Journal of Manufacturing Systems
Vol. 15/No. 2
1996

A DES-Theory-Based Hybrid Supervisory Control System for Manufacturing Systems

R.A. Williams, B. Benhabib, and **K.C. Smith,** University of Toronto, Toronto, Ontario, Canada

Abstract

Manufacturing systems generally encompass processes that are discrete in time and space. Among the variety of real-time supervisory control techniques reported in the literature, controlled-automata-based discrete event system (DES) theory is one of the few with mathematical formalism. However, the control of even a moderately complex system using this theory may require a very large control strategy.

To attempt to cope with this problem, a hybrid supervisory controller was developed that splits operations between a DES-theory-based supervisory controller and an alternate mechanism. The alternate mechanism reroutes part production in real time whenever a priori unmodeled events occur. This modified approach provides a significantly more efficient controller than could be attained using solely a DES-theory-based supervisory controller.

Keywords: *Flexible Manufacturing Workcells, Automation, Supervisory Control, Rerouting of Production, Discrete Event System (DES) Theory*

1–Introduction

1.1–Background and Motivation

Parts produced in quantities of fewer than 10,000 units annually account for greater than 50% of the expenditures on manufactured parts produced in job shops of small and medium-sized firms.[1] The introduction of flexible manufacturing systems (FMSs) in the past decade resulted in improvements in the productivity of medium and large-volume parts production. FMSs were not, however, a great improvement for small production volumes. On the other hand, the mini-plant concept, originally developed in Norway, did improve the productivity of small-sized firms.[2] This concept proposed the grouping of machines and supporting resources into workcells.

An (automated) workcell generally "consists of a group of devices, such as robots, numerical control machines, sensors, and so on, under the control of a centralized supervisor capable of performing a specific set of manufacturing functions."[3] The design of a supervisory controller entails the formulation of control laws and the synthesis of supervisors. The laws specify how the supervisor is to react to the behavior of the manufacturing system, the goal being to have some production specifications satisfied within the standing control enforcement constraints. Petri nets,[4,5] real-time temporal logic,[6,7] knowledge engineering,[3,8] timed-transition models, and controlled automata[9] have all been used for this purpose. This paper will focus on the application and development of a DES-theory-based supervisor that utilizes controlled automata concepts.

DES-theory-based controllers have the desirable feature that they may be proved and verified correct before implementation. However, the control of even moderately complex systems can easily require an immensely large DES strategy.* A hybrid approach that uses some alternate mechanism in addition to a DES supervisory controller would relieve the controller of the need for so many states by (1) taking on the responsibility for some of the control objectives and (2) asserting control whenever events diverge from the significantly reduced number of states of the DES supervisory controller. This paper describes such a hybrid supervisory controller that was developed.

Only limited research has been carried out on the application of DES theory to the control of manufacturing environments.[9,11,12] The only research to date in which DES theory is reported to form part of a hybrid control mechanism is Balemi's furnace controller.[13] His DES-type supervisory control has two parts: a supervisor and a controller. The supervisor ensures that safety constraints are enforced. The controller steers the system toward the desired goal, which is to accomplish a sequence of tasks. In this system, a command chosen by the controller will always be compatible with the safety constraints

* Ho[10] notes that when solving basic control synthesis problems, although they have been shown to be of polynomial complexity in the number of states, the number of states in a practical system can be exponential in the number of constituent processes.

and, therefore, will be accepted by the supervisor. This arrangement allows the controller to be bypassed, via manual override, while maintaining safety constraints. To limit the size and complexity of his hybrid supervisory controller, Balemi uses an incomplete controller that does not accept all possible responses.

1.2–A New Approach

The hybrid supervisory controller (HSC) proposed in this paper consists of three main elements (*Figure 1*):

1. DES supervisor, which contains the nominal supervisory control strategy,
2. Diagnostic system, and
3. Alternate-strategy driver (ASD), which generates alternate part routes when needed.

The diagnostic system interprets sensory data. It feeds its interpretation to the DES supervisor and to the ASD. If the ASD determines that new (alternate) part routes are needed, it derives them and submits them to the DES supervisor. The DES supervisor reacts to the information received from the diagnostic system and accepts the alternate routes from the ASD. It proceeds to synthesize control commands based on the safety constraints and the nominal part routes incorporated into itself. It submits the commands to the equipment of the workcell.

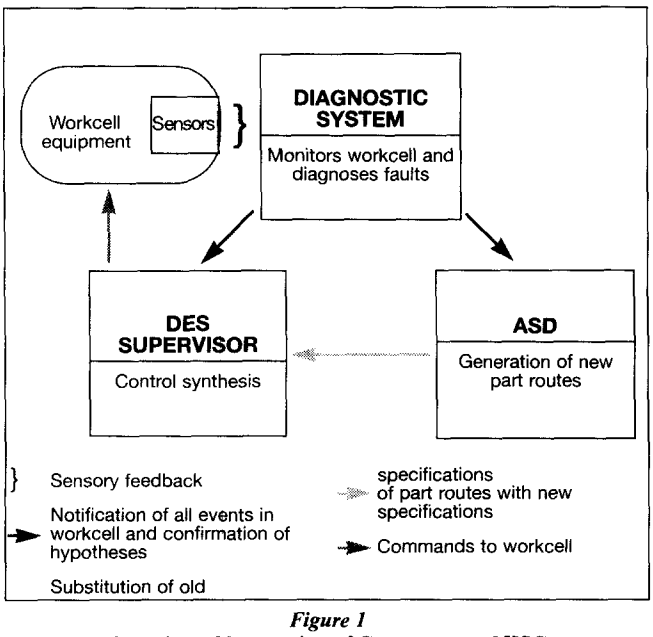

Figure 1
Overview of Interaction of Components of HSC

Despite having been formulated independently, the distribution of tasks between the ASD and DES supervisor in this work bears a resemblance to that of Balemi's control system. The significant difference between this work and Balemi's is that Balemi's controller is implemented within the context of DES theory, whereas the ASD proposed in this paper operates by heuristic means.

It is possible for Balemi's controller to be implemented within the context of DES theory because the environment it controls is much simpler than typical manufacturing workcells. There is only one order (route) in which to perform operations, and one piece of equipment on which to perform them. In the case of workcells, however, there would be many possible orders of operations and choices of equipment on which to perform them.

2–DES Fundamentals*

2.1–Representation of Controlled DES

A discrete event system (a plant) model comprises a finite set of states with transitions between them. The DES-based supervisor exerts control on the plant by disabling certain events that the system can generate or accept.

Two disjointed sets of events have to be considered during the generation of a DES-theory-based control strategy, namely controllable and uncontrollable events. Controllable events are preventable, that is, they may be disabled. The remote start of a machine is an example of a controllable event in the context of a manufacturing workcell. Uncontrollable events, on the other hand, are not preventable, that is, they cannot be disabled. Rather, they can only be observed by the supervisor. A machine breakdown is an example of an uncontrollable event.

During the control of a DES, controllable events are normally disabled, while uncontrollable events are assumed to always be enabled and cannot be disabled. They can only be avoided by preventing the occurrence of controllable events that lead to the undesired uncontrollable events. The supervisor enables controllable events according to logic specifications depending on the current state. This process is called *control synthesis.*

* This paper attempts to keep reference to DES theory to a minimum. Readers are referred to Ramadge and Wonham[14] for more in-depth information about DES theory.

2.2–Generating the Supervisor

A supervisory control strategy is constructed as follows:[9]

1. The workcell to be supervised is modeled.
2. The plant behavioral specifications are developed. Each specification defines a set of admissible event paths.
3. The supervisor is constructed based on the plant model and behavioral specifications.
4. Logical conflicts, if present, are removed to prevent blocking.

3–Description of DES Supervisor

In this section, the DES supervisor of the proposed HSC is discussed. Descriptions of the other prime modules, namely the diagnostic and the ASD modules, are presented in Sections 4 and 5, respectively.

3.1–Generating a DES Supervisor

The DES supervisor for a workcell is synthesized from (1) a model for the workcell, (2) specifications for the routes of the parts, referred to as linear part-routing specifications, and (3) specifications that enforce other constraints, referred to as safety specifications.

In the following subsections, the generators and specifications of the DES supervisor are presented as a preamble to the discussion of the new DES supervisor design in the context of the proposed HSC.

Plant Models

The DES supervisor requires control models of the equipment in the workcell, constructed from a set of individual equipment models—plants. Each plant is a generator for a set of events. Significant features of the equipment related to supervisor control are captured in the plant definitions in the form of state-transition diagrams. Two types of plants are defined in this paper: transport plants (such as robots and conveyors) and stationary plants (for example, machining centers and part buffers).

There are four types of events that can be generated by the plants, as follows:

α: Start of an operation (controllable)

β: Completion of an operation (uncontrollable)

λ: Failure of a plant (uncontrollable)

μ: Repair of a plant (controllable)

Due to the complexity of the supervisor being developed, other pertinent information had to be attached to event labels in the form of a five-digit alphanumeric code (*Figure 2*).

The generic specifications for a stationary plant and a transport plant are shown in *Figures 3a* and *3b*, respectively. For example, $\alpha_{gh\text{-}j\text{-}}$ is interpreted as: an α event affecting part number h of part type g currently on transport plant j performing an unspecified task (-), and that an unspecified stationary plant (-) would be affected by that event. An (X) symbol in the third digit would imply that no stationary plant may be affected by this event.

Safety Specifications

The goals of the safety specifications are to maintain safe operation of the plants in the workcell and to restrict certain operations that would cause parts to become deadlocked:

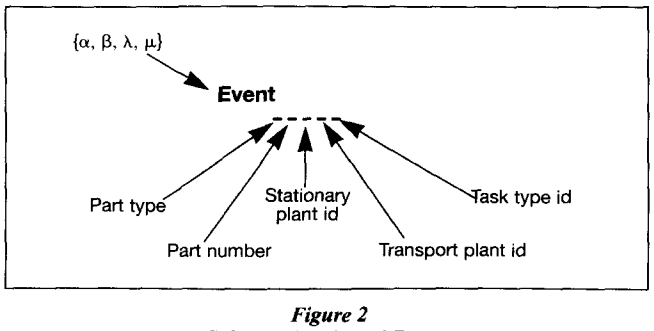

Figure 2
Subclassification of Events

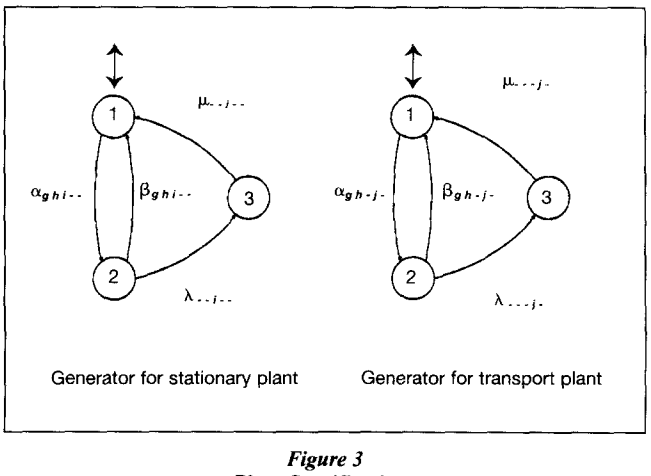

Figure 3
Plant Specifications

1. *Plant-activity specifications:* to restrict activity of each plant to operating on only one part at a time,
2. *Plant-buffer specifications:* to ensure that no more than one part is on a plant at a time,
3. *Plant-activity specifications:* to restrict the number of plants cooperating on a part to a maximum of one stationary plant and one transport plant,
4. *Cell-part limit specifications:* to limit the number of parts in the workcell, and
5. *Cell-part-type limit specifications:* to limit the number of parts of each part type in the workcell.

A sixth set, part arrival and part departure specifications, is included as one of the safety specifications. It is used to distinguish between parts that are inside and outside of the workcell. All safety specifications are discussed in Williams.[15]

Linear Part-Routing Specifications

Linear part-routing specifications are used to enforce routes of parts, that is, the sequence of operations and the plants that will perform those operations. They are generated from the generic *part-routing specification* templates, as follows, which are shown in *Figure 4*.

State 1: A part is being removed from a stationary plant by a transport plant.

State 2: A part is on a transport plant that is idle.

State 3: A part is being placed onto a stationary plant by a transport plant.

State 4: A part is on an idle stationary plant.

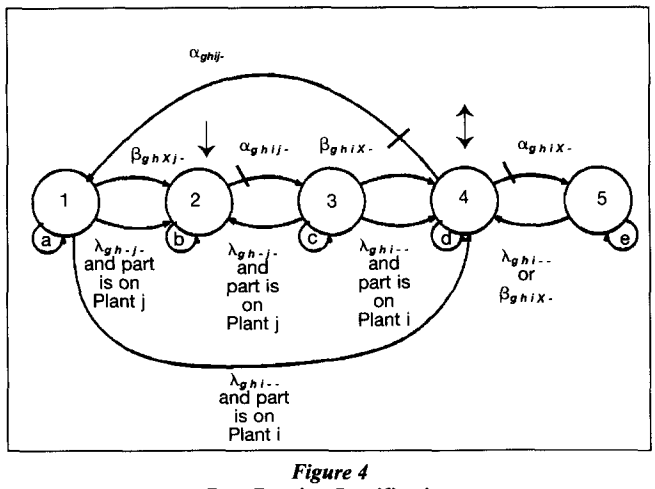

Figure 4
Part-Routing Specification

State 5: A stationary plant is performing a manufacturing operation on the part.

3.2–The New DES Supervisor Design

A standard approach must be followed for the synthesis of DES-based supervisors, such as the four-step procedure described in subsection 2.2. However, it should be noted that any change in the specification requires the repetition of procedure steps 3 and 4, namely resynthesis of a new supervisory strategy.

Steps 3 and 4 generally cannot be performed on-line because they would be computationally very intensive. Therefore, the task is to circumvent steps 3 and 4 and allow linear part-routing specifications to be changed quickly without human intervention.

The proposed solution is a DES supervisor consisting of two modular supervisors (*Figure 5*). The first is synthesized from the safety specifications. The second is a conjunction of linear part-routing specifications, which are treated as modular (sub)supervisors. Only the linear part-routing specifications are regenerated during run time. The safety specifications are not changed during run time; thus, neither is their modular supervisor.

Because the specifications of the modular supervisor for the safety specifications would not be changed during run time, conflicts between these specifications could be removed off-line. By designing the linear part-routing specifications so that they

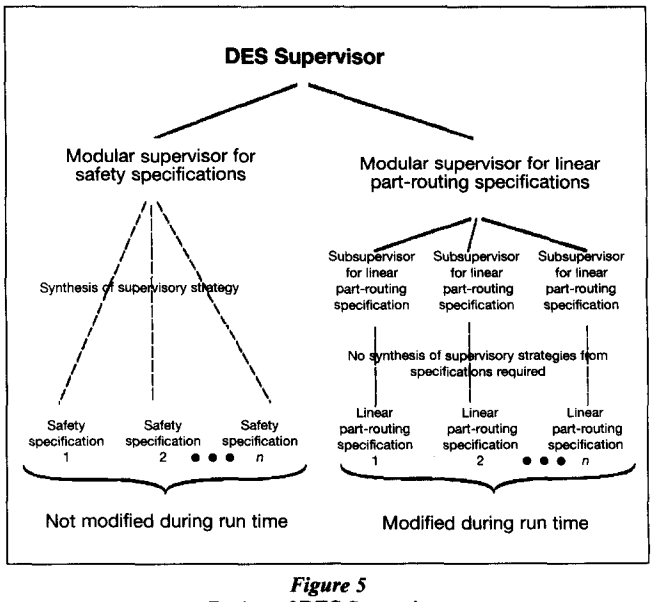

Figure 5
Design of DES Supervisor

only shared one type of event (type μ events), which are self-looping for every state of their strategies, the linear part-routing specifications would not conflict with each other either. Therefore, the only remaining possibilities for conflicts would be between the specifications of the two modular supervisors. The nonblocking property of two modular supervisors is not necessarily closed under conjunction. If no conflicts are found, no specifications need to be revised; hence, resynthesis is not required.

3.3–Dealing with Conflicts

The standard conflict-analysis procedure maps the state-spaces of the two modular controllers onto one centralized state-space, which is then reviewed. Due to the size of the route-space, conflict recognition and blocking removal procedures quickly become overwhelmed. In general, conflicts may be resolved by any of the following four approaches, either alone or in combination:[15]

1. Revising or adding specifications to restrict operation of the workcell and eliminate the conflicts.
2. Using heuristics to ensure that certain transitions that result in blocking are not chosen.
3. Extending markings to allow a larger class of languages to be nonconflicting.
4. Ignoring conflicts if they are expected to occur sufficiently infrequently. If blocking does occur, it could be resolved by some external intervention.

3.4–Implementation of DES Supervisor

The strategy of the two modular supervisors of the DES supervisor is encoded in lookup tables of event descriptions. Some tables are embedded into the computer code of the implementation. This is the case for the modular supervisor for the safety specifications. Other tables were not embedded into computer code. They remain in the form of tables so that their contents may be revised during run time. This is the case for the modular supervisor for the linear part-routing specifications.

Control synthesis is enforced in a two-step procedure. First, a set of controllable events is enabled according to the strategy of the DES supervisor (all uncontrollable events are always enabled). Second, when an enabled event occurs, it is processed by making the appropriate transitions in the tables and by updating the hypothesis about the current state of the system.

As will be discussed in Section 5, the ASD is responsible for rerouting parts. This task includes generating new linear part-routing specifications, which are transferred to the DES supervisor. There are two types of linear part-routing specifications generated by the ASD: full and partial. These are used in place of the nominal linear part-routing specifications, which a priori reside in the DES supervisor.

A nominal linear part-routing specification defines the production route for a part type from start to finish. The nominal specifications, in addition to being the preferred routes, are the reference routes and are never altered. The full specification is an alternate complete route created at run time for the production of a part type. Partial specifications are used to continue the production of parts that are partially manufactured. When the production of the part at hand is completed, the partial specification is discarded and a full specification is used for the next part.

4–The Diagnostic System

4.1–Model-Based Diagnosis

Davis[16] suggests a basis on which to decide on the use of a model-based approach over others: if there are subtle and complicated interactions in a device, if it is difficult to predict the outcome of these interactions, or if the knowledge of the device is truly anecdotal and empirical, it will probably be too difficult to model the device. Instead, it would be easier to observe the device and capture the experience in the form of rules. The model-based approach, on the other hand, is appropriate if the structure and behavior are too complex for exhaustive simulation to be practical, but well enough understood to be modeled. Although both fault trees and expert systems are commonly used in FMSs,[17] it would appear that a model-based approach would be most suited to a workcell supervised with a DES-type supervisor.

The general diagnostic engine (GDE) of de Kleer and Williams[18] was chosen as the means for meeting the needs of the HSC in regard to failure detection. The GDE is a model-based diagnostic technique. Its decisions are based on the knowledge that a device must be faulty if its behavior is inconsistent with its model. The interaction of observation and prediction is the basic paradigm of model-based reasoning for diagnosis. A

GDE-based diagnostic system would, in general, be implemented as described in the following sections.

Modeling

It is assumed that the workcell is a single device. The plants within the workcell are the components of this device, described in a hierarchical manner (*Figure 6*). Connections are used for modeling interactions between the components of the workcell.

Fault Detection

The failure of a component is detected by the occurrence of a discrepancy between the model and the observation.

Fault Isolation Strategy

A workcell controller would only be interested in knowing which plants have failed. Thus, when a fault is isolated to several subcomponents of a plant, the on-line search can be stopped and the plant declared as "failed." Later, when the plant is being repaired, the GDE could be used off-line to determine which of the components in the plant are faulty. However, if a failure occurs while two plants are interacting, the GDE must acquire more information to determine whether one or both plants have failed.

4.2–Implementation of Diagnostic System

A simplified diagnostic system based on the generic GDE approach was developed and used in this work. The new system preserves some of the most beneficial aspects of the generic GDE approach and allows for the implementation of the "full-blown" GDE in the future.

Modeling

As mentioned before, during run time, when a failure occurs it is only necessary to determine which plant contains the fault. For the case of a plant operating by itself, this is a relatively simple task. Plant interactions, however, must be handled explicitly using special models. A model would be required for each pair of a transport plant and a stationary plant. Because it is assumed that stationary or transport plants cannot interact among themselves, no models are required for such combinations.

The second step in simplifying the diagnostic mechanism is the use of specialized models, where the source of the fault is directly implied by the model that detected that fault. In specific regard to transport-stationary plant interactions, it is assumed that the failed plant would identify itself; otherwise, both plants are declared as "failed."

Fault Detection and Isolation

The result of the above simplifications of the diagnostic mechanism is a library of models for different scenarios. The implementation problem now becomes knowing which models to apply and what information to provide them with. Information about the current states of the specifications of the DES supervisor can be used in the solution of this problem.

As discussed in Sections 2 and 3, in a DES supervisor each state defines which plants are operative, where each part is, which part each plant is operating on, and so on. However, a state is not a declaration of how the system is; rather, it is a hypothesis of what the state of the manufacturing system should be. The task is to verify that the hypotheses

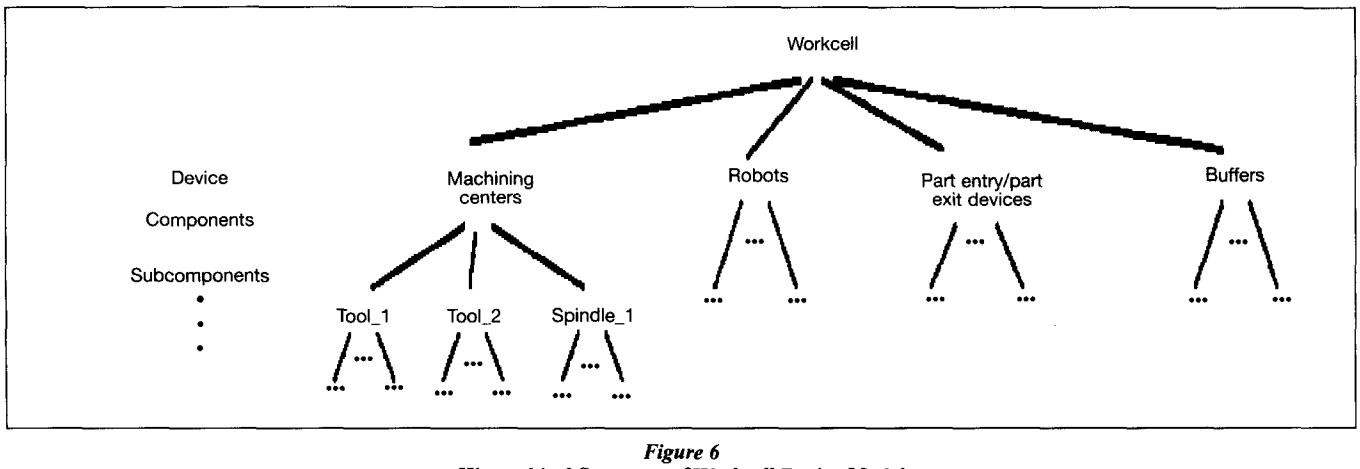

Figure 6
Hierarchical Structure of Workcell Device Model

agree with reality. The diagnostic mechanism developed can use the hypotheses built into each state of the DES supervisor as the basis for choosing which models to use to check the workcell. The diagnostic mechanism thus performs two tasks simultaneously: failure detection and part-location verification:

Failure detection: The proposed failure-detection strategy proceeds through three stages to identify plants which have failed. In the first stage, all parts that are in the workcell are considered sequentially. For a chosen part, the type of activity performed on it is checked. The four possible outcomes, when checking the part-activity specifications and pertinent linear part-routing specifications, could be that:

- No plant is operating on the part,
- A stationary plant is operating on the part,
- A transport plant is operating on the part, or
- Both a stationary plant and a transport plant are operating on the part.

In the second and third stages, all other stationary and transport plants that were not checked during the first stage are checked, respectively.

Verification of part location: The objective is the verification of the location of the parts. In cases where it is permissible for a part to be on one of several plants, the identity of the plant on which the part resides must also be determined.

Interaction of Diagnostic Mechanism with Controllers

The diagnostic mechanism would interact with the controllers whenever plants have failed or were to be repaired. The sequence of operations on detection of a plant failure is as follows:

1. Detection of failures.
2. Generation of a list of failed plants.
3. Sending of corresponding λ-event signals to the DES supervisor.
4. Sending a list of the failed plants to the ASD along with a list of the locations of the parts in the workcell.
5. In response to Step 4, the ASD:
   5.1. Disables all α-type events of the DES supervisor to ensure that the DES supervisor does not allow new operations to begin in the workcell.
   5.2. Determines new linear part-routing specifications.
   5.3. Asks the diagnostic mechanism for a confirmation about the most recent state of the system.
   5.4. Transfers the new linear part-routing specifications, and the new present states of the parts, to the DES supervisor.
   5.5. Returns control to the DES supervisor.

The interaction process on the repair of a plant is as follows:

1. Generation of a list of repaired plants.
2. Sending corresponding μ-event signals to the DES supervisor.
3. Sending the list of repaired plants to the ASD.
4. Repeating steps 5.1-5.5 of the above procedure for dealing with plant failures.

5–Alternate-Strategy Driver

During run time, a scheduled production plan might have to be revised in response to the occurrence of unplanned events. However, a global optimal solution cannot be achieved for the rescheduling problem due to time restrictions. Instead, only a portion of the scheduled production plan can be modified. The routing level is the most amenable to, and capable of, delivering quick responses to the altered conditions. So, now the efforts of rescheduling the production plan are focused on the routing level. In this context, the ASD reroutes parts by replacing their linear part-routing specifications in the DES supervisor with ones that constitute the new routes.

5.1–Choosing a Rerouting Strategy

A "mixed" rerouting strategy was developed in this work. It combines many of the positive attributes of the off-line and on-line approaches reported in the literature.[19-21] Specifically, however, the scheduling approaches of Camarinha-Matos and Steiger-Garcao,[22] Long et al.,[23] and Hadavi et al.[24] had a significant impact on the design of the implemented mixed approach. Two of these are briefly discussed below.

Camarinha-Matos and Steiger-Garcao conceptualize and partly implement a mixed approach. In the off-line phase, a generic production plan is generated for the whole production run. The on-line system enforces the generic production plan, although it can make small adjustments to its actions in response to

sensory feedback. In the case of an unscheduled event, it instigates subplans, but it does not replace the general production plan. Long et al.'s approach is of particular interest because it is a rare example of a scheduling approach that must cooperate with a state-based supervisory controller. Their supervisory controller is constructed of Petri nets, whereas the HSC in this work is based on DES theory.

The mixed-rerouting approach proposed and implemented in the HSC is simpler than those mentioned above. In this case, the "best" routes, referred to as the nominal routes, represent the original routes of the scheduled production plan, and they are assumed to have been provided to the HSC. Deviation from a nominal route is allowed whenever it cannot be maintained. This occurs under circumstances of unmodeled events, or deadlocks. Instead of generating subplans for each portion of a nominal route that cannot be maintained, "complete routes" are generated. These routes correspond to the full or partial linear part-routing specifications discussed in Section 3.

5.2–Definition of a Route and Its Descriptive Terms

A route is a path of production (that is, a sequence of manufacturing and transport operations) through the resources of the manufacturing system. To facilitate the discussion of the rerouting strategies, the different units of the route are defined in this section. These definitions were inspired by the descriptive language developed by Adlemo et al.[25] for describing their failure-robust manufacturing system, namely the general recursive system (GRS). The algorithms of the ASD construct and combine different units and subunits into a complete route.

- *Entities*
  An entity is a set of plants (individually referred to as *atomic* entities).
- *Mission*
  A mission is a part arrival, part departure, or machining operation, henceforth collectively termed manufacturing operations. One such operation is referred to as an atomic mission.
- *Path*
  An entity path is a sequence of transport operations (atomic paths) that takes a part from one stationary plant to another.
- *Production-stage entity*
  A production-stage entity is a combination of: (1) an (entity) path that a part will follow to a stationary plant, where the manufacturing operation of this production stage is to be performed and (2) an atomic mission, namely the specific manufacturing operation.
- *Route*
  A route is a sequential collection of production-stage entities. Complete routes appear in two forms: (1) part-production routes, namely sequences of atomic paths and missions derived from the production-stage entities and (2) routing specifications, namely linear part-routing specifications used in the DES supervisor, which are in the form of event-transition lookup tables. The tables are converted forms of the sequences of the part-production routes.
- *Serviceability*
  If a unit, such as an atomic path, atomic mission, or entity path, can be performed, it is referred to as being serviceable.

5.3–Implementing the ASD

In most manufacturing systems, there will be many possible part-production routes for each part (that is, many paths and/or manufacturing operations available for the objective at hand). A mapping of these routes forms the part-production route space.

An optimization procedure can be used to maintain the nominal route, as per the goal of the proposed strategy, and which chooses alternate solutions only when the nominal entities are not serviceable. To achieve this objective, costs are assigned herein to the use of each atomic entity (that is, a plant) which services an atomic path (a transport operation) or an atomic mission (a manufacturing operation).*

The optimization procedure for finding an alternate route developed here consists of four levels, where each level has absolute priority over all the other levels underneath it. This four-level procedure drives a recursive tree search of the part-production route space. As it searches, it constructs a part-production route—one (serviceable) production-stage entity at a time.

*Although in these simulations the costs were set arbitrarily, they could have been empirically measured or arrived at through some analytical methods.

Level 1 returns an atomic mission for the production-stage entity that is being constructed. Level 2 returns an atomic entity to service this atomic mission. Level 3 returns an entity path that will take the part to the atomic entity of Level 2. Level 4 returns an atomic entity for each of the atomic paths, which form the entity path selected in Level 3. Optimal production-stage entities, determined during the four-level search, are then compiled into a route.

The "preferred" plants and paths used during the optimization mark the nominal route. Because of the manner in which the procedure is designed, when a preferred unit can be used in a route that takes a part to completion, it will be selected and none of its alternatives will be considered. This is needed for enforcing the return to the nominal route, when failed plants have been repaired.

The implemented mixed approach would generally have a very short search time because the strategy of maintaining the nominal route significantly reduces the amount of route space that is searched.

5.4–Algorithms for Rerouting

There are three steps to producing a new linear part-routing specification within the ASD:

1. Construction and maintenance of lookup tables of serviceable paths. This step corresponds to levels 3 and 4 of the optimization procedure described in Section 5.3. An entry exists in each table for each ordered pair of plants.
2. Compilation of the part-production route. An atomic mission (a manufacturing operation) is selected (optimization level 1) and matched with an atomic entity (a stationary plant) that will service it (optimization level 2). An appropriate serviceable path is retrieved from the tables. These are combined into a production-stage entity. Production-stage entities are strung together to form a part-production route.
3. Conversion of the part-production route into a linear part-routing specification, if the new route contains nonnominal paths or manufacturing operations.

Rerouting Around Failed Equipment

The algorithm for rerouting around failed equipment, ASD_MAIN, is illustrated in global terms in *Figure 7*. It reroutes parts in response to the failure or repair of equipment, whereupon it creates new routes in the form of linear part-routing specifications, which it submits to the DES supervisor.

ASD_MAIN consists of nine regions.[15] Before new routes can be derived, all α-type controllable events permissible by the DES supervisor are disabled (Region A). There are two reasons for disabling controllable events. First, the reroutings made by the ASD reflect the known positions of the parts in the system at the time the derivations are begun. If some parts are allowed to progress to different positions and different levels of production, the routes that are derived may be inapplicable. Second, if the part is allowed to progress along its present course, it may become blocked needlessly.

In Region B, Step 1 in the creation of a linear part-routing specification begins in terms of the compilation of two tables, one for atomic paths and another for entity paths. These tables are used for both full and partial specifications.

Steps 2 to 4 in the creation of full and partial specifications and the corresponding special measures required for creating them are performed in Regions C to F. The special measures include forcing transitions, processing λ events, and saving states and transitions of the linear part-routing specification that is currently being enforced on the part for use in the new partial specification.

The last two sections, G and H, involve the transfer of the new specifications to the DES supervisor and the reenabling of the occurrence of the α events.

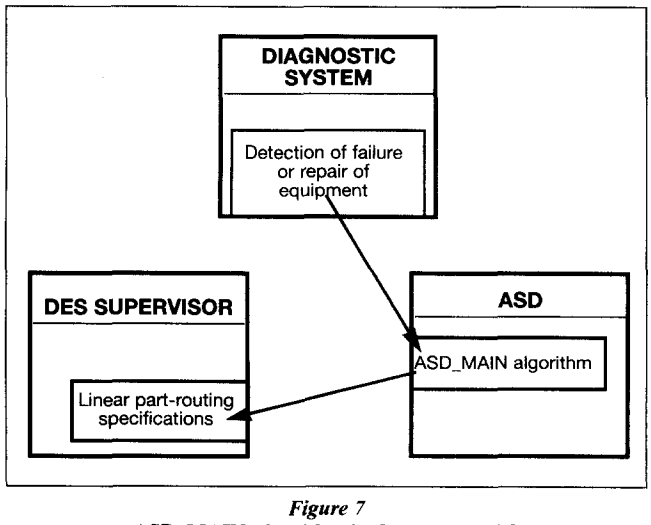

Figure 7
ASD_MAIN Algorithm in Context of HSC

Rerouting Around Deadlocks

Each plant has a set capacity for the number of parts it can hold. The plant specifications ensure that the DES supervisor does not allow a part to be placed on it or picked by another plant that is already full. An unfortunate consequence of these specifications is part lockup, more commonly referred to as deadlock. Tanenbaum[26] identifies two types of deadlocks: direct and indirect.

Direct deadlock involves two parts residing, respectively, on two stationary plants. Both parts are routed to move to the opposing stationary plant via the same transport plant. Neither stationary plant can accept an incoming part due to the full buffers. Therefore, neither part can proceed. An indirect deadlock is the general case of the direct deadlock where several parts are involved. Each part needs to be sent to a neighboring stationary plant. Unfortunately, none of the plants has a free buffer available to receive an incoming part.

Tanenbaum mentions a solution by Merlin and Schweitzer[27] to the problem of deadlocks; they construct a directed graph having nodes that represent buffers. Pairs of buffers are connected by arcs. Packets of information pass between buffers along the arcs of the graph. The graph is constructed in such a way that, if it is followed, no deadlocks can occur.

If the linear part-routing specifications were not treated as modular with regard to the other specifications of the DES supervisor, the supervisory strategy could be searched for blockages and revised so that occurrence of deadlocks would be eliminated wherever possible. The results would be much like Merlin and Schweitzer's. There would be a graph—the DES supervisory strategy—that, if followed, would not permit deadlocks to be entered.

This is a passive approach to dealing with deadlocks, an approach in which deadlocks are resolved by never letting them occur. The HSC, however, requires the linear part-routing specifications to remain modular from the other specifications, so the passive approach is not an option. Instead, an active approach must be taken.

In the active approach, entry into deadlocks is allowed. When deadlocks are entered, the affected parts can no longer be moved. This is an effect of the enforced linear part-routing specifications that limit the α events that may be enabled and the plant buffer specifications that disable all the enabled α events. To counter this effect, the ASD must continually monitor the control-synthesized events of the DES supervisor. If it determines that some α events have been disabled due to deadlock, it initiates rerouting of the parts and subsequent revision of the linear part-routing specifications.

An algorithm, Yield, was developed for recognizing and resolving deadlocks. The algorithm consists of four sections.[15] In Section A, possible deadlock candidates are identified. In Section B, sets of deadlocked parts are constructed from the candidates. A deadlock is resolved by diverting parts so that they may pass around each other. In Section C, paths are derived for diverting parts and stored in customized-path tables. If a diverting path is successfully found for a part, a partial specification incorporating that path is generated in Section D.

In implemented form, the Yield algorithm must work in tandem with the control synthesizer of the DES supervisor. Therefore, Section A is incorporated into the control synthesis stage of the DES supervisor (*Figure 8*). When deadlocks are recognized, a signal is sent to the other sections of Yield, which attempt to reroute the deadlocked parts.

6–Simulation of HSC

The layout of the manufacturing workcell used in the computer simulations is shown in *Figure 9*. Sensors on each piece of equipment monitor the operational behavior of the equipment as well as the placement of parts. Two types of parts are to be manufactured. At most, three parts, or two of each part type, are permitted within the workcell.

Preparing the HSC for the simulation of the exemplary workcell required:

- Definition of part-production-stage plans, which define the nominal part-production routes as well as all alternate units, for each part type,
- Construction of a DES supervisor using appropriate sets of specifications,
- Creation of models for the diagnostic mechanism, and
- Definition of costs for using alternate plants for alternate operations.

The nominal route for the production of a part of Type 1 was defined as follows:

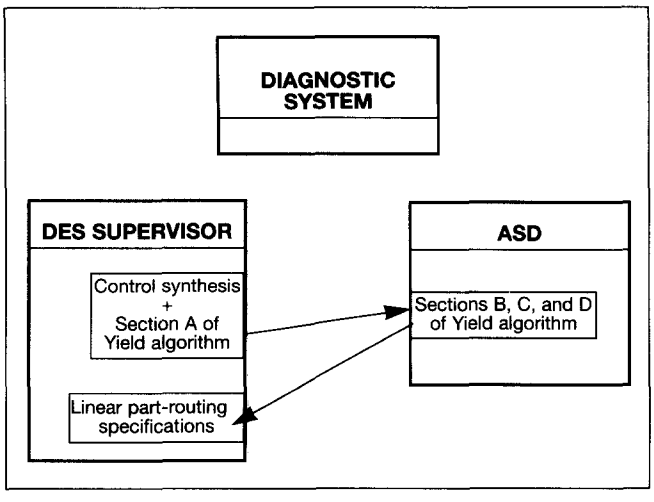

Figure 8
Yield Algorithm in Context of HSC

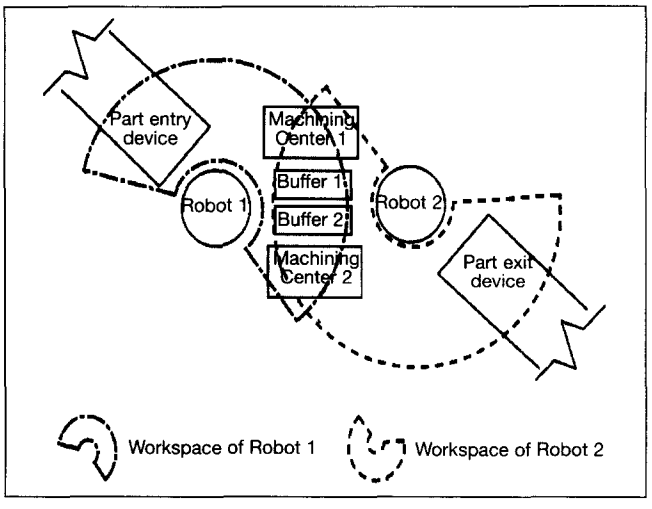

Figure 9
Arrangement of Equipment and Workspaces of Robots

1. Part arrives at part-entry device.
2. Robot 1 moves the part to Machining Center 1.
3. Machining Center 1 performs a milling operation on the part.
4. Robot 1 moves the part to Machining Center 2.
5. Machining Center 2 performs drilling operation #2 on the part.
6. Machining Center 2 performs drilling operation #1 on the part.
7. Robot 2 moves the part to the part-exit device.
8. Part departs the workcell.

The nominal route for the production of a part of Type 2 was defined as follows:

1. Part arrives at part-entry device.
2. Robot 1 moves the part to Machining Center 2.
3. Machining Center 2 performs drilling operation #1 on the part.
4. Robot 1 moves the part to Machining Center 1.
5. Machining Center 1 performs a milling operation on the part. 6. Robot 2 moves the part to the part-exit device.
7. Part departs the workcell.

Because the HSC was implemented on a Sun Sparc workstation in the C language, event occurrences were input via the keyboard in a random fashion. Namely, at every workcell state, once the HSC has compiled the feasible part routings, a set of allowable events are displayed on the monitor. Then, an event can be randomly selected and input into the program. In a similar fashion, sensory inputs can also be input at any time because the HSC diagnostic module continuously monitors for such inputs.

To test the operation of the HSC, Machining Center 2 was disabled. When simulated, the HSC successfully rerouted parts of Type 1 and 2 using alternate machining operations on Machining Center 1 (to replace those previously carried out on Machining Center 2).

The alternate route for the production of a part of Type 1 was generated as follows:

1. Part arrives at part-entry device.
2. Robot 1 moves the part to Machining Center 1.
3. Machining Center 1 performs a "combined" drilling operation on the part (replacing the previous two drilling operations performed on Machining Center 2).
4. Machining Center 1 performs a milling operation on the part.
5. Robot 2 moves the part to the part-exit device.
6. Part departs the workcell.

The alternate route for the production of a part of Type 2 was generated as follows:

1. Part arrives at part-entry device.
2. Robot 1 moves the part to Machining Center 1.
3. Machining Center 1 performs drilling operation #1 on the part.
4. Machining Center 1 performs a milling operation on the part.
5. Robot 2 moves the part to the part-exit device.
6. Part departs the workcell.

Also, a test (via computer simulation as well) to determine whether the HSC could successfully respond to a part deadlock was achieved by allowing one part of each part type to enter the workcell and to complete the first machining operations. Following the first machining operations, the route of each part would take it to the location of the other. Nominally, they would be transported by Robot 1, but, Robot 1 cannot transport two parts at the same time.

It was noted that the HSC successfully resolved this deadlock by assigning Robot 1 to transporting the first part and Robot 2 to transporting the second part. If the first transport operation starts by removing part type 2 from Machining Center 2, Robot 1 will do the task. Robot 2 will be responsible for moving part type 1 from Machining Center 1 to Machining Center 2.

It must be noted that the resolution of the deadlock required the use of an alternate robot to transfer parts between the two machining centers. Were this scenario repeated with Robot 2 being failed, the part on Machining Center 2 would first be placed temporarily in Buffer 1. Robot 1 would then carry out all the transfer operations between the machining centers and the buffer.

In summary, during various simulations, the production of the parts was exposed to failure of equipment at different points along the production of the parts. Tests showed that: (1) the ASD can reroute parts as prescribed, (2) deadlocks can be actively resolved, (3) the DES supervisor can successfully enable and disable operations within the workcell, and (4) the DES supervisor can continue to function when modified to accommodate changes in the workcell environment.

7–Conclusions

Control strategies based on the use of the HSC proposed in this paper are fixed at some level of capability in dealing with unplanned, and thus unmodeled, events. In other words, the HSC, like all other deterministic systems (for example, knowledge-based, controlled automata, and Petri nets), cannot effectively cope with plant failures that were not thought of and accounted for a priori. However, unlike with other systems, the a priori consideration and modeling of a large number of possible events does not negatively affect the HSC.

For instance, a (pure) DES supervisory controller of equal capability to that of an HSC could be constructed. Unfortunately, this would require a significant amount of computation time, and the resulting DES supervisory controller would have a very large number of states. For example, the supervisory strategy for the simulation case examined in this paper required the utilization of only 1082 states by the HSC, where a (pure) DES-based supervisory controller would have required in excess of 10^{28} states.

The reason for this large difference is the principal feature of the HSC in not requiring the generation of the complete DES control strategy, but only the generation of the part routes. This turns out to be the real advantage of the HSC over the standard DES approach, or any other method alike it. The HSC manages to maintain the DES formalism without being defeated by an unmanageable number of states.

Finally, this work is also original in its application of a model-based technique to the run-time diagnosis of a workcell. However, it must be noted that the modular nature of the HSC allows the use of any other type diagnosis scheme.

Acknowledgment

This work was partially funded by the Natural Sciences and Engineering Research Council of Canada.

References

1. J. Kimemia and S.B. Gershwin, "Flow Optimization in Flexible Manufacturing Systems," *International Journal of Production Research* (v23, n1, 1985), pp81-96.

2. C.Y. Chen, "Development of a Manufacturing Workcell Management System," Master's Thesis (Toronto: University of Toronto, Dept. of Mechanical Engineering, 1989).

3. B. Benhabib, C.Y. Chen, and W.R. Johnson, "An Integrated Manufacturing Workcell Management System," *Manufacturing Review* (v2, n4, 1989), pp266-276.

4. J. Long, B. Descotes-Genon, and P. Ladet, "Hierarchical and Intelligent Control of Flexible Manufacturing Systems," 7th IFAC Symposium on Information Control Problems in Manufacturing Technology (Toronto: 1992), pp243-248.

5. R. David, "Modeling of Dynamic Systems by Petri Nets," ECC91 (European Control Conference) (Grenoble, France: 1991), pp136-147.

6. J.S. Ostroff, "Deciding Properties of Timed Transition Models," *IEEE Transactions on Parallel and Distributed Systems* (v1, n2, 1990), pp170-183.

7. J. Ostroff and W.M. Wonham, "A Framework for Real-Time Discrete-Event Control," *IEEE Transactions on Automatic Control* (v35, n4, 1990), pp386-397.

8. L.M. Camarinha-Matos and A. Steiger-Garcao, "Robotic Cell Programming: A Knowledge-Based Approach," *Robotics and Artificial Intelligence '86* (1986), pp533-551.

9. B.A. Brandin, W.M. Wonham, and B. Benhabib, "Discrete Event System Supervisory Control Applied to the Management of Manufacturing Workcells," *Computer-Aided Production Engineering,* C. Venkatesh and J.A. McGeough, eds. (Amsterdam: Elsevier, 1991), pp527-536.

10. Y-C. Ho, "Performance Evaluation and Perturbation Analysis of Discrete Event Dynamic Systems," *IEEE Transactions on Automatic Control* (v32, n7, 1987), pp563-572.

11. B.A. Brandin, W.M. Wonham, and B. Benhabib, "Manufacturing Cell Supervisory Control,A Timed Discrete-Event System Approach," IEEE International Conference on Robotics and Automation (Nice, France: 1992), pp931-936.

12. B.A. Brandin, W.M. Wonham, and B. Benhabib, "Manufacturing Cell Supervisory Control,A Modular Timed Discrete-Event System Approach," IEEE International Conference on Robotics and Automation (Atlanta, GA: 1993), pp847-851.

13. S. Balemi, "Discrete Event Systems Control of a Rapid Thermal Multiprocessor," 7th IFAC Symposium on Information Control Problems in Manufacturing Technology (Toronto: 1992), pp53-58.

14. P. Ramadge and W.M. Wonham, "The Control of Discrete Event Systems," *IEEE Proceedings* (v77, n1, 1989), pp81-98.

15. R.A. Williams, "A Hybrid Supervisory Control System for Flexible Manufacturing Workcells," Master's Thesis (Toronto: University of Toronto, Dept. of Mechanical Engineering, 1993).

16. R. Davis, "Diagnostic Reasoning Based on Structure and Behaviour," *Journal of Artificial Intelligence* (v24, 1984), pp347-410.

17. M.G. Abu-Hamdan and A.S. El-Gizawy, "An Error Diagnosis Expert System for Flexible Assembly Systems," 7th IFAC Symposium on Information Control Problems in Manufacturing Technology (Toronto: 1992), pp451-456.

18. J. de Kleer and Brian C. Williams, "Diagnosis with Behavioral Modes," 11th International Joint Conference on Artificial Intelligence (Detroit: 1989), pp1324-1330.

19. F.A. Rodammer and K.P. White, "A Recent Survey of Production Scheduling," *IEEE Transactions on Systems, Man and Cybernetics* (v18, n6, 1988), pp841-851.

20. C.S. Tang and E.V. Denardo, "Models Arising from a Flexible Manufacturing Machine, Part I: Minimization of the Number of Tool Switches," *Operations Research* (v36, n5, 1988), pp767-777.

21. M.J. Shaw, "Dynamic Scheduling in Cellular Manufacturing Systems: A Framework for Networked Decision Making," *Journal of Manufacturing Systems* (v7, n2, 1988), pp83-94.

22. L.M. Camarinha-Matos and A. Steiger-Garcao, "Robotic Cell Programming: A Knowledge-Based Approach," RIA/IPAR '86 (Toulouse, France: 1986), pp533-551.

23. J. Long, B. Descotes-Genon, and P. Ladet, "Distributed Intelligent Control and Scheduling of Flexible Manufacturing Systems," 8th International Conference on CAD/CAM, Robotics and Factories of the Future (Metz, France: 1992), pp1760-1767.

24. K. Hadavi, M.S. Shahraray, and K. Voigt, "ReDS,A Dynamic Planning, Scheduling, and Control System for Manufacturing," *Journal of Manufacturing Systems* (v9, n4, 1990), pp332-344.

25. A. Adlemo and S-A. Andreasson, "Fault Detection in Manufacturing Systems with Data Network Partitions," *Proceedings* of 1992 IEEE International Conference on Robotics and Automation (Nice, France: 1992), pp975-980.

26. A.S. Tanenbaum, *Computer Networks,* 2nd ed. (Englewood Cliffs, NJ: Prentice-Hall, 1988), pp289-320.

27. P.M. Merlin and P.J. Schweitzer, "Deadlock Avoidance,Store-and-Forward Deadlock," *IEEE Transactions on Communications* (v28, 1980), pp345-354.

Authors' Biographies

Mr. R.A. Williams obtained his BASc in engineering science (manufacturing option) in 1987 and his MASc in mechanical engineering in 1993, both from the University of Toronto. This paper was part of his MASc thesis in the area of automatic supervisory control of manufacturing systems.

Dr. B. Benhabib obtained his PhD in mechanical engineering in 1985 from the University of Toronto, where he is currently an associate professor in the Department of Mechanical Engineering. His research interests are in the general area of computer-integrated manufacturing. He is a senior member of SME and is a member of ASME, IEEE, and AAAI. He is a registered professional engineer.

Dr. K.C. Smith obtained his PhD in physics in 1960 from the University of Toronto, where he is currently a professor in the Department of Electrical and Computer Engineering. His research interests include analog VLSI, multiple-valued logic, and flexible manufacturing. He is a fellow of IEEE and a registered professional engineer.