop a fuzzy rule-based system for controlling a kanban system. Grabot and Geneste (1994) use fuzzy logic principles to combine dispatching rules for multi-criteria problems. Krucky (1994) used fuzzy logic to minimize setup times for a production line with a medium-to-high product mix by clustering assemblies into families of products that share the same setup. The clustering was achieved by balancing a product's placement time between multiple-high-speed placement process steps. Tsujimura et al. (1993) presented a hybrid system, which uses fuzzy set theory to model the processing times as triangular fuzzy numbers (TFNs). Each job is defined by two TFNs,

a lower bound and an upper bound. A branch and bound procedure is utilized to minimize makespan through the shop based on these estimates.

3.5. Commercial Systems

A number of university software systems use these techniques to do scheduling. A few commercial software systems use expert systems and genetic algorithms. Commercial hardware and software systems are available that implement neural networks, but none have been designed specifically for scheduling.

4. MANUFACTURING EXECUTION SYSTEMS (MES)

During the 1990s a new category of software system, manufacturing execution system (MES), emerged that consolidated and automated a number of functions involved in the management and operation of a production facility. An MES is a collection of hardware/software components that enables the management and optimization of production activities from order launch to finished goods. While maintaining current and accurate data, an MES guides, initiates, responds to, and reports on plant activities as they occur. An MES provides mission-critical information about production activities to decision support processes across the enterprise (MESA 1997; Wallace 1999). The term *order launch* is to be interpreted as initiation of physical production activities, typically beginning with materials preparation or machine preparation. Activities relating to planning and scheduling physical production operations are included within the scope of MES, but activities related to defining physical operations are not. The word ''component'' is used in a generic way to mean a separable portion of a larger whole.

Wallace (1999) provides a list of 12 major functions, derived from the original list in MESA (1997). These functions are similar to those found in the PERA model described in Section 3.

1. Resource allocation and tracking
2. Operations/detailed scheduling
3. Production unit dispatching
4. Specification management
5. Data collection/acquisition
6. Labor management
7. Quality management
8. Process management
9. Maintenance management
10. Product tracking
11. Performance analysis
12. Material management

The relationship between these functions and software products that call themselves MES is not clear. Some MES products can be purchased prepackaged as a single unit that performs all of these functions. Many other products provide only a subset of these functions; some call themselves MES, some do not. Standard interfaces would facilitate the integration of components into an overall MES and facilitate the integration of that MES with other enterprise software applications such as ERP (enterprise resource planning). No viable standards are emerging to fulfill this need, but there are four organizations looking at the problem: ISO, ISA, SEMI, and OMG.

Within the International Organization for Standardization (ISO), the Manufacturing Management Data Exchange (MANDATE) work in TC184/SC4/WG8 is focusing MES (http://www.iso.ch/meme/TC184SC4.html). Within ISA, MES-related work is being done in SP95 (http://www.isa.org/sc/committee/1,1512,145,00.html), see **2.1**. Within SEMI, MES-related standards are being generated as part of the standardization of CIM Framework specifications, (http://www.semi.org/web/wstandards.ns), see **2.2**. Within OMG, the MES-related work is being done by the Manufacturing Execution Systems/Machine Control group (http://www.omg.org/homepages/mfg/mfgmesmc.htm). Most of the remaining material in this section is based on work done in the OMG working group.

4.1. A More Detailed Look at MES Data

MES functions create, collect, modify, analyze, react to, and manage a great deal of data. These data are summarized below:

Dispatch data: job/operation dispatch list, or commands to operators and equipment

Equipment resource data: resource state, staffing, setup, current operations and assignments, and job usage history

Labor resource data: personnel availability and tracking information, and job assignment history

Maintenance data: machine availability data, maintenance history, and usage data

Material location data: location and state of materials with respect to active resources and material-handling components

Order data: units of a particular product to be manufactured, including status and associations with particular material groups

Performance, cost, and usage data: cost of operations performed, materials and resources used, idle time

Process control data: process control parameters

Product data (WIP): the amount, state, and disposition of materials in production and their relationship with manufacturing orders

Quality analysis data: data resulting from the quality analysis function, that is, interpreted measurements of process and product

Quality data: product and process measurement data, which can include such data taken from process operations

Resource description data: characteristics of labor and equipment resources such as capabilities, skills, types, and assigned cost

Schedule data: allocation of resources to jobs and processes per time period

Shop-floor data: raw data collected by data collection systems that can be used to derive product (WIP) data, resource data, performance data, and so on

Specification data: specifications for how to perform a manufacturing process, including sequence of operations to be performed, equipment, tooling and skills requirements, and materials to be used

Tooling resource data: usage, location and allocation information, which may be used for tracking, scheduling, and maintenance of tools

Table 2 shows the nature of the data usage by the MES functions. The numbers in each cell correspond to the numbers assigned above. The designations HRM, ERP, and PPE refer to non-MES systems of the enterprise that use some of the same data. HRM refers to human resource management systems, ERP refers to enterprise resource planning systems, and PPE refers to product and process engineering systems.

4.2. MES Object Models

An analysis of the MES functions and the data in Table 2 led to the development of the two MES object models shown in Figure 6 and Figure 7. These models, which are based on the work in Ray and Wallace (1995) and OMG (1997b), organize the functions and data in a way that facilitates the implementation of MES components based on object or component middleware technology. They also provide an implementor or integrator with an organization of a distributed MES architecture that simplifies integration with other manufacturing information systems.

The partitions that we have shown with dotted boxes in Figure 6 are groupings of model entities that are closely coupled or functionally similar. Coupling between these partitions should be supported via domain specific names or keys. Each partition in the model is described in detail below. While all these partitions are important to MES, they need not all be supported directly within an MES. We make special note below of those that we consider to be core MES partitions. The tags, shown in italics above or below each partition box in the figures, are general characterizations of the resource entities within the corresponding partition. These are further elaborated in the text below:

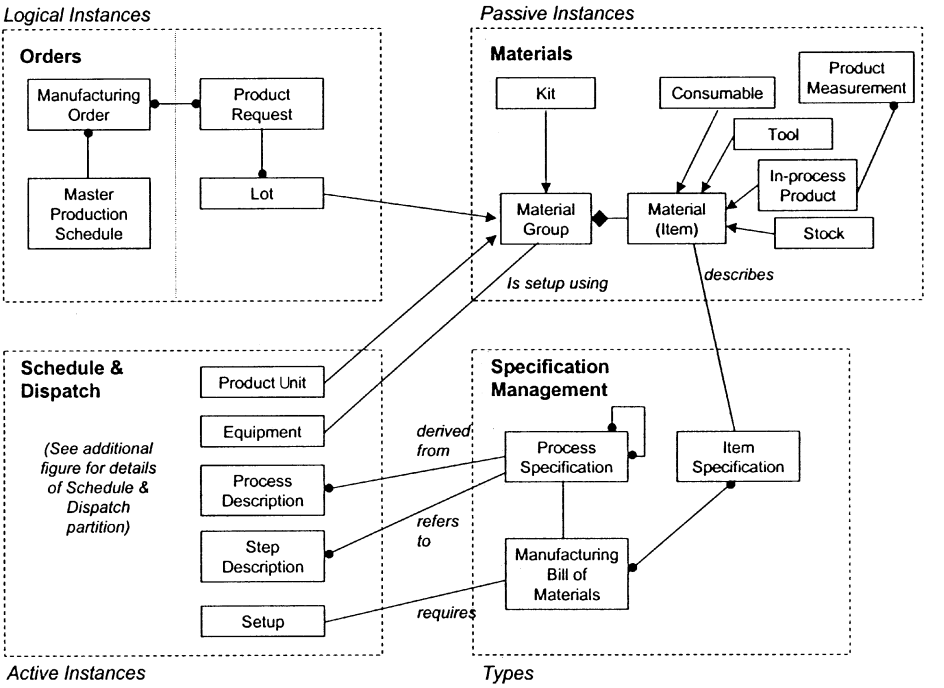
Orders: The orders partition contains two planning entity classes: those that are created by ERP systems and those that may be MRPII entities. This distinction is made here since it is not clear whether MRPII is considered an ERP or MES function. We have tagged this partition ''logical instances'' in the diagram to indicate that the entities within it are logical resources, which merely represent groups of physical resources or planned physical resources that are modeled in other partitions.

Schedule and dispatch: This partition is at the heart of MES operations, containing the primary entities involved in scheduling and dispatching. If the MES supports reactive scheduling, then the internal components must be able to share a common understanding of process description. Therefore, the model has a separate entity for process description, which is distinct from (and an instance of which is derived from a) process specification. We note with the tag ''active instances'' that unlike the entities in the other partitions, many Schedule and Dispatch entities can initiate actions.

Specification management: The entities in this partition have similar access characteristics but only a loose coupling. This means that some efficiency may be gained by putting all these entities into one component. Nevertheless, as long as all access requirements shared by these entities are met, these entities could be stored in multiple components. We note with the tag ''types'' that this partition contains information resource entities that provide type information used to instantiate or describe other entities.

TABLE 2 Relationship of Data to MES Activities

	Collects	Creates	Changes	Manages	Analyzes	Reacts To	Delivers	Displays	Derived From
Dispatch data		3	3, 8			8, 10, 12 Control	3	3, 10	
Equipment resource data	5		5, 9	1	9	2, 3			
Labor resource data	6		10	6, HRM	11	2, 3			
Maintenance data	5	9	9	9	9, 10	2, 3, 9	9		
Material location data	5, 12	12	12	12		3, 10		12	Shop-floor data
Order data	ERP		10	ERP		2, 3, 10	ERP	2, 10	Product data
Performance data		11		11		ERP, P/PE	11	11	Resource data, Schedule data, Shop-floor data, Process data, Product data
Process control data		P/PE	8			Control	8		
Product data (WIP)	5		7, Control	10	11	2, 10, ERP		10	Shop-floor data, Process data
Quality analysis data		7		7, 10		3, 9	7	7	Quality data
Quality data	5?, 7	Control		7	7, 10	?			Product data, Process data
Resource description data		ERP, HRM, P/PE		6, HRM, ERP, P/PE		2, 3			
Schedule data		2	2	2	11	2, 3			
Shop-floor data	5			5	2, 10	2, 3	5		
Specification data	P/PE	P/PE	4	4		3, 7	P/PE		
Tooling resource data	12	12	Control	1, 10		2, 3, 10			



*The graphic notation used in this figure and subsequent object models, is a simplified version of the object modeling technique (OMT) notation described in Rumbaugh (1991).

Figure 6 Simplified Object Model.*

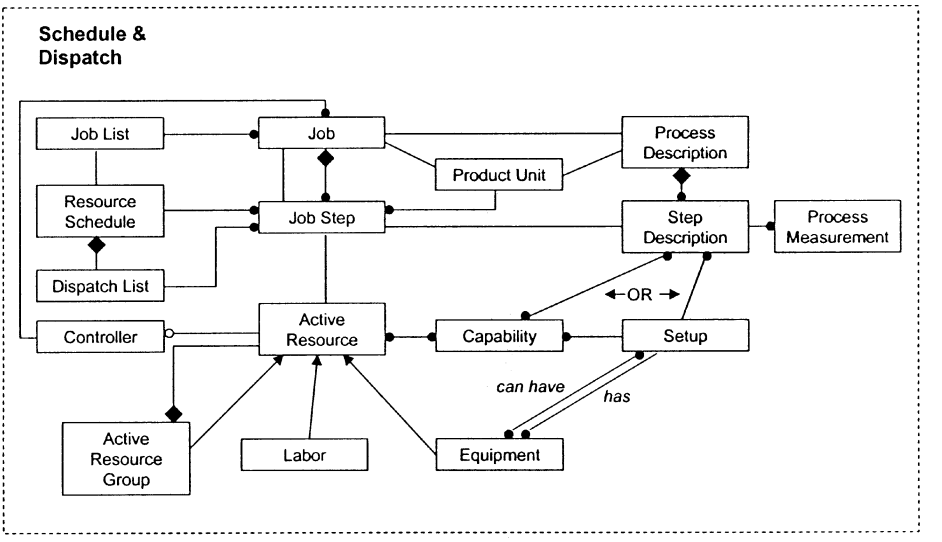


Figure 7 Detailed View of Scheduling and Dispatching.

Materials: The entities in this partition are related to material instances such as material status, history, and genealogy data. It may also provide inventory information. We note with the tag ''passive instances'' that the entities in this partition represent physical resources that are manipulated or acted upon by active resources represented in other models.

The entities of the MES Object Model shown in Figures 6 and 7 are described in detail below:

Active resource: a physical resource, or a logical grouping a resources, that can perform process operations and supports specified capabilities during manufacturing activities—people and equipment such as machines, robots, storage devices, transport devices, etc.

Capability: the ability of a resource to perform a specified operation. For human resources, a capability is sometimes called a (certified) skill. For equipment resources, this potential may be dependent on setup and on having an appropriately skilled operator.

Consumable: a unit of material that is used and expended during manufacturing or support processes.

Controller: a software system, or human agent, that provides the intelligence needed by an active resource to perform all or part of a process operation automatically.

Item specification: the complete description of a kind of material item including its shape, weight, handling characteristics, image, etc.

Job: a unit of work assigned to an active resource. At the schedule/dispatch level, the unit of work is a lot; at lower levels, the unit of work is a process operation, transport operation, or operation step.

Job step: an instantiation of a step description. Depending on its granularity, a job step becomes a job for the active resource to which the step execution is assigned.

Kit: a physical collection of material items that is handled as a unit.

Labor: a human active resource (employee or contractor or group) who performs certain jobs directly or in collaboration with a machine. The ability of a human resource to perform a specific kind of process with specific kinds of equipment is called a certified skill.

Lot: the unit of product into which a manufacturing order is decomposed for scheduling, dispatching, and tracking. A lot is a conceptual material group. The actual corresponding product units on the factory floor may be decomposed and regrouped, depending on production policies and needs.

Manufacturing bill of materials: a list of the types and quantities of all materials needed to manufacture a unit of product or to perform a particular process.

Manufacturing order: a quantity of a particular product to manufacture as specified by an ERP system.

Master production schedule: A long-term schedule created and maintained by enterprise planning systems that defines quantities of particular products to be produced in particular time frames based on customer demands, manufacturing capacity, and resource availability.

Material item: a physical resource that is acted upon or used by an active resource in the performance of a manufacturing activity. It is characteristic of most material items that they can be moved around the factory floor, and it is often the case that their location is tracked in some way.

Material group: a logical or physical collection of material instances.

Process description: a breakdown of a process into subtasks or steps, expressing the requirements for each step and its relationship to other steps in the recipe. Every process description has an associated active resource that is responsible for planning and executing the process described. Process descriptions come in many forms: NC programs, routings, operation sheets, recipes, etc.

Process specification: archival form of a process description, usually a document. An information resource is targeted for a particular type of active resource. It is copied as needed to create process descriptions for individual resources.

Product in-process: a material item, which becomes part of a final product.

Product request: a unit of product that is the basis for planning/scheduling activities at the highest level of a manufacturing execution system.

Product unit: a quantity of product (or a group of in-process product items) that undergoes manufacturing activities together.

Resource schedule: a collection of assignments of active resources to job steps at specific times.

Setup: a particular configuration of an equipment resource that results in a set of active capabilities for that resource.

Step description: a part of a process description that identifies a job step to be performed, resources necessary to perform the job step, other information needed to plan the job step, and, usually, the process specification that the active resource will use to perform the job step activity.

Stock Material: a kind of material that is the starting point for, or an ingredient in, a unit of product.

Tool: a material item used by an active resource in the performance of some manufacturing activity. A tool is a material item that is needed during the manufacturing process but is not (usually) consumed and does not become part of the finished goods. Tools are used to set up an equipment resource, or augment a workstation setup, in order to enable the workstation to perform a particular process operation.

4.3. Market Trends and Future Directions

Originally predicted to be a strong new market area, the MES product category had only grown about $218 million in 1998 (Callaway 1998). This has been attributed to such diverse factors as manufacturing management's preoccupation with higher-level enterprise systems such as enterprise resource planning (ERP) and supply chain management (SCM) and resources being diverted for Year 2000 projects. No matter what the cause, MES has not become a ubiquitous system across all manufacturing domains.

However, the functionality it targeted remains important as the changing marketplace for manufactured goods forces manufacturers to respond more quickly to market opportunities and participate in more dynamic supply chains. An MES provides the higher level of integration needed to support rapid access to data, as well as the means to respond rapidly to new production tasks or priorities. Because this need still remains, vendors in other product categories have expanded their product line to include many of the MES functions described in Section 2.2, with mixed success. This encroachment has come from above with ERP systems and from below with open control systems. While it is certain that the functionality and data described in this section will be available in the factory of the future, it is difficult to predict what systems will be managing it. This underscores the importance of developing standard interfaces and data models for this area so that manufacturers can make use of this technology no matter what the products are called or where the lines are drawn between them.

5. SUMMARY

This chapter has addressed three subjects related to the control of shop-floor operations: control architectures, AI-based scheduling, and manufacturing execution systems (MES). The ideas discussed in this chapter are applicable to many different types of manufacturing and service industries. To date, very few of the advanced scheduling techniques have been incorporated into commercial MES software packages, and little or no effort has been put into integrating these packages into an open architecture for shop-floor control. We expect this situation to change dramatically over the next few years as companies push more toward Internet-based electronic commerce.

REFERENCES

Adachi, T., Moodie, C. L., and Talavage, J. J. (1988), "A Pattern-Recognition-Based Method for Controlling a Multi-loop Production System," *International Journal of Production Research,* Vol. 26, No. 12, pp. 1943–1957.

Adachi, T., Talavage, J. J., and Moodie, C. L. (1989), "A Rule-Based Control Method for a Multi-loop Production System," *Artificial Intelligence in Engineering,* Vol. 4 , No. 3, pp. 115–125.

ANSI/ISA-S95.00.01-2000 (2000), *Enterprise-Control System Integration Part 1: Models and Terminology,* International Standards Assocation, Research Triangle Park, NC.

Arizono, I., Yamamoto, A., and Ohta, H. (1992), "Scheduling for Minimizing Total Actual Flow Time by Neural Networks," *International Journal of Production Research,* Vol. 30, No. 3, pp. 503–511.

Bourret, P., Goodall, S., and Samuelides, M. (1989), "Optimal Scheduling by Competitive Activation: Application to the Satellite-Antennae Scheduling Problem," in *Proceedings of the International Joint Conference on Neural Networks*.

Callaway (1998), "A Second Chance for MES," in *Managing Automation,* Vol. 13, No. 12, pp. 34–43.

Chandra, J., and Talavage, J. (1991), "Intelligent Dispatching for Flexible Manufacturing," *International Journal of Production Research,* Vol. 29, No. 11, pp. 2259–2278.

Chang, T. (1996), ''A Fuzzy Rule-Based Methodology for Dynamic Kanban Control in a Generic Kanban System,'' Ph.D. dissertation, Purdue University, West Lafayette, IN.

Chang, T., and Yih, Y. (1998), ''A Fuzzy Rule-Based Approach for Dynamic control of kanbans in a generic kanban system,'' *International Journal of Production Research,* Vol. 36, No. 8, pp. 2247–2257.

Chang, T., and Yih, Y. (1999), ''Constructing a Fuzzy Rule System from Examples,'' *Journal of Integrated Computer-Aided Engineering,* Vol. 6, pp. 213–221.

Chen, C., Yih, Y., and Wu, Y. (1999), ''Auto-bias Selection for Developing Learning-Based Scheduling Systems,'' *International Journal of Production Research,* Vol. 37, No. 9, pp. 1987–2002.

Chiu, C., and Yih, Y. (1995), ''Learning-Based Methodology for Dynamic Scheduling in Distributed Manufacturing Systems,'' *International Journal of Production Research,* Vol. 33, No. 11, pp. 3217–3232.

Cho, H., and Wysk, R. A. (1993), ''A Robust Adaptive Scheduler for an Intelligent Workstation Controller,'' *International Journal of Production Research,* Vol. 31, No.4, pp. 771–789.

Daouas, T., Ghedira, K., and Muller, J. (1995), ''Distributed Flow Shop Scheduling Problem versus Local Optimization,'' in *Proceedings of the First International Conference on Multi-Agent Systems,* MIT Press, Cambridge, MA.

Davis, L. (1985), ''Job Shop Scheduling with Genetic Algorithms,'' in *Proceedings of an International Conference on Genetic Algorithms and Their Applications* (Carnegie Mellon University), pp. 136–140.

Doscher, D., Ed. (1998), *Computer Integrated Manufacturing* (*CIM*) *Framework Specification Version 2.0*, Technology Transfer #93061697J-ENG, SEMATECH, Austin, Tx.

Farhoodi, F. (1990), ''A Knowledge-Based Approach to Dynamic Job-Shop scheduling,'' *International Journal of Computer Integrated Manufacturing,* Vol. 3, No. 2, pp. 84–95.

Foo, Y. S., and Takefuji, Y. (1988a), ''Stochastic Neural Networks for Solving Job-Shop Scheduling. Part 1: problem representation,'' in *IEEE International Joint Conference on Neural Networks,* Vol. 2, pp. 275–282.

Foo, Y. S., and Takefuji, Y. (1988b), ''Stochastic Neural Networks for Solving Job-Shop Scheduling. Part 2: Architecture and Simulation,'' in *IEEE International Joint Conference on Neural Networks,* Vol. 2, pp. 283–290.

Fox, M. (1983), ''Constraint-Directed Search: A Case Study of Job Shop Scheduling,'' Ph.D. dissertation, Carnegie Mellon University.

Goldberg, D. (1988), *Genetic Algorithms in Search Optimization and Machine Learning,* Addison-Wesley, Menlo Park, CA.

Goldberg, D., and Lingle, R. (1985), ''Alleles, Loci, and the Traveling Salesman Problem,'' in *Proceedings of the of the International Conference on Genetic Algorithms and Their Applications* (Carnegie Mellon University), pp. 162–164.

Grabot, B., and Geneste, L. (1994), ''Dispatching Rules in Scheduling: A Fuzzy Approach,'' *International Journal of Production Research,* Vol. 32, No. 4, pp. 903–915.

Gupta, M. C., Judt, C., Gupta, Y. P., and Balakrishnan, S. (1989), ''Expert Scheduling System for a Prototype Flexible Manufacturing Cell: A Framework,'' *Computers and Operations Research,* Vol. 16, No. 4, pp. 363–378.

Harel, D. (1987), ''Statecharts: A Visual Formalism for Complex Systems,'' *Science of Computer Programming* Vol. 8, pp. 231–274.

Hawker, J. (1999), ''CIM Framework and Application Models,'' in *Information Infrastructure Systems for Manufacturing,* J. Mills, and F. Kimura, Eds., Kluwer, Boston.

Hodges, R. (2000), private communication.

Huang, H., and Chang, P. (1992), ''Specification, Modeling and Control of a Flexible Manufacturing Cell,'' *International Journal of Production Research,* Vol. 30, No. 11, pp. 2515–2543.

Jones, A., Ed. (1990), *Proceedings of COMCON'90,* NIST Special Publication, National Institute of Standards and Technology, Gaithersburg, MD.

Kim, C., Min, H., and Yih, Y. (1998), ''Integration of Inductive Learning and Neural network for Multi-objective FMS Scheduling,'' *International Journal of Production Research,* Vol. 36, No. 9, pp. 2497–2509.

Kim, G. H., and Lee, C. S. G. (1995), ''Genetic Reinforcement Learning Approach to the Machine Scheduling Problem,'' in *Proceedings of IEEE International Conference on Robotics and Automation,* Vol. 1, pp. 196–201.

Krucky, J. (1994), ''Fuzzy Family Setup Assignment and Machine Balancing,'' *Hewlett-Packard Journal,* June, pp. 51–64.

Lin, C., and Kim, H. (1991), "CMAC-Based Adaptive Critic Self-Learning Control," *IEEE Transactions on Neural Networks,* Vol. 2, No. 5, pp. 530–533.

MESA International (1997), "MES Explained: A High Level Vision," White paper #6, MESA International, Pittsburgh.

Miller, W. T., Glanz, F. H., and Kraft, L. G., III (1990), "CMAC: An Associative Neural Network Alternative to Backpropagation," *Proceedings of the IEEE,* Vol. 78, No. 10, pp. 1561–1567.

Min, H., Yih, Y. and Kim, C. (1998), "A Competitive Neural Network Approach to Multi-objective FMS Scheduling," *International Journal of Production Research,* Vol. 36, No. 7, pp. 1749–1765.

Moody, J. (1989), "Fast Learning in Multi-Resolution hierarchies," *Advances in Neural Information Processing,* Vol. 1, pp. 29–39.

Object Management Group (OMG) (1995), *Object Management Architecture Guide, Revision 3.0*, OMG, Framingham, MA.

Object Management Group (OMG) (1996), *CORBAservices: Common Object Services Specification,* OMG, Framingham, MA.

Object Management Group (OMG) (1997a), *Common Object Request Broker: Architecture and Specification,* Revision 2.1, OMG, Framingham, MA.

Object Management Group (OMG) (1997b), "Manufacturing Domain Task Force RFI-3 Manufacturing Execution Systems (MES)," Document mfg/97-11-01, OMG, Framingham, MA.

O'Grady, P., and Lee, K. H. (1988), "An Intelligent Cell Control System for Automated Manufacturing," *International Journal of Production Research,* Vol. 26, No. 5, pp. 845–861.

O'Grady, P., and Seshadri, R. (1992), "Operation of X-Cell—an Intelligent Cell Control System," *Computer Integrated Manufacturing Systems,* Vol. 5, No. 1, pp. 21–30.

Parunak, H., Irish, B., Kindrick, J., and Lozo, P. (1985), "Fractal Actors for Distributed Manufacturing Control," in *Proceedings of the Second IEEE Conference on Artificial Intelligence Applications,* pp. 653–660.

Pluym, B. V. D. (1990), "Knowledge-Based Decision-Making for Job Shop Scheduling," *International Journal of Computer Integrated Manufacturing,* Vol. 3, No. 6, pp. 354–363.

Potvin, J., Shen, Y., and Rousseau, J. (1992), "Neural Networks for Automated Vehicle Dispatching," *Computers and Operations Research,* Vol. 19, Nos. 3/4, pp. 267–276.

Quinlan, J. (1986), "Induction of Decision Trees," *Machine Learning,* Vol. 1, pp. 81–106.

Rabelo, L. (1990), "Hybrid Artificial Neural Networks and Knowledge-Based Expert Systems Approach to Flexible Manufacturing System Scheduling," PhD. dissertation, University of Missouri-Rolla.

Rabelo, L., Yih, Y., Jones, A., and Tsai, J. (1993), "Intelligent Scheduling for Flexible Manufacturing Systems," in *Proceedings of IEEE International Conference on Robotics and Automation,* Vol. 3, pp. 810–815.

Rabelo, L., Sahinoglu, M., and Avula, X. (1994), "Flexible Manufacturing Systems Scheduling Using Q-Learning," in *Proceedings of the World Congress on Neural Networks* (San Diego), pp. I378–I385.

Ray, S., and Wallace, S. (1995), "A Production Management Information Model for Discrete Manufacturing," in *Production Planning and Control,* Vol. 6, No. 1, pp. 65–79.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W. et al. (1991), *Object-Oriented Modeling and Design,* Prentice-Hall, Englewood Cliffs, NJ.

Rumelhart, D., McClelland, J., and the PDP Research Group (1986), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1: Foundations,* MIT Press, Cambridge, MA.

Shaw, M., Park, S. and Raman, N. (1992), "Intelligent Scheduling with Machine Learning Capabilities: The Induction of Scheduling Knowledge," *IEE Transactions on Design and Manufacturing,* Vol. 24, pp. 156–168.

Sim, S. K., Yeo, K. T., and Lee, W. H. (1994), "An Expert Neural Network System for Dynamic Job Shop Scheduling," *International Journal of Production Research,* Vol. 34 , No. 8, pp. 2353–2373.

Slany, W. (1994), "Scheduling as a Fuzzy Multiple Criteria Optimization Problem." CD-Technical Report 94/62, Technical University of Vienna.

Starkweather, T., Whitley, D., Mathias, K., and McDaniel, S. (1992), "Sequence Scheduling with Genetic Algorithms," in *Proceedings of the US/German Conference on New Directions for OR in Manufacturing,* pp. 130–148.

Starkweather, T., Whitley, D., and Cookson, B. (1993), "A Genetic Algorithm for scheduling with Resource Consumption," in *Proceedings of the Joint German/US Conference on Operations Research in Production Planning and Control,* pp. 567–583.

Sun, Y. and Yih, Y. (1996), "An Intelligent Controller for Manufacturing Cells," *International Journal of Production Research,* Vol. 34, No. 8, pp. 2353–2373.

Sutton, R. (1988), "Learning to Predict by the Methods of Temporal Differences," *Machine Learning,* Vol. 3, pp. 9–44.

Talavage, J. J., and Shodhan, R. (1992), "Automated Development of Design and Control Strategy for FMS," *International Journal of Computer Integrated Manufacturing,* Vol. 5, No. 6, pp. 335–348.

Tesauro, G. (1992), "Practical Issues in Temporal Difference Learning," *Machine Learning,* Vol. 8, pp. 257–277.

Tsujimura, Y., Park, S., Chang, S., and Gen, M. (1993), "An Effective Method for Solving Flow Shop Scheduling Problems with Fuzzy Processing Times," *Computers and Industrial Engineering,* Vol. 25, pp. 239–242.

Wallace E., Ed. (1999), "NIST Response to MES Request for Information," Internal Report 6397, National Institute of Standards and Technology, Gaithersburg, MD.

Watkins, C. (1989), "Learning from Delayed Rewards," Ph.D. dissertation, King's College, Cambridge.

Werbos, P. (1995), "Neurocontrol and Supervised Learning: An Overview and Evaluation," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches,* Van Nostrand Reinhold, New York, pp. 65–89.

Williams, T. (1992), *The Purdue Enterprise Reference Architecture—A Technical Guide for CIM Planning and Implementation,* Instrument Society of America, Research Triangle Park, NC.

Wu, S. D., and Wysk, R. A. (1988), "Multi-pass Expert Control System—a Control Scheduling Structure for Flexible Manufacturing Cells," *Journal of Manufacturing Systems,* Vol. 7, No. 2, pp. 107–120.

Wu, S. D., and Wysk, R. A. (1989), "An Application of Discrete-Event Simulation to On-line Control and Scheduling in Flexible Manufacturing," *International Journal of Production Research,* Vol. 27, No. 9, pp. 1603–1623.

Yih, Y. (1988), "Trace-Driven Knowledge Acquisition (TDKA) for Rule-Based Real-Time Scheduling Systems," *Journal of Intelligent Manufacturing,* Vol. 1, No. 4, pp. 217–230.

Yih, Y. (1994), "An Algorithm for Hoist Scheduling Problems," *International Journal of Production Research,* Vol. 32, No. 3, pp. 501–516.

Yih, Y., and Jones, A. T. (1992), "Candidate Rule Selection to Develop Intelligent Scheduling Aids for Flexible Manufacturing Systems (FMS)," in *Proceedings of a Joint German/US Conference,* Hagen, Germany, pp. 201–217.

Yih, Y., Liang, T., and Moskowitz, H. (1993), "Robot Scheduling in a Circuit Board Production Line: A hybrid OR/ANN Approach," *IIE Transactions,* Vol. 25, No. 2, pp. 26–33.

Zhang, C., Yan, P., and Chang, T. (1991), "Solving Job-Shop Scheduling Problem with Priority Using Neural Network," in *IEEE International Joint Conference on Neural Networks,* pp. 1361–1366.

Zhang, M., and Zhang, C. (1995), "The Consensus of Uncertainties in Distributed Expert Systems," in *Proceedings of the First International Conference on Multi-Agent Systems,* MIT Press, Cambridge, MA.

Zhang, W., and Dietterich, T. (1996), "High-Performance Job-Shop Scheduling with a Time-delay TD(() network," *Advances in Neural Information Processing Systems,* MIT Press, Cambridge, MA, pp. 1025–1030.

Zhou, D. N., Cherkassky, V., Baldwin, T. R., and Olson, D. E. (1991), "A Neural Network Approach to Job-Shop Scheduling," *IEEE Transactions on Neural Networks,* Vol. 2, No. 1, pp. 175–179.