A Reconfigurable Hybrid System and Its Application to Power Plant Control

Humberto E. Garcia, Asok Ray, *Senior Member, IEEE,* and Robert M. Edwards, *Member, IEEE*

Abstract— **This paper presents a reconfigurable approach to implement decision and control systems for complex dynamic processes. The proposed supervisory control system is a reconfigurable hybrid architecture structured into three functional levels of hierarchy, namely, execution, supervision, and coordination. While the bottom execution level is constituted by either reconfigurable continuously varying or discrete-event systems, the top two levels are necessarily governed by reconfigurable sets of discrete-event decision and control systems. Based on the process status, the set of active control and supervisory algorithm is chosen. The underlying concept of the proposed reconfigurable hybrid decision and control is described along with its implementation on the feedwater system at the Experimental Breeder Reactor II of the Argonne National Laboratory. The performance of the hybrid system to accommodate component failures is then evaluated in an in-plant experiment.**

I. Introduction

COMPLEX dynamic processes are usually difficult to control because they are: i) operated over wide ranges, ii) subjected to unpredictable or partially predictable large disturbances, and iii) controlled by spatially distributed sensors and actuators. To meet the performance specifications, the control system should include different strategies for each individual operating condition, adaptive behavior to react under uncertain or unfamiliar situations, and the capability to coordinate distributed controllers to accomplish the system task in a coherent way. Large scale integrated systems have been designed using the concepts of centralized and distributed controls [14]. These control techniques, however, although implemented via computers, are often based on the existing design methodologies of analog technology and continuously varying systems (CVS's). The CVS's have dynamics described by differential or difference equations. In contrast to analog technology, digital computers can effectively store, process, and communicate system information providing a means for other process modules to access remotely generated information. Often, ad hoc techniques based on practical experience and ingenuity of the designers are used to partially satisfy local and global performance criteria. These techniques are application-dependent, however, and they do not address the generality associated with an effective theoretical framework.

To take the advantage of enhanced control capabilities, control systems implemented via digital computers and based on computer technology should be analyzed and synthesized in a computer framework which functions as discrete-event systems (DES's). Specifically, DES's are causal, dynamic, asynchronous, and logical. The evolution of these systems is better described by interactions of discrete events rather than by differential or difference equations. By combining the CVS and DES concepts, control systems can be effectively synthesized and implemented to execute dynamic commands that are directed to coordinate their control functions as a whole. Dynamic systems whose behavior results from the interaction of continuous-time processes with discrete-event processes are called hybrid systems or intelligent controller [11]. Recently, there has been an increasing interest in hybrid dynamical systems as asserted from many recent papers and books (e.g., [1] and [12]). These systems appear in many engineering applications such as process control and manufacturing systems (e.g., [2], [15], [9], [19]). For process control, the continuous part of a hybrid system generally represents a given physical process needed to be controlled while a discrete-event system represents a supervisor which reacts to events generated by the continuous process to achieve the specified system performance. In reconfigurable hybrid systems, as presented in this paper, the set of operating processes, either continuously varying or discrete-event, could be time dependent. Consequently, CVS and DES techniques are combined into a hybrid system for decision and control. The proper operation of a hybrid system relies in part on a framework for the control of DES's. Numerous approaches to modeling, analysis, synthesis, and control DES's have been reported in the literature (e.g., [2], [3], [6], [13], [15], [16], [20], [21]). A review on the control of DES's is provided in [22] while the class of DES's capable of reconfiguration has been recently addressed in [6], [7], and [9].

This paper presents the concept and related formulations of a reconfigurable hybrid supervisory system in a general manner. The efficacy of the proposed control concept is demonstrated by its implementation on the feedwater system at the Experimental Breeder Reactor II (EBR-II) of the Argonne National Laboratory. The significance of this application is discussed below.

The potential reduction in the net positive suction head (NPSH) to feedwater pumps and the associated damage during transient operations of power plants were identified as a

Manuscript received January 7, 1994; revised August 10, 1994. Recommended by Associate Editor, G. Davis. This research work was supported in part by the Department of Energy Grant DE-FG-07-89ER 12889 and Contract W-31-109-Eng-38.

H. E. Garcia is with the Argonne National Laboratory, Idaho Falls, ID 83403-2528 USA.

A. Ray is with the Mechanical Engineering Department, The Pennsylvania State University, University Park, PA 16802 USA.

R. M. Edwards is with the Nuclear Engineering Department, The Pennsylvania State University, University Park, PA 16802 USA.

IEEE Log Number 9410598.

problem that might be effectively addressed by the proposed reconfigurable hybrid supervisory system. In particular, the feedwater system at EBR-II is equipped with a microprocessor-based control system and was made available for conducting a validation experiment. The test demonstrated that the proposed control system was capable of safely and significantly extending the plant operating range whereas the existing conventional control system would have failed within a few seconds.

The paper is organized in eight sections, including the Introduction, as follows. Section II presents a general reconfigurable supervisory control scheme for process control while Section III illustrates the resulting hybrid system. The framework used for modeling DES's is then indicated in Section IV. Section V shows the physical process selected to conduct an in-plant demonstration as well as the control problem at hand and the proposed solution. Section VI discusses the implemented reconfigurable hybrid system in EBR-II, and Section VII describes the experimental realization and results obtained from this in-plant demonstration. Conclusions are given in Section VIII.

II. Architecture of the Reconfigurable Supervisory Control System

Decision making and control in complex dynamic processes such as a power plant have certain unique features. As these processes are composed of a large set of distributed and possibly inter-related subprocesses, characterized by a wide range of operating conditions and often subjected to partially predictable or unpredictable disturbances, it is difficult to formulate mathematical models for process control. Thus, when devising supervisory control systems (SCS's) for complex processes, the SCS's should have adaptive behavior to properly react under uncertain or unfamiliar situations, different control techniques available to execute the control policy required for the current operating conditions, and the capability to supervise the operation of control modules to achieve the system-wide objective.

The proposed SCS is hierarchically structured into three levels, namely, the execution, supervision, and coordination levels. These layers are in part ordered according to the principle of increasing intelligence with decreasing precision [22]. The execution level refers to the process dynamics. In particular, the execution level generates direct control actions over the process (or subprocesses). The entities responsible for implementing the control policies would be named executors. In general, an executor is capable of exercising more than one control algorithm by incorporating a set of available subcontrollers. These executors are governed by supervisors at the supervision level. The supervisors assign control patterns to the executors to select the control policies to be used on the plant. And, at the highest layer of the hierarchical structure, the coordination level overlooks the operations of the supervisors. For simplicity, a single entity called the coordinator is defined to implement this layer. The coordinator dynamically selects appropriate supervisors and assigns strategy patterns to them to assure that the process behaves in a desirable manner.

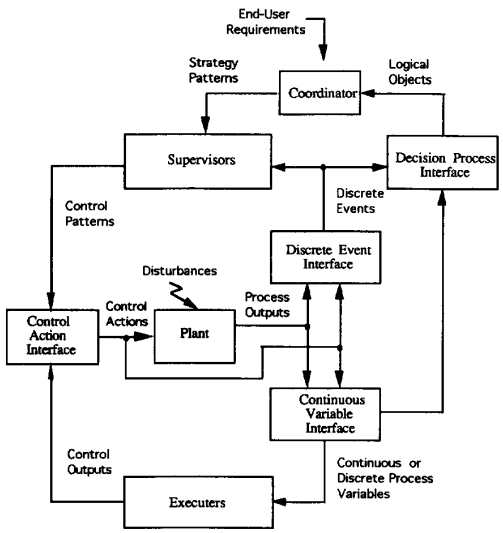


Fig. 1. Abstract models in the supervisory control system.

Because process information is treated differently, each level operates at different time scales and levels of abstraction as seen in Fig. 1. Specifically, this hierarchy accommodates a foreground–background operation with the execution level operating at the foreground and the other two levels operating at the background in support of the foreground operations. Similarly, different models describe the process at each level. At the execution level, process variables are observed and controlled directly and quantified by numerical or symbolic values. In general, the dynamics of the process is better described by differential or difference equations and control algorithms are usually designed according to continuous or discrete time dynamics models. Symbolic models can also be used to devise controllers that mainly perform symbolic processing of the information.

On the other hand, at the supervision level, process variables are observed and controlled indirectly. Process information is better represented by event-driven rather than by time-driven dynamics. Specifically, the dynamics of the processes are described by discrete-event equations, and supervisory algorithms are usually designed based on discrete-event models or DES's [3]. Finally, at the coordination level, process information is used for decision making. The resulting decision models define the hypothetical states of the process with every decision problem exhibiting four elements: acts, outcomes, probabilities, and payoff. Thus, a unique physical process is analyzed under three different contexts, that is, as a continuous (or discrete) variable system at the execution level, as a DES at the supervision level, and as a decision system at the coordination level. Because decision systems are modeled in the discrete-event settings, the coordination level is also a discrete-event process.

In general, a complex process may originate two implementation problems for supervisory control. First, with respect to the execution level, there may not exist a single control algorithm which can satisfy the performance criteria for the complete spectrum of process operating conditions. Second,

with respect to the supervisory level, a single "overly complex" supervisor may be required to accommodate the overall supervising process requirements. This may unnecessarily complicate the formulation, verification, maintenance, and upgrading of the supervisory system. To overcome these problems, reconfiguration at both levels is proposed. A reconfigurable strategy is a dynamic scheme that identifies the current operating status of the controlled environment, evaluates control performances by dynamically monitoring process behavior, and then selects the appropriate system configuration based on this process observation as discussed below.

A. Reconfigurable Executing Strategy

To facilitate the analysis and synthesis of control algorithms, the divide-and-conquer approach [4] is well suited for complex processes. Specifically, the operating range of the process is partitioned into a number of mutually exclusive and exhaustive subranges and then controllers are designed for each subrange to meet the specified performance criteria. The transitions among subranges may cause the control system to switch from one control scheme to another. When there is more than one controller for a given subrange, dynamic control-confident measurements computed based on observed performance are assigned to each controller. The controller with the highest confidence value is chosen as the current best controller to act on the plant. Learning algorithms [17] can then be used by the supervisors to select the control strategy expected to be the current optimal choice.

B. Reconfigurable Supervisory Strategy

Similarly, the DES model of the process is partitioned into a number of DES subsystems. Supervisors are designed for each of the subsystems, and the overall system response is then obtained by reconfiguring the supervising strategy based on current state of the process. A one-to-one correspondence between the set of operating condition regimes and the set of supervising algorithms (i.e., supervisors) exist. The process (or each subprocess) is governed by a supervisor that belongs to the set of all supervisors designed for the complete operating regime. Decision algorithms can then be used by the coordinator to select the supervisory strategy expected to be the current optimal choice.

III. A Hybrid Supervisory Control System

As seen from Section II, the reconfigurable supervisory control system is constituted from a mix of continuous/discrete time processes and discrete-event processes. Fig. 2 shows the resulting hybrid supervisory control system. To simplify the discussion, the resulting system is shown as a hybrid system of two levels. Specifically, the low-level controllers represent the control algorithms that reside in the executors, and the high-level controllers represent the decision-making (e.g., supervisory or coordinating) algorithms that implement the coordinator or supervisors. While low-level controllers compute control actions based on the current process dynamics, the high level decisions are based on current observations

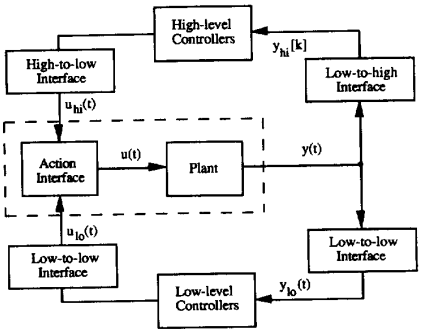


Fig. 2. Resulting hybrid control system.

in the supervised world. To appropriately inform each level of the plant behavior and to take the pertinent actions on it, inter-level interfaces are provided. In particular, the low-to-low interface at the lower level transforms the plant output $y(t)$ into the low-level plant representation $y_{\mathrm{lo}}(t)$ used by the low-level controllers to act on the plant through the low-to-low interface at the upper level by means of the low level control input $u_{\mathrm{lo}}(t)$. Similarly, the low-to-high interface transforms the plant output $y(t)$ into a more abstract representation $y_{\mathrm{hi}}[K]$. This signal is interpreted by the high level controllers to generate the high level control input $u_{\mathrm{hi}}(t)$ that emanates from the high-to-low interface shown in Fig. 2. At this point, the action interface takes both of these control signals, i.e., $u_{\mathrm{lo}}(t)$ and $u_{\mathrm{hi}}(t)$, to produce the plant input control action $u(t)$ which directly acts on the plant. Therefore, a series of mappings are defined in this hybrid system.

To achieve proper system response, specified performance criteria must be satisfied at each level. That is, control systems analysis and synthesis techniques must be provided to assure appropriate operation of each level. For the low-level case, control synthesis tools have been extensively reported (e.g., [18]). For the high-level case, the paradigm of reconfigurable DES has been investigated in [6] and [7]. Suitable response of the overall control system, however, cannot be guaranteed based only on the performance of the individual levels. It requires the combined information from all levels. Specifically, the control input $u(t)$ must ensure that the plant output $y(t)$ evolves as desired to guarantee proper overall execution of the hybrid system. This signal $u(t)$ is equal to

$$u(t) = T(u_{\mathrm{lo}}(t), u_{\mathrm{hi}}(t)) \tag{1}$$

where $u_{\mathrm{lo}}(t)$ and $u_{\mathrm{hi}}(t)$ denotes the high- and low-level control inputs, respectively, and $T(\cdot)$ denotes the mapping corresponding to the action interface as shown in Fig. 2. Because the (time-dependent) supervisory signal $u_{\mathrm{hi}}(t)$ uniquely determines the action that higher level controllers exert over the plant behavior, analytical studies must be conducted first to assess, for example, the effect of $u_{\mathrm{hi}}(t)$ over the transient behavior of the plant and then to define allowable ranges of supervising actions. This task is application dependent and general methodologies are not currently available.

IV. Framework for Discrete-Event Systems

As indicated in Sections II and III, different model paradigms are utilized to describe processes at different levels. In particular, processes with either time-driven or event-driven dynamics are present in the proposed hybrid configuration. To describe and implement time-driven processes, such as low level controllers, standard differential or difference equations are utilized. On the hand, to describe and implement event-driven processes, such as high level controllers or supervisors, a discrete-event framework is formulated as discussed in this section.

A. The Discrete-Event Model

Based on [21], the following discrete-event model is introduced in [6] to represent the DES

$$M := (\Gamma, \Psi) \tag{2}$$

where M is called the discrete-event mechanism and Γ and Ψ are called the static and dynamic components of M, respectively. The static component Γ is defined over the following three-tuple

$$\Gamma := (\mathcal{V}, \Sigma, \mathcal{P}) \tag{3}$$

where $\mathcal{V}, \Sigma$, and $\mathcal{P}$ are finite sets. Specifically, $\mathcal{V}$ is the nonempty set of internal state variables used to represent internal system dynamics. The set Σ is the event space or input alphabet representing the set of events defined for the process M with an event being a qualitative change occurring in the system. Finally, the set $\mathcal{P}$ is the collection of state predicates defined on the state space $\mathcal{X}$. Each internal variable $x_i \in \mathcal{V}$, also called a state variable, has an associated range, $\text{range}\,(x_i)$, which can be finite or infinite. It is important to indicate here that it is the designers' responsibility to formulate conditions within the range of allowable variables. On the other hand, the dynamic component Ψ of M is defined over the following four-tuple

$$\Psi := (I, d, f, O) \tag{4}$$

where $I(\cdot)$ is called the input function, $O(\cdot)$ is the output function, $d(\cdot)$ is the possible events function, and $f(\cdot)$ is a partial function called the state transition function. The input and output functions $I(\cdot)$ and $O(\cdot)$ are, in general, mappings of the form

$$I: \underset{i}{\times} \mathcal{X}_i \times 2^{\Sigma} \to \mathcal{X} \times \Sigma$$
$$O: \mathcal{X} \times \Sigma \to \mathcal{X}^O \times 2^{\Sigma}$$

where $\mathcal{X}_i$ is the state space of the ith mechanism attached to M, $\mathcal{X}^O$ is the output state space of M, and 2^{Σ} is the power set of the alphabet Σ. The possible events function $d: \mathcal{X} \to 2^{\Sigma}$ is a set-valued function that specifies the set of possible events defined at each state; i.e., $d(x) := \{\sigma \in \Sigma: D \cdot \sigma \cdot x = 1\}$ where the predicate $D \cdot \sigma: \mathcal{X} \to \{0, 1\}$ represents the enabling condition for the event σ. Finally, the state transition function $f: \mathcal{X} \times \Sigma \to 2^{\mathcal{X}}$ is a partial function that defines how changes in states occur in a given DES due to incoming events. The dynamic of M is then modeled by f as follows

$$x[k+1] = f(x[k], \sigma[k+1]); \quad \text{with} \quad P_o \cdot x[0] = 1 \tag{5}$$

where $x[k] \in \mathcal{X}$ is the state after the kth event, $\sigma[k]$ is the kth event, and $x[0]$ is an initial state satisfying a given initial predicate P_o. In this paper, any discrete-event mechanism M is modeled as a deterministic machine, i.e., $f(x[k], \sigma[k+1])$ is a singleton for every $x[k]$ and $\sigma[k+1]$. A mechanism M can be interpreted as a device that starts within an initial state region $\mathcal{X}_o$ (i.e., $x[0] \in \mathcal{X}_o$) and executes state transitions as a response (or cause) of a sequence of events. Notice that $\sigma \in d(x[k])$ implies that $f(x[k], \sigma)$ is defined. Therefore, the next event $\sigma[k+1]$ in (5) must satisfy the following equation

$$\sigma[k+1] \in d(x[k]) \qquad \forall k. \tag{6}$$

It is customary to observe the rule that no two transitions fire simultaneously.

B. Control Approach

To control DES, certain events in the system are enabled or disabled by choice of control inputs thus governing, whenever possible, transitions among states with control specifications given in terms of predicates. The design problem is to formulate a supervisor that assigns control patterns (to be defined below) at each state so that a specified predicate can be satisfied. Based on their controllability, the set Σ is partitioned into two disjoint subsets, Σ_c and Σ_u. While the subset Σ_c represents the set of controllable events, which can be prevented from occurring, Σ_u represents the set of uncontrollable events that cannot be prevented from occurring with $\Sigma_u = \Sigma - \Sigma_c$. Notice that although disabled events are certainly prevented from occurring, enabled events are not necessarily forced to occur. For example, controllable events may represent commands that control system actuators and uncontrollable events may represent system changes detected by sensors. Likewise, controllable events can be attained to the activation or restraint of specified controllers or supervisors. In this case, the enabling of certain controllable events could be interpreted as to select particular controllers. Dynamic enabling of events permits on-line reconfiguration of the control strategy as discussed in the in-plant experiments in Section VII. Reconfiguration of supervisors could be similarly implemented for more complex applications.

The control law for a discrete-event process is realized by a control pattern for M. A control pattern for M is defined as a predicate $U \cdot \sigma$ that specifies for each controllable event σ if σ is allowed to occur at a given state x. Specifically, σ is enabled by $U \cdot \sigma$ at x if $U \cdot \sigma \cdot x = 1$; it is disabled, otherwise. Consequently, the control input $u[k] \in 2^{\Sigma_C}$ to M is defined as the set of controllable events allowed by the supervisor to occur (i.e., not disabled) at the instant k on M. This can be expressed as follows

$$u[k] := \{\sigma \in \Sigma_C: \sigma \in d(x[k]), U \cdot \sigma \cdot x = 1\}. \tag{7}$$

As indicated earlier in Section II, the supervisor has authority only over controllable events. On the other hand, the disturbances acting on a DES, denoted by $w[k]$, is the set of

uncontrollable events that may occur at a given state $x[k]$ defined as follows

$$w[k] := \{\sigma \in \Sigma_u : \sigma \in d(x[k])\}. \tag{8}$$

The next possible event $\sigma[k+1]$, given by (6), now becomes

$$\sigma[k+1] \in u[k] \cup w[k]. \tag{9}$$

The controlled discrete-event process M_c, which results from the original machine M under the context of Σ_c, emphasizes the event constraints and control mechanism explicitly indicated by (9). M_c can be interpreted as a version of M that admits external control [21]. The resulting control system, that is, the controlled DES process M_c with a supervisor generating $u[k]$, is called the supervised system. We will restrict our attention to the case where the plant is driven by events and generates outputs that the supervisor monitors. Thus, a supervisor M_s executes a sequence of state transitions in response to environmental state changes and, possibly, incoming events and generates a sequence $u[k]$ to control another discrete-event process M_c as shown in Fig. 3. Following [23] and Fig. 3, two types of channels can be identified: condition channels, which transport piecewise constant signals representing state conditions; and event channels, which transport discrete (delta) signals representing system events. The plant M_c influences the state transitions of M_s by means of observed plant state and, possibly, incoming events while M_c is driven by a sequence of disturbances $\{w[k]\}$ and control inputs $\{u[k]\}$ determined by the consecutive states $x[k]$ of M_s. This class of supervision will be called event/state feedback control. As in the case of event/state feedback control, the input and output mappings $I(\cdot)$ and $O(\cdot)$ defined in Section IV-A can be made dependent on the functional characteristics that the given DES model describes. Thus, $I(\cdot)$ and $O(\cdot)$ are here redefined for each logical mechanism as

$$\begin{aligned} \text{Plant } M_c: \quad & I^P: 2^\Sigma \to \Sigma \\ & O^P: \mathcal{X}^P \times \Sigma \to \mathcal{X}^O \times \Sigma_u \\ \text{Supervisor } M_s: \quad & I^S: \underset{i}{\times}\mathcal{X}_i \times \Sigma \to \mathcal{X}^S \times \Sigma \\ & O^S: \mathcal{X}^S \to 2^{\Sigma_c} \end{aligned}$$

where $\mathcal{X}^P$ and $\mathcal{X}^S$ represent the state-space of a plant M_c and a supervisor M_s, respectively, $\mathcal{X}_i$ represents the state-space of the ith mechanism M_i attached to a given supervisor, and Σ_c and Σ_u are the controllable and uncontrollable subsets of Σ. While the space X^O is the output space of the plant and usually equal to $\mathcal{X}^P, \mathcal{X}^S$ is formed in general as

$$\mathcal{X}^S = \underset{i}{\times}\mathcal{X}_i \times \mathcal{X}^S_{\text{int}}$$

where $\mathcal{X}^S_{\text{int}}$ is defined as the internal state space of the supervisor. Specifically, $\mathcal{X}^S_{\text{int}}$ is introduced to allow a given supervisor to record process history information. This auxiliary information may be required when enforcing supervisory constraints derived from dynamic specifications. Supervisors observing an internal space are called dynamic supervisors; otherwise, they are called static or "memoryless" supervisors

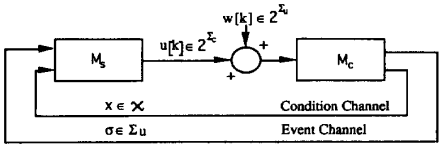


Fig. 3. discrete-event model for closed-loop supervision.

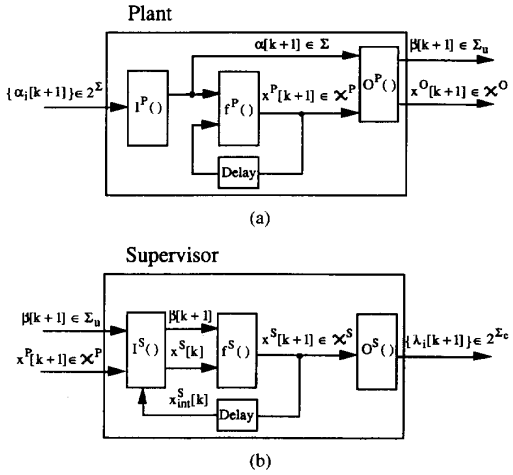


Fig. 4. (a) Block diagram for a plant. (b) Block diagram for a supervisor.

[13], [15]. In view of these mappings, Fig. 4(a) and (b) illustrate the input–output characterization for each mechanism, respectively.

V. Statement of the Control Problem and Solution

To demonstrate and evaluate hybrid reconfigurable control systems, the feedwater system at the experimental breeder reactor (EBR-II) was selected for conducting the control experiments. The EBR-II power plant has as its primary system a sodium-cooled pool type fast reactor rated at 62.5 MW thermal. An intermediate sodium loop removes heat from the primary system sodium and transfers the energy to a conventional steam electric power producing a cycle that generates 20 MWe. The major sequence of components in the steam plant consists of a condensate pump, closed feedwater heater, deaerating heater, feedwater pump, two high pressure closed feedwater heaters, steam drum, evaporators, superheaters, and a turbine generator system. A simplified representation of the EBR-II feedwater deaerator system is shown in Fig. 5 where $l(t)$ and $p(t)$ denote the water level and pressure in the vessel, respectively; $q_{i1}(t), q_{i2}(t)$ and $q_o(t)$ denote the steam, condensate and feedwater flow, respectively; and $T_1(t)$ and $T_2(t)$ denote the condensate and the feedwater temperatures, respectively. The EBR-II deaerator is vertically oriented. Nominal liquid level of above 170 inches represents approximately a six-minute supply of feedwater at the full power condition.

The condensate flow is regulated by a coarse control and a fine control valves, operating in a parallel configuration as shown in Fig. 5. There are sensors that can measure any of the above process variables. To control the process variables

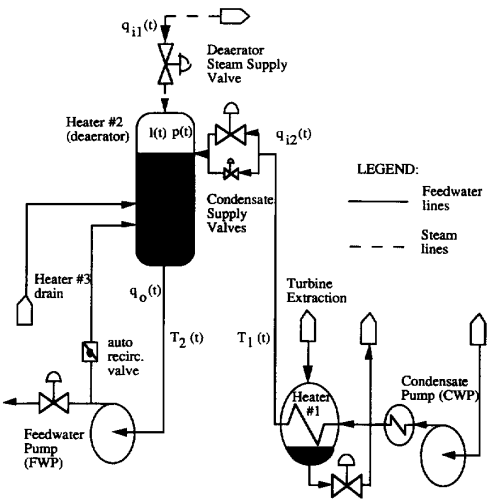


Fig. 5. The feedwater-deaerator system.

[namely, $l(t)$ and $p(t)$], $q_{i1}(t)$ and $q_{i2}(t)$ are regulated by the steam and condensate flow valves, respectively, as shown in Fig. 5. Let $u_{\rm st}(t)$ and $u_{\rm co}(t)$ denote the steam and condensate position commands to the steam and the condensate valves, respectively. In the current installation, $u_{\rm st}(t)$ and $u_{\rm co}(t)$ result from single loop PID control algorithms based on deaerator pressure and level, respectively. As shown in Fig. 5, there are actually two parallel flow paths, with associated flow control valves, for regulating condensate flow $q_{i2}(t)$ at EBR-II. Steam header pressure, which maintains the deaerating steam flow $q_{i1}(t)$, is regulated to 150 psig by manipulating a valve in an extraction line from the main steam header or from the plant auxiliary steam supply system.

To provide the required NPSH to the feedwater pumps, the deaerator is elevated approximately 25 feet above the feedwater pump suction. Specifically, there are two feedwater pumps, only one of which is used during normal operations. Whereas pressure effects in the deaerator are acoustically transmitted and are almost immediately propagated to the feedwater pump inlet, changes of internal energy at the pump inlet are delayed by the transport time which is dependent on the velocity of water in the pipeline. At the full power condition, it takes about 12 seconds for water leaving the deaerator to arrive at the inlet to feedwater pump 1 and approximately 24 seconds for water to arrive at feedwater pump 2. Due to the transport delay from the deaerator to the feedwater pumps, there can be a significant reduction in NPSH if there is a rapid reduction in deaerator pressure. This potential reduction in NPSH during transients motivated the design of the fault-accommodating supervisory system. In particular, due to a failure in the steam head assemblage (e.g., steam controller and steam valve), there could be an uncontrolled change in the steam flow $q_{i1}(t)$. Because the condensate valve position $u_{\rm co}(t)$ [and, consequently, $q_{i1}(t)$] is based on the current deaerator level $l(t)$, the deaerator pressure $p(t)$ is left unregulated. Therefore, any pressure transient could occur within the deaerator, which may cause a hazard situation. For example, a rapid reduction in deaerator pressure may cause cavitation of the feedwater pump or pressure increases may cause large pressure fluctuations in the deaerator, which is also potentially dangerous. To avoid such pressure fluctuations, the proposed solution was to provide the system with an alternative condensate controller that regulates condensate flow to control the pressure in the deaerator. To keep the implementation simple and to maintain consistency with the existing controller, this alternative condensate controller has a single loop PID control structure with the deaerator pressure as input. Based on the current deaerator operating condition, a supervisor included in the control system selects a given condensate controller to regulate $q_{i2}(t)$. This results in a control system that has a hybrid structure as discussed next.

VI. The Implemented Reconfigurable Hybrid System

The reconfigurable hybrid system, as implemented in EBR-II, is a hierarchy of two layers as seen in Fig. 6. At the lower layer (called the execution layer), the existing controllers regulate the physical process in the traditional manner. These controllers continuously monitor the plant output and generate the real-time control inputs to the plant, namely, condensate flow control $u_{\rm co}(t)$ and steam flow control $u_{\rm st}(t)$. Specifically, there are three low level controllers, namely, $C_{\rm lo_1}, C_{\rm lo_2}$, and $C_{\rm lo_3}$. The first two controllers, $C_{\rm lo_1}$ and $C_{\rm lo_2}$, regulate the condensate flow by manipulating the condensate inlet valves (see Fig. 5) based on the deaerator level or pressure, respectively. The third controller, $C_{\rm lo_3}$, regulates the steam flow by manipulating the steam inlet valve based on the deaerator pressure. Let $P_{\rm lo}$ denote the model of the physical plant P at the execution level. $P_{\rm lo}$ can be represented by differential equations with actuator inputs and sensor outputs evolving in real time. That is

$$\dot{x}^P_{\rm lo}(t) = f(x^P_{\rm lo}(t), u_{\rm lo}(t)) \tag{10}$$

$$y^P_{\rm lo}(t) = g(x^P_{\rm lo}(t), u_{\rm lo}(t)) \tag{11}$$

where the subscript "lo" and the superscript "P" are used to identify variables defined at the lower layer and belonging to the plant, respectively. The state vector $x^P_{\rm lo}(t) \in \mathcal{R}^4$, the control vector $u_{\rm lo}(t) \in \mathcal{R}^2$, and the output vector $y^P_{\rm lo}(t) \in \mathcal{R}^4$ of the continuously varying plant are defined as

$$\begin{aligned} x^P_{\rm lo}(t) &= [l(t)p(t)T_1(t)T_2(t)]^T \\ u_{\rm lo}(t) &= [u_{\rm co}(t)u_{\rm st}(t)]^T \\ y^P_{\rm lo}(t) &= [l(t)p(t)T_1(t)T_2(t)]^T \end{aligned} \tag{12}$$

which are described in Section V. The explicit structure of the plant dynamics can be found in [8] and is not given here. As seen from Fig. 6, $x^P_{\rm lo}(t)$ is branched in two with one branch going to the execution level while the other going to the supervisory level. Controllers at the lower level regulate the physical process by directing continuous-time commands in response to the continuous sensor signals. These controllers use a modified version of $x^P_{\rm lo}(t)$ to compute their respective control actions. Let $x^P_{\rm lolo}(t)$ denote the information low-to-low transmitted to the low level controllers $C_{\rm lo_i}, i = 1, 2, 3$, via the low-to-low interface $\phi_{\rm lolo}$. The low-to-low interface

ϕ_{lolo} at the lowest level may or may not utilize all the sensory information. In this particular case, only level $l(t)$ and pressure $p(t)$ were used. These signal were filtered using analog hardware which is a part of the sensory system and then fed to the controllers of the proportional-integral type [18]. Formally

$$\phi_{\text{lolo}}: \mathcal{X}_{\text{lo}}^P \rightarrow \mathcal{X}_{\text{lolo}}^P \tag{13}$$

where $\mathcal{X}_{\text{lo}}^P$ is the state space of P_{lo} and $\mathcal{X}_{\text{lolo}}^P$ is a transformed low-to-low state space of P_{lo} equal to

$$x_{\text{lolo}}^P(t) = \phi_{\text{lolo}}(x_{\text{lo}}^P(t)) = [l(t)p(t)]^T. \tag{14}$$

At the upper layer, there exists a supervisor that regulates the physical process via symbolic commands in response to the symbolic inputs $w_{\text{hi}}[k]$ generated by the low-to-high interface $\phi_{\text{lohi}}(\cdot)$ as seen in Fig. 6. In general, an event is controllable if it is a command and uncontrollable if it is a plant response. Because more aggregated information is used by the supervisor at the supervisory level, the frequency of decisions decreases with increase in the level of the hierarchy because a more abstract representation of the physical process is utilized. The higher abstraction P_{hi} of P at the supervision layer is essentially a simplified model of the plant model P_{lo}. The mapping $\phi_{\text{lohi}}(\cdot)$ transforms continuous state plant variables into a set of discrete events as

$$\phi_{\text{lohi}}: \mathcal{X}_{\text{lo}}^P \rightarrow 2^{\Sigma_u^P} \tag{15}$$

where $\mathcal{X}_{\text{lo}}^P$ is the state space of P_{lo} and Σ_u^P is the set of uncontrollable events defined for P_{hi}. Therefore, $w_{\text{hi}}[k]$ is the disturbance to P_{hi} as defined in Section IV with $w_{\text{hi}}[k] \in 2^{\Sigma_u^P}$. Specifically, Σ_u^P is given by

$$\Sigma_u^P = \Sigma_u^{\text{temp}} \cup \Sigma_u^{\text{npsh}} \cup \Sigma_u^{\text{incp}} \cup \Sigma_u^{\text{decp}} \cup \Sigma_u^{\text{disys}} \cup \Sigma_u^{\text{hist}} \cup \Sigma_u^{\text{ctrl}} \tag{16}$$

where

$$\begin{aligned}
\Sigma_u^{\text{temp}} &= \{\sigma_{\text{good}}^{\text{temp}}, \sigma_{\text{bad}}^{\text{temp}}, \sigma_{\text{und}}^{\text{temp}}\},\\
\Sigma_u^{\text{npsh}} &= \{\sigma_{\text{good}}^{\text{npsh}}, \sigma_{\text{bad}}^{\text{npsh}}, \sigma_{\text{und}}^{\text{npsh}}\},\\
\Sigma_u^{\text{incp}} &= \{\sigma_{\text{good}}^{\text{incp}}, \sigma_{\text{bad}}^{\text{incp}}, \sigma_{\text{und}}^{\text{incp}}\},\\
\Sigma_u^{\text{decp}} &= \{\sigma_{\text{good}}^{\text{decp}}, \sigma_{\text{bad}}^{\text{decp}}, \sigma_{\text{und}}^{\text{decp}}\},\\
\Sigma_u^{\text{disys}} &= \{\sigma_{\text{good}}^{\text{disys}}, \sigma_{\text{bad}}^{\text{disys}}, \sigma_{\text{und}}^{\text{disys}}\},\\
\Sigma_u^{\text{hist}} &= \{\sigma_{\text{good}}^{\text{hist}}, \sigma_{\text{bad}}^{\text{hist}}, \sigma_{\text{und}}^{\text{hist}}\},\\
\Sigma_u^{\text{ctrl}} &= \{\sigma_{\text{co}_1}, \sigma_{\text{co}_2}, \sigma_{\text{none}}\}
\end{aligned}$$

with the symbols as defined below:

$\sigma_{\text{good}}^{\text{temp}}$, $\sigma_{\text{bad}}^{\text{temp}}$, $\sigma_{\text{und}}^{\text{temp}}$	deaerator water/steam temperature (temp) is good, bad or undesirable
$\sigma_{\text{good}}^{\text{npsh}}$, $\sigma_{\text{bad}}^{\text{npsh}}$, $\sigma_{\text{und}}^{\text{npsh}}$	net positive suction head (npsh) at the feedpump is good, bad or undesirable
$\sigma_{\text{good}}^{\text{incp}}$, $\sigma_{\text{bad}}^{\text{incp}}$, $\sigma_{\text{und}}^{\text{incp}}$	deaerator water/steam pressure increase (incp) rate is good, bad or undesirable

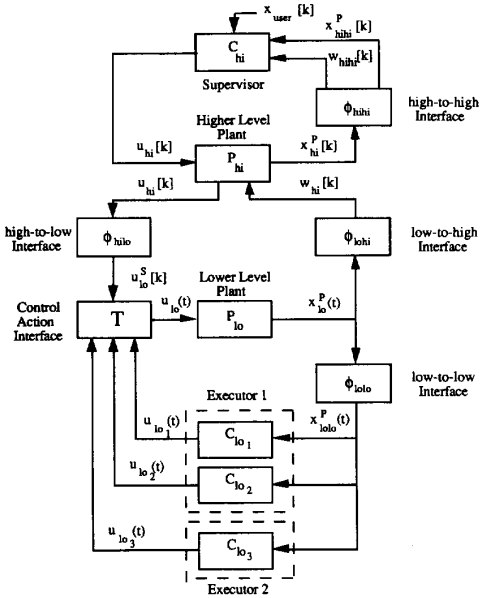


Fig. 6. The evaluated hybrid supervisory control system.

$\sigma_{\text{good}}^{\text{decp}}$, $\sigma_{\text{bad}}^{\text{decp}}$, $\sigma_{\text{und}}^{\text{decp}}$	deaerator water/steam pressure decrease (decp) rate is good, bad or undesirable
$\sigma_{\text{good}}^{\text{disys}}$, $\sigma_{\text{bad}}^{\text{disys}}$, $\sigma_{\text{und}}^{\text{disys}}$	overall diagnosed system (disys) status is good, bad or undesirable
$\sigma_{\text{good}}^{\text{hist}}$, $\sigma_{\text{bad}}^{\text{hist}}$, $\sigma_{\text{und}}^{\text{hist}}$	deaerator operating history (hist) is acceptable, inadequate or undefined.

On the other hand, $\sigma_{\text{co}_1}, \sigma_{\text{co}_2}$, and σ_{none} indicate that the level-based condensate controller, the pressure-based condensate controller, or no controller is being used to manipulate the condensate valve, respectively. Since the above set of events is chosen by the designer, it is critical to ensure that this set adequately informs the supervisor of the pertinent events generated by the condensate system in Fig. 5. In particular, the uncontrollable events are generated after applying a set of different performance algorithms. Some of these algorithms involve elaborate logic which are described in detail in [8].

As indicated in Sections II and III, processes at the higher layers are modeled in the discrete-event setting. Two discrete-event mechanisms can thus be identified here, namely, a discrete-event representation M_c of the plant P, i.e., P_{hi}, as well as a discrete-event representation of a high level controller (or supervisor M_s) denoted by C_{hi}. With respect to P_{hi}, the following state transition and output equations describe the dynamics of P_{hi}

$$x_{\text{hi}}^P[k+1] = f^P(x_{\text{hi}}^P[k], \sigma[k+1]) \tag{17}$$

$$y_{\text{hi}}^P[k] = O^P(x_{\text{hi}}^P[k], \sigma[k]) \tag{18}$$

where the subscript "hi" is used to identify variables defined at the higher or supervision layer. The functions $f^P(\cdot)$ and $O^P(\cdot)$ are the state transition and output functions of a M_c as defined in Section IV while the seven-dimensional discrete state vector $x_{\text{hi}}^P[k] \in \{-1,0,1\}^6 \times \{-3,-2,-1,0,1,2,3\}^1, \sigma[k] \in \Sigma^P$, and $y_{\text{hi}}^P[k] \in \{-1,0,1\}^7$ are the state, input and output vectors

of the discrete model M_c of P defined as

$$x_{\text{hi}}^P[k] = [x_1^P[k]x_2^P[k]x_3^P[k]x_4^P[k]x_5^P[k]x_6^P[k]x_7^P[k]]^T \quad (19)$$

$$\sigma[k] = I^P(u_{\text{hi}}[k] \cup w_{\text{hi}}[k]) \quad (20)$$

$$y_{\text{hi}}^P[k] = [x_1^P[k]x_2^P[k]x_3^P[k]x_4^P[k]x_5^P[k]x_6^P[k]x_7^P[k]]^T \quad (21)$$

with $I^P(\cdot)$ being the input function of P_{hi} as defined in Section II and

$x_1^P[k]$	temperature status of the deaerator system.
$x_2^P[k]$	net positive suction head status at the feedwater pump.
$x_3^P[k]$	pressure increase rate status in the deaerator.
$x_4^P[k]$	pressure decrease rate status in the deaerator.
$x_5^P[k]$	overall diagnosed system status.
$x_6^P[k]$	pressure and level history status in the deaerator.
$x_7^P[k]$	commanded/operating condensate controller.
$u_{\text{hi}}[k] \in \Sigma_c^P$	control input to P_{hi} generated by C_{hi}.
Σ_c^P	set of controllable events defined for P_{hi}.
$\Sigma^P = \Sigma_c^P \cup \Sigma_u^P$	set of events defined for P_{hi}.

Notice that $(\forall i = 1, \cdots, 6)\,\text{type}\,(x_i^P) = \{-1, 0, 1\}$ and $\text{type}\,(x_7^P) = \{-3, -2, -1, 0, 1, 2, 3\}$. In particular, for any $i \in \{1, \cdots, 6\}$, a value of -1, 0, and 1 in x_i^P signifies that the corresponding plant status is undefined, acceptable, or inadequate, respectively. For x_7^P, a value of -1, 0, 1 indicates that none, level-based, or pressure-based condensate controller, respectively, is currently manipulating the condensate flow valve. A value of -3 (respectively, -2) indicates that level condensate controller has been commanded by the supervisor while pressure (respectively, level) based controller was in operation; the interpretation for x_7^P when equal to three (respectively, two) can be deduced by negation of the above. Notice that the high level description P_{hi} of P_{lo} is freely chosen by the designer. Therefore, the modeling criteria should be carefully examined to assure that P_{hi} is an adequate representation of the physical process at the supervisor level. In particular, the plant vector given by (19) was considered to be the minimal information needed to assert the pertinent status of the system shown in Fig. 5. The set Σ_c^P of controllable events defined for P_{hi} is actually determined by the high level controller governing P_{hi}. Thus, Σ_c^P relates the choices of commands made available to the high level controller. In this implementation, Σ_c^P was chosen to be

$$\Sigma_c^P = \{\sigma_{1\bar{2}}, \overline{\sigma}_{1\bar{2}}\} \quad (22)$$

where $\sigma_{1\bar{2}}$ (respectively, $\overline{\sigma}_{1\bar{2}}$) indicates to enable the level-based (respectively, pressure-based) condensate controller and disable pressure-based (respectively, level-based) condensate controller. The transition function $f^P(\cdot)$ of P_{hi} given in (17) is shown in Fig. 7(a) for one specific element of the state vector $x_i^P[k], i = 1, 2, \cdots, 6$ and $\# \in$ {temp, npsh, incp, decp, disys, hist}. Fig. 7(b) illustrates the transition function for the remaining state $x_7^P[k]$ where $x_7^P[k] = -1$ has been omitted for clarity. Also, the $[k+1]$ subscript on the σ's is not shown. An arrow indicates the transition from an initial state $x_i^P[k]$ to a final state $x_i^P[k+1]$ due to the occurrence of an event $\sigma[k+1]$ at $k+1$. Notice that when $x_7^P[k]$ is equal to ± 2 or ± 3, the high level controller has "proposed" a given controller to act on the condensate valve. It is possible, however, that the corresponding environmental response indicates no enforcement of such controller (to ensure operational and safety constraints, as seen later) but another low level controller is been used instead. In this case, the supervisor has failed to enforce a given controller. This event is accounted by notifying the coordinator with a special message. This is an unmodeled event, however, and therefore does not appear in any model. From (18) and (21), it follows that the output function $O^p(\cdot)$ of P_{hi} is simply the current plant state; i.e., $O^P(x_{\text{hi}}^P[k], \sigma[k]) = x_{\text{hi}}^P[k]$. This state vector is then modified before being processed by the supervisor as seen from Fig. 6. Specifically, $w_{\text{hihi}}[k]$ and $x_{\text{hihi}}^P[k]$ is the high-to-high information transmitted to the supervisor or high level controller C_{hi} via the mapping $\phi_{\text{hihi}}(\cdot)$. In particular, $\phi_{\text{hihi}}(\cdot)$ transforms high level plant states into a given discrete-event and a plant state as

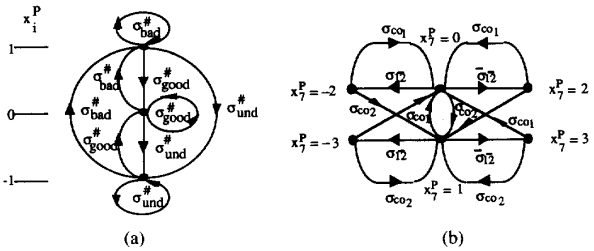


Fig. 7. Transition function of the high-level plant model.

$$\phi_{\text{hihi}}: \mathcal{X}_{\text{hi}}^P \rightarrow \mathcal{X}_{\text{hihi}}^P \times \Sigma_u^S \quad (23)$$

where $\mathcal{X}_{\text{hi}}^P$ is the state space of P_{hi}, $\mathcal{X}_{\text{hihi}}^P$ is a transformed high-to-high state space of P_{hi} and Σ_u^S is the set of uncontrollable events defined for C_{hi}. In particular, $x_{\text{hihi}}^P[k] \in \{-1, 0, 1\}$ where $\{-1, 0, 1\}$ has the same interpretation that $x_7^P[k]$ as given in (19). On the other hand, $w_{\text{hihi}}[k] \in \Sigma_u^S$ with

$$\Sigma_u^S = \{\sigma_{\text{bad}}, \sigma_{\text{good}}, \sigma_{\text{und}}\} \quad (24)$$

where $\sigma_{\text{bad}}, \sigma_{\text{good}}$, and σ_{und} indicates that the current operation of the enabled condensate flow controller is inadequate, acceptable, or undefined. To generate $w_{\text{hihi}}[k]$ from $x_{\text{hi}}^P[k]$, the following logic is used in ϕ_{hihi}

$$w_{\text{hihi}}[k] = \begin{cases} \sigma_{\text{bad}} & \text{if } x_1^P[k] = 1 \\ \sigma_{\text{bad}} & \text{if } (x_2^P[k] = 1) \vee (x_3^P[k] = 1) \\ & \quad \vee (x_4^P[k] = 1) \\ \sigma_{\text{good}} & \text{if } (x_2^P[k] = 0) \vee (x_3^P[k] = 0) \\ & \quad \vee (x_4^P[k] = 0) \\ \sigma_{\text{bad}} & \text{if } x_5^P[k] = 1 \\ \sigma_{\text{und}} & \text{if } x_6^P[k] = -1 \\ \sigma_{\text{good}} & \text{if } x_6^P[k] = 0 \\ \sigma_{\text{bad}} & \text{if } x_6^P[k] = 1. \end{cases} \quad (25)$$

As given in Fig. 6, the high level controller C_{hi} resides at the top of the hierarchy. The following equations describe the input/output characterization of C_{hi}

$$x^S_{\text{hi}}[k+1] = f^S(x^S_{\text{hi}}[k], w_{\text{hihi}}[k+1]) \tag{27}$$

$$u_{\text{hi}}[k] = O^S(x^S_{\text{hi}}[k]) \tag{28}$$

where the superscript "S" is used to identify variables defined for the supervisor C_{hi}. The functions $f^S(\cdot)$ and $O^S(\cdot)$ are the state transition and output functions of C_{hi} as defined in Section IV. Specifically, $f^S(\cdot)$ is a mapping $f^S: \mathcal{X}^S_{\text{hi}} \times \Sigma^S_u \to \mathcal{X}^S_{\text{hi}}$ where Σ^S_u is the set of uncontrollable events defined for C_{hi} (or the supervisor S) as given in (24) and $\mathcal{X}^S_{\text{hi}}$ is the state space of C_{hi} generated as follows

$$\mathcal{X}^S_{\text{hi}} = \mathcal{X}_{\text{user}} \times \mathcal{X}^P_{\text{hihi}} \times \mathcal{X}^S_{\text{int}} \tag{29}$$

with $\mathcal{X}_{\text{user}}$ denoting the "user" commands space, $\mathcal{X}^P_{\text{hihi}}$ the transformed high-to-high state space of P_{hi} as given earlier, and $\mathcal{X}^S_{\text{int}}$ the internal state space of C_{hi}. As indicated in Section IV–B and Fig. 4(b), this internal state space provides C_{hi} with "memory," i.e., C_{hi} is a dynamic state feedback supervisor [13]. In particular, $\mathcal{X}^S_{\text{int}}$ is given as $\mathcal{X}^S_{\text{int}} = \{0, 1/N, 2/N, \cdots, 1\}^2$ where N is chosen to be an integer. On the other hand, $\mathcal{X}_{\text{user}}$ is defined as $\mathcal{X}_{\text{user}} = \{0, 1\}$. Thus, $x^S_{\text{hi}}[k]$ is equal to

$$x^S_{\text{hi}}[k] = [x^S_1[k] x^S_2[k] x^S_3[k] x^S_4[k]]^T \in \mathcal{X}^S_{\text{hi}} \tag{30}$$

where $x^S_1[k]\,(:= x_{\text{user}}[k])$ represents the user command with zero or one indicating that the user prohibits or authorizes control reconfiguration, respectively; $x^S_2[k]$ equals $x^P_{\text{hihi}}[k]$; $x^S_3[k]$ (resp. $x^s_4[k]$) is a confidence measurement for the level- (respectively, pressure-) based condensate controller in being the best current controller for operating the condensate flow valve. Based on this interpretation of the supervisor's states, N is chosen to be an even integer so that a middle point between the extreme values zero and one can exist and initialization of the supervisor with no preference in selecting a given controller can be possible. The task of the supervisor can now be defined as to control the events $\sigma_{1\overline{2}}$ and $\overline{\sigma}_{1\overline{2}}$ to ensure that a given desired predicate is satisfied. It is important to indicate that care must be taken in accurately modeling a specification. In this implementation, the desired predicate P_{des} is defined as

$$P_{\text{des}} = P_{\text{sp1}} \vee P_{\text{sp2}} \tag{31}$$

where $P_{\text{sp}j}: C_j x^S_{\text{hi}} \leq b_j, j = 1, 2$ with $C_1 = [0\ 0\ 1\ 0], C_2 = [0\ 0\ 0\ 1]$, and $b_j = 0.1; j = 1, 2$. The predicate given by (31) encodes the control objective specifying that, after a finite time period, the plant must use the controller (to operate the condensate flow) that generates statistically more "good" environment responses. Recall from (24) that these responses reflect the current performance of the control system, which is only "indirectly" influenced by control actions and usually affected by "unpredictable" plant disturbances and malfunctions. To devise a supervisor that guarantees that the desired predicate is (asymptotically) satisfied (with probability 1), $f^S(\cdot)$ and $O^S(\cdot)$ must be found. The methodology used for the supervisor synthesis is omitted here and can be found in [6]. The explicit structure of $f^S(\cdot)$ resulted as follows in (32) shown at the bottom of the page, with $f^S(\cdot)|_{x^S_3, x^S_4}$ indicating only the effect of $f^S(\cdot)$ over $x^S_3[\,]$ and $x^S_4[\,]$. For $x^S_i[\cdot], i = 1, 2, f^S(\cdot)$ has no effect. In (32), the $[k]$'s has been dropped from $x^S_1[\,], x^S_3[\,]$, and $w_{\text{hihi}}[\,]$ for a more compact notation. It follows from (32) that $(\forall k)\, x^S_3[k] + x^S_4[k] = 1$. On the other hand, the output function $O^S(\cdot)$ is a mapping $O^S: \mathcal{X}^S_{\text{hi}} \to \Sigma^S_c$ where $\mathcal{X}^S_{\text{hi}}$ is the state space of C_{hi} and Σ^S_c is the set of controllable events defined for C_{hi} equal to Σ^P_c as given in (22). To implement $O^S(\cdot)$, several design decisions must be resolved. For example, $O^S(\cdot)$ can be either a deterministic or a stochastic mapping. A deterministic mapping may be implemented as follows

$$u_{\text{hi}}[k] = \begin{cases} \sigma_{1\overline{2}} & \text{if } (x^S_1[k] = 0) \vee (x^S_3[k] \geq x^S_4[k]) \\ \overline{\sigma}_{1\overline{2}} & \text{otherwise} \end{cases} \tag{33}$$

where $\sigma_{1\overline{2}}$ and $\overline{\sigma}_{1\overline{2}}$ are defined in (22). On the other hand, there are situations where stochastic mappings are more convenient [15], [24] to devise supervisory algorithms. For the specific case of this implementation, it was expected that the sensed signal could be corrupted with notable levels of noise. Thus, a filtering capability was desired to be included in the dynamics of the supervisor. To this end, $O^S(\cdot)$ was realized as follows

$$u_{\text{hi}}[k] = \begin{cases} \sigma_{1\overline{2}} & \text{if } (x^s_i[k] = 0) \vee (0 \leq \mu[k] \leq x^S_3[k]) \\ \overline{\sigma}_{1\overline{2}} & \text{otherwise} \end{cases} \tag{34}$$

where $\mu[k]$ is a pseudo-random number uniformly distributed between zero and one. Equation (34) (and specifically, condition $0 \leq \mu[k] \leq x^S_3[k]$) can be explained as follows. The supervisor ith internal state $x^S_i[k], i = 3, 4$, can be interpreted as a degree of confidence on the ith condensate controller to

$$[x^S_3[k+1] x^S_4[k+1]]^T = f^S([x^S_1[k] x^S_2[k] x^S_3[k] x^S_4[k]]^T, w_{\text{hihi}}[k+1])|_{x^S_3, x^S_4}$$

$$= \begin{cases} [(x^S_3[k] + 1/N)(x^S_4[k] - 1/N)]^T & \text{if } ((x^S_1 = 1) \wedge (x^S_3 \neq 0, 1) \wedge (x^S_2 = 0) \wedge (w_{\text{hihi}} = \sigma_{\text{good}})) \\ & \quad \vee ((x^S_1 = 1) \wedge (x^S_3 \neq 0, 1) \wedge (x^S_2 = 1) \wedge (w_{\text{hihi}} = \sigma_{\text{bad}})) \\ [(x^S_3[k] - 1/N)(x^S_4[k] + 1/N)]^T & \text{if } ((x^S_1 = 1) \wedge (x^S_3 \neq 0, 1) \wedge (x^S_2 = 0) \wedge (w_{\text{hihi}} = \sigma_{\text{bad}})) \\ & \quad \vee ((x^S_1 = 1) \wedge (x^S_3 \neq 0, 1) \wedge (x^S_2 = 1) \wedge (w_{\text{hihi}} = \sigma_{\text{good}})) \\ [(x^S_3[k])(x^S_4[k])]^T & \text{if } ((x^S_1 = 1) \wedge (w_{\text{hihi}} = \sigma_{\text{und}})) \\ & \quad \vee ((x^S_1 = 1) \wedge (x^S_3 = 0 \vee 1) \wedge (w_{\text{hihi}} = \sigma_{\text{good}})) \\ [(1/N)(1 - 1/N)]^T & \text{if } ((x^S_1 = 1) \wedge (x^S_3 = 0) \wedge (w_{\text{hihi}} = \sigma_{\text{bad}})) \\ [(1 - 1/N)(1/N)]^T & \text{if } ((x^S_1 = 1) \wedge (x^S_3 = 1) \wedge (w_{\text{hihi}} = \sigma_{\text{bad}})) \\ [(0.5)(0.5)]^T & \text{if } (x^S_1 = 0) \end{cases} \tag{32}$$

be best for the current operating condition. Therefore, as the value of $x_i^S[k]$ is decreased due to unfavorable performance responses, the probability of choosing this ith controller consequently reduces as given by (34). Notice that the supervisor can thus exercise reconfiguration of the control scheme by enabling or disabling its controllable events. For example, if at time $k, u_{\text{hi}}[k] = \sigma_{1\bar{2}}$, and at time $k+1, u_{\text{hi}}[k+1] = \bar{\sigma}_{1\bar{2}}$ to accommodate new operating conditions, reconfiguration of the condensate flow control results by switching from a level-based to a pressure-based algorithm, as the assumed enabling and disabling of events suggests.

The signal $u_{\text{hi}}[k]$ is a symbolic command generated at the supervisor level. This command must be translated and transmitted to the execution level. Let $u_{\text{lo}}^S(t)$ denote the information high-to-low transmitted to the low level from the supervisory level via the interface ϕ_{hilo}. Specifically, the interface ϕ_{hilo} transforms controllable discrete events into an integral signal. In this implementation, ϕ_{hilo} is a discrete-event mechanism with the following dynamics

$$x_{\text{hilo}}[k+1] = f^{\text{hilo}}(x_{\text{hilo}}[k], u_{\text{hi}}[k+1]) \tag{35}$$

$$u_{\text{lo}}^S(t) = O^{\text{hilo}}(x_{\text{hilo}}[k]). \tag{36}$$

The functions $f^{\text{hilo}}(\cdot)$ and $O^{\text{hilo}}(\cdot)$ are the state transition and output functions of ϕ_{hilo}. Specifically, $f^{\text{hilo}}(\cdot)$ is a mapping $f^{\text{hilo}}: \mathcal{X}_{\text{hilo}} \times \Sigma_c^S \to \mathcal{X}_{\text{hilo}}$ where Σ_c^S is the set of controllable events given in (22) and $\mathcal{X}_{\text{hilo}}$ is the (internal) state space of ϕ_{hilo} given by

$$x_{\text{hilo}}[k] = [x_1^{\text{hilo}}[k] x_2^{\text{hilo}}[k]]^T \tag{37}$$

with $x_1^{\text{hilo}}[k]$ defined to be the currently advised condensate controller with zero or one indicating that the level-based or the pressure-based condensate controller is recommended for use, respectively; (notice that $\text{type}(x_1^{\text{hilo}}[k]) = \{0,1\}$) and $x_2^{\text{hilo}}[k]$ defined to be the (Limited) recorded history of advised control actions with $\text{type}(x_2^{\text{hilo}}[k]) = \{0,1,\cdots,M_{\max}\}$ and $M_{\max}$ a selected nonnegative integer defining the memory depth. The explicit structure of $f^{\text{hilo}}(\cdot)$ is shown in Fig. 8. On the other hand, the output function $O^{\text{hilo}}(\cdot)$ is given by

$$O^{\text{hilo}}(x_{\text{hilo}}[k]) = x_1^{\text{hilo}}[k]. \tag{38}$$

From (36) and (38), it is seen that $u_{\text{lo}}^S(t) = x_1^{\text{hilo}}[k]$. This signal is used by the "control action interface" to generate the low level control input $u_{\text{lo}}(t)$ to the physical plant as seen in Fig. 6. Let T denote the function implementing the control action interface. Then

$$u_{\text{lo}}(t) = T(u_{\text{lo}_1}(t), u_{\text{lo}_2}(t), u_{\text{lo}_3}(t), u_{\text{lo}}^S(t)) \tag{39}$$

where, as seen from Fig. 6, $u_{\text{lo}_1}(t), u_{\text{lo}_2}(t), u_{\text{lo}_3}(t)$ are the control signals generated by the level-based condensate, pressure-based condensate, and steam flow controllers, respectively, and $u_{\text{lo}}^S(t)$ is the information high-to-low transmitted to the low level by the supervisor. Recall from (12) that $u_{\text{lo}}(t) = [u_{\text{co}}(t) u_{\text{st}}(t)]^T$ with $u_{\text{co}}(t)$ and $u_{\text{st}}(t)$ denoting the condensate and steam position commands that control the aperture of the condensate and the steam valves, respectively. For the case of the steam valve, $(\forall t)\, u_{\text{st}}(t) = u_{\text{lo}_3}(t)$. This comes as a result that the executor for this valve has only one control algorithm

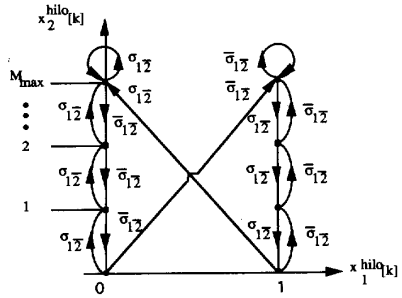


Fig. 8. Transition function of the high-to-low interface.

available at any time. On the other hand, the condensate valve executor has two control choices, i.e., $u_{\text{lo}_1}(t)$ and $u_{\text{lo}_2}(t)$. To select a given control algorithm, the following logic is employed by T

$$u_{\text{co}}(t) = \begin{cases} u_{\text{lo}_1}(t) & \text{if } (P_{\text{ERR}} \cdot x_{\text{lolo}}^P(t) = 1) \\ & \quad \vee (P_{\text{ENF}} \cdot x_{\text{lolo}}^P(t) = 0 \wedge u_{\text{lo}}^S(t) = 0) \\ u_{\text{lo}_2}(t) & \text{if } (P_{\text{ENF}} \cdot x_{\text{lolo}}^P(t) = 1) \\ & \quad \vee (P_{\text{ERR}} \cdot x_{\text{lolo}}^P(t) = 0 \wedge u_{\text{lo}}^S(t) = 1) \end{cases} \tag{40}$$

where P_{ERR} and P_{ENF} denote state predicates that realize certain control criteria. In particular, P_{ERR} defines the state region indicating intolerable values of process variables while P_{ENF} defines the state region where a given control action must be enforced. Explicitly, for the current implementation, P_{ERR} and P_{ENF} were defined as follows

$$P_{\text{ERR}}\text{: } (p(t) > p_{\max}) \vee (l(t) > l_{\max}) \tag{41}$$

$$P_{ENF}\text{: } (p(t) > p_{\text{ok max}}) \vee (l_{\text{ok min}} \le l(t) \le l_{\text{ok max}}) \tag{42}$$

with $l(t)$ and $p(t)$ denoting level and pressure in the deaerator and $p_{\max}, p_{\text{ok max}}, l_{\text{ok min}}, l_{\max}$, and $l_{\text{ok max}}$ denoting design parameters. Notice that the combined execution of the high-to-low and the control action interfaces may result in that certain commands from the high level controller no being forced to occur in the "plant." As indicated, these interfaces enforce operational and safety constraints.

VII. Experimental Realization

A. Types of Conducted Experiments

Four types of experiments were conducted, namely, off-line simulation, real-time simulation, hardware-in-the-loop simulation, and an in-plant test. The off-line simulation consisted of a non real-time simulation of the plant, controllers, and the supervisor to verify process model and environment and to modify control system design until performance objective were achieved. The real-time simulation was similar to the off-line simulation although the real-time behavior was incorporated to observe real-time response of the system. This was achieved by incorporating a real-time scheduler and by distributing simulating processes among a number of computer systems interconnected by a computer network. To investigate system components and control performance in an integrated real-time test bed, hardware-in-the-loop (HIL) test beds were developed. An HIL consists of a real-time simulation of the

plant on a distributed set of computers to meet the real-time constraints. The distinct characteristic of HIL is that the interfaces, controllers, and supervisor are implemented via control hardware such as programmable controllers and computer interface units. In this way, HIL provides real-time input–output communications and allows subcomponents to be tested before the complete system is implemented by simulating the missing subcomponents. Details on the HIL's and simulating and experimental results can be found in [6] and [8]. Next, a discussion on only the in-plant test setting and experimental results are given below.

B. In-Plant Control System Components

The described hybrid reconfigurable control system has been implemented in site at EBR-II. The control system components include the following modules with a simplified arrangement of the experimental setup illustrated in Fig. 9:

1) Two standard Bailey NETWORK 90 controllers (SC) to implement the control algorithms for the steam and condensate flow valves.
2) A Bailey NETWORK 90 programmable multi-function controller (MFC) to implement all the inter-level interfaces and the supervisor.
3) A 486-based PC computer to interface with the used microprocessor based controllers to down load and monitor control algorithms and parameters.
4) A UNIX workstation (UNIX host), which is typically a SUN or a CONCURRENT 6300 series workstation, to implement a real-time graphical interface for monitoring and controlling the in-plant test.
5) The EBR-II data acquisition system (DAS) used to obtain process information which is broadcast on the ANL ETHERNET network.
6) The two available local area networks (LANs) present at EBR-II, i.e., the NETWORK 90 plant loop and ANL ETHERNET network to communicate process information among system components.
7) Two computer interfaces units (CIU) to connect external computers to the NETWORK 90 plant loop.

C. Hierarchy of the Software Implementation

A block diagram of the control software is given in Fig. 10. The implementation consists in an hierarchy of four levels. At the lowest level, called the instrumentation level, the sensing and actuating hardware directly interacts with the physical process. It is implemented on dedicated field equipment. The level above, called the control level, computes the low-level control actions that govern the plant. It is implemented on SC's communicating on a token ring as shown in Fig. 9. These two level form the execution level as discussed in Section II. The following level corresponds to the supervision level responsible of executing the supervisory routines. It is implemented on an MFC with all programs written in C. It communicates with the execution level through a token ring network using specialized I/O routines provided by the SC's and with the above level through a computer interface unit (CIU) connected to the UNIX host by a serial line as

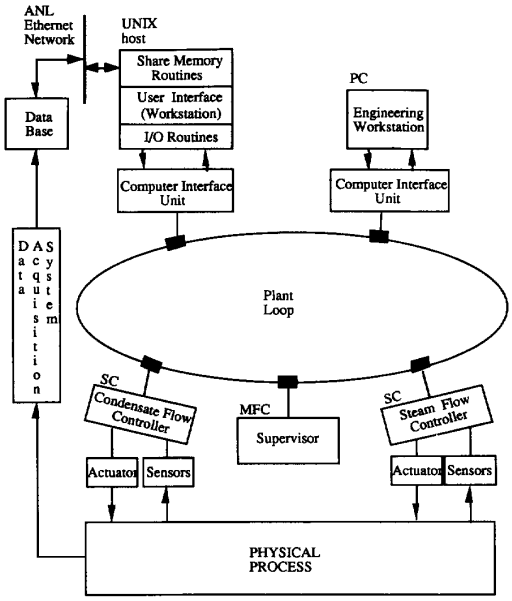


Fig. 9. Experimental in-plant setup at the Experimental Breeder Reactor II.

shown in Fig. 9. Besides providing additional process information, this CIU provides a mean for the user to communicate with the supervisor. Finally, the top level corresponds to the coordination level responsible of executing the coordinating routines. It is implemented on an UNIX host with the I/O and coordinating routines written in C and the graphic routines implemented *DATAVIEWS* from Visual Intelligence, Inc. Graphic user interfaces constantly allow the operator to monitor process variables and to send commands to the coordinator (and, consequently, to the supervisor). These windows are updated anytime process-variables or supervisor-state changes occur. For example, deaerator pressure history can be displayed or the operator can put the system in manual or automatic control. Interprocess communications among UNIX-based machines are supported via TCP/IP sockets over ethernet. A more detailed description of the software system can be found in a companion paper [7].

D. In-Plant Results

The experiment consisted of three major parts each conducted at 40 and 60 MW. The first part was controller gain verification and tuning of the alternative pressure-based condensate controller. The second and third part of the test consisted of a target 5% and 10% reduction in the steam flow to the heater (see Fig. 5) at both power levels. This paper presents only the results obtained with steam reductions at 40MW. Complete results of the experiment can be found in [6] and [8]. The goal of the following experiments is to demonstrate that a hybrid system can be used to supervise the operations of low level controllers to extend the operating life of the process during failure (in this case, loss of steam supply in the deaerator). This results in a control system with fault accommodating capabilities. Fig. 11 shows the experimental results starting from a steady power level of 40 MW for a

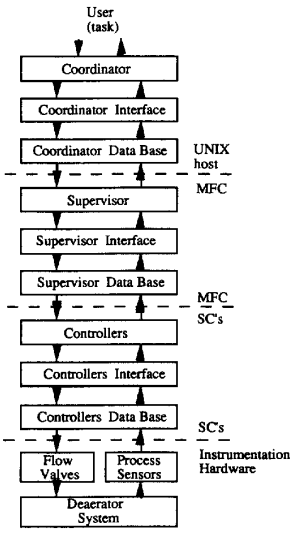


Fig. 10. Hierarchy of the implemented software.

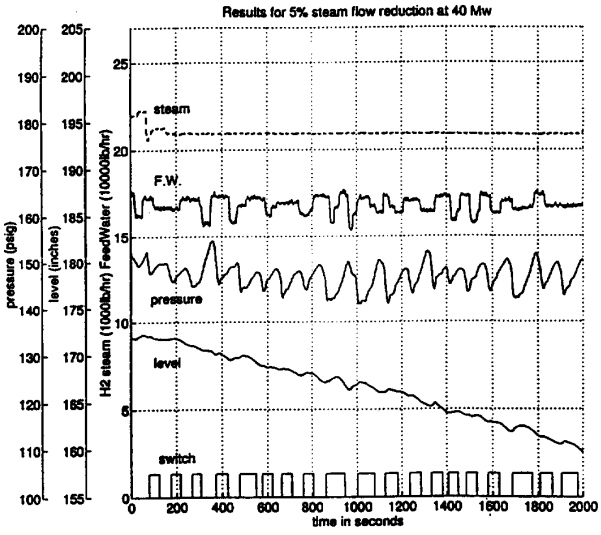


Fig. 11. Results for 5% reduction in steam flow at 40 Mw.

5% reduction in steam flow at approximately 100 seconds. The reduction was achieved by placing the normal pressure controller in manual mode and the plant operator then closed the steam supply valve. After a small transient, the steam flow remained very steady at the reduced level for the duration of the experiment. The reconfigurable controller's enforcement of level or pressure control using the condensate flow control valves is indicated by the $x_7^P[k]$ (switch) curve at the bottom of the figure. A nonzero value of $x_7^P[k]$ indicates pressure control. As can be seen, the reconfigurable controller chooses between the controllers to maintain pressure above 140 psig while level is permitted to drift downward by 12 inches over half an hour. The system could easily accommodate the loss of an additional four feet of water (an additional two hours without any other corrective actions) before technical specifications require the shutdown of the feedwater pump. Operation to such an extreme condition, however, was unnecessary to demonstrate the efficacy of the controller and the experimental procedure called for restoration of normal pressure control when level reached 160 inches. To comply with experimental safety guidelines, the operator recovered steam flow at about 2000 seconds. Without reconfiguration, the unabated initial pressure decrease would have required the operator to intervene within a few seconds as pressure fell below an experimentally prescribed 135 psig lower limit. Feedwater flow (F.W.) indicated in Fig. 11 is the rate at which water is removed from the deaerator by the feedwater pump and changes in response to external requirements to maintain steam drum level. It can be seen that feedwater flow fluctuates significantly by as much as 10% in Fig. 11. These feedwater fluctuations plus other unmeasured variations in entering and leaving deaerator flows, such as the drain flow from heater number 3 (see Fig. 5), represent external disturbances which must be accommodated by the control system. The size of the actual plant feedwater flow disturbances cause noticeable pressure transients which sometimes cause larger pressure transients than that due to the reduction in steam flow. High pressure conditions result in a poor performance evaluation which can also cause the reconfigurable controller to enforce the alternative pressure controller. The pressure effects of the normal feedwater flow fluctuations also serve to dramatize the fact that the in-plant experimentally specified steam flow disturbances (5% and 10% reductions) were conservative to avoid unnecessary stressing of the actual plant. Simulations [9] have demonstrated, however, the fault-accommodating capability of the reconfigurable controller to safely extend plant operations for severe conditions.

Fig. 12 presents the 40 MW results for a 9% reduction in steam flow at approximately 220 seconds. Here, the faulty system is maintained operative under failure for over 10 minutes (with pressure above 143 psig as level decreased by 12 inches) under the reconfigurable control strategy rather than a few seconds if it had not been used. Level decreases more rapidly in this case due to the reduced mass flow contribution of the steam flow in comparison to the 5% steam flow reduction results in Fig. 11. Feedwater flow fluctuations in this case do not cause the pressure to recover above the original 150 psig setpoint.

Before concluding, it is important to state that the experiments were intended to evaluate the process behavior under supervision when a system fault occurs and not to devise a system with the best control strategy for all cases. If that had been the case, instead of using a controller with a single driving process variable (in this case, pressure), an appropriately designed alternative condensate controller would have been one basing its output in, for example, both pressure and level. This was not done to keep the implementation as simple as possible so that the experimental procedures could be more easily developed and approved.

VIII. Summary and Conclusions

This paper presents a hybrid system, called reconfigurable supervisory control system (RSCS), to supervise and control complex dynamic processes. The proposed system is structured

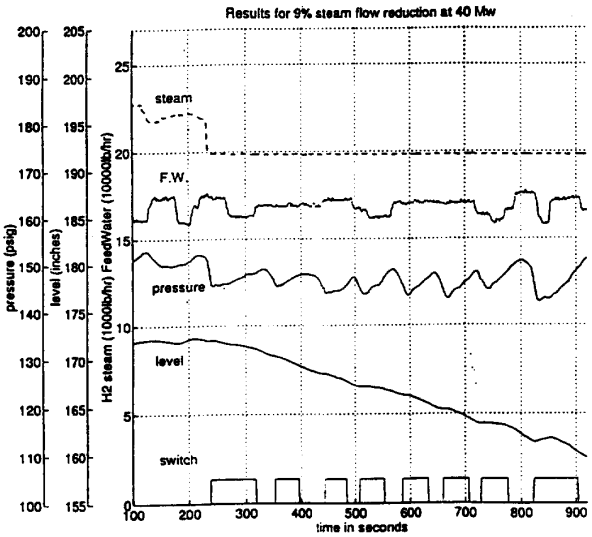


Fig. 12. Results for 9% reduction in steam flow at 40 Mw.

as a hierarchy of three levels, namely, execution, supervision, and coordination levels. While the execution level exercises direct control over the process, the supervision level generates decisions to govern the operations of the control algorithms at the execution level. The coordination level, in turn, governs the supervisory level to assure proper overall system behavior. The resulting system possesses different control and supervisory strategies to accommodate different operating conditions, adaptive behavior to react under uncertain or unfamiliar situations and the capability to coordinate distributed controllers to accomplish the system task.

The notions of hybrid control and reconfiguration are critical for the development of intelligent control systems. Hybrid refers to the fact that the system is constituted from heterogeneous components. Specifically, the system dynamic behavior results from the interaction of continuously varying and discrete-event processes. Reconfiguration which provides the capability to change system configuration based on operational conditions is necessary for controlling processes with large operational changes. In reconfigurable hybrid systems, the set of operating processes, either continuously varying or discrete-event, could be timedependent. By allowing reconfiguration at the execution and supervision levels, the proposed RSCS has the capability of generating reconfigurable CVS and DES processes.

To illustrate and evaluate the efficacy of the proposed RSCS, a reconfigurable hybrid implementation has been implemented at the Experimental Breeder Reactor II of Argonne National Laboratory. The implemented hybrid system illustrates many of the concepts discussed in this paper and demonstrates the potential of enhancing control techniques for improving safety, reliability, and performance. Although the structure of the hybrid system is simple, it can provide functions beyond the classical continuously varying control system. In particular, the tested hybrid system shows its fault accommodating capability by reconfiguring its execution level under the observation of a supervisor. In this in-plant experiment, a failure of the normal pressure regulating process was emulated by reducing the steam flow into the deaerator by as much as 10%. The hierarchical control system maintained pressure at the acceptable level for several minutes; otherwise, operator intervention would have been necessary within a few seconds. Because the tested system supervised the condensate process using a unique supervisor, there was no need for reconfiguration at the supervisory level and no coordinator was required. To fully utilize the concepts presented in this paper as described in Section II, a more complex control problem needs to be specified that would require an elaborate supervisory scheme involving many executors and supervisors and observing reconfiguration at both execution and supervision levels. In the presented research work, the concept of hybrid reconfiguration has been evaluated for process control of a power plant system. It can be employed in other engineering applications, however, including manufacturing systems; intelligent vehicle highway systems (IVHS); and command, control, communication and intelligence (C^3I) systems. This is an area for future experimental work.

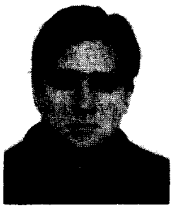
ACKNOWLEDGMENT

The authors would like to acknowledge the assistance offered by the staff of the Experimental Breeder Reactor II and, in particular, the efforts of N. C. Messick during the in-plant experiment, and thank two of the reviewers for their detailed revisions and helpful comments to improve the presentation of the paper.

REFERENCES

[1] Antsaklis, Stiver, and Lemmon, "Hybrid system modeling and autonomous control systems," in *Hybrid Systems, Lecture Notes in Computer Science vol. 736*. Berlin: Springer-Verlag, 1993.
[2] S. Balemi, *et al.*, "Supervisory control of a rapid thermal multiprocessor," *IEEE Trans. Automat. Contr.*, vol. 38, no. 7, pp. 1040–1059, July 1993.
[3] X. R. Cao and Y. C. Ho, "Models of discrete-event dynamic systems," *IEEE Contr. Syst. Mag.*, pp. 69–76, June 1990.
[4] E. W. Dijkstra, *A Discipline of Programming*. Englewood Cliffs, NJ: Prentice-Hall, 1976.
[5] P. Fishwick, "A simulation environment for multimodeling," *Discrete-Event Dynamic Systems: Theory and Applications*, vol. 3, 1993, pp. 151–171.
[6] H. E. Garcia, "A Reconfigurable Hybrid Supervisory Control System," Ph.D. dissertation, Electrical Eng. Dept., Penn. State Univ., Dec. 1993.
[7] H. E. Garcia and A. Ray, "On reconfigurable discrete-event systems," submitted for review in *Int. J. Contr.*, Dec. 1993.
[8] H. E. Garcia and R. M. Edwards, "On-line Performance Feedback for Control System Reconfiguration," IDCRL Tech. Rep., Penn State Univ., Oct. 1993.
[9] H. E. Garcia and A. Ray, "A reconfigurable discrete-event system for process control," presented at *SIAM Conf. Contr.*, Minneapolis, MN, Sept. 15–17, 1992.
[10] H. E. Garcia, A. Ray, and R. M. Edwards, "Reconfigurable control of power plants using learning automata," *IEEE Contr. Syst. Mag.*, vol. 11, no. 1, pp. 85–92, 1991.
[11] A. Gollu and P. Varaiya, "Hybrid dynamical systems," in *Proc. 28th Conf. Decis. Contr.*, Tampa, FL, Dec. 1989, pp. 2708–2712.
[12] Kohn and Nerode, "Models for hybrid systems: Automata, topologies, controllability, observability," *Hybrid Systems (Lecture Notes in Computer Science) vol. 736*. New York: Springer-Verlag, 1993.
[13] R. Kumar, V. Garg, and S. I. Marcus, "Predicates and predicate transformers for supervisory control of discrete-event dynamical systems," *IEEE Trans. Automat. Contr.*, vol. 38, no. 2, Feb. 1993.
[14] R. E. Larson, P. L. McEntire, and J. G. O'Reilly, *Tutorial, Distributed Control 2nd ed.*. Silver Spring, MD: IEEE Computer Society Press, 1982, 2nd ed.

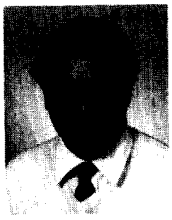
[15] Y. Li and W. M. Wonham, "Control of vector discrete-event systems I—The base model," *IEEE Trans. Automat. Contr.*, vol. 38, no. 8, pp. 1214–1227, Aug. 1993.
[16] ——, "Strict concurrency and nondeterministic control of discrete-event systems," in *Proc. 28th Conf.Decis. Contr.*, Tampa, FL, Dec. 1989, pp. 2731–2736.
[17] K. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*. Englewood Cliffs, NJ: Prentice–Hall, 1989.
[18] K. Ogata, *Modern Control Engineering*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1990.
[19] K. M. Passino and U. Ozguner, "Modeling and analysis of hybrid systems: Examples," in *Proc. 1991 IEEE Int. Symp. Intell. Contr.*, Arlington, VA, Aug. 1991, pp. 251–256.
[20] P. Peleties and R. DeCarlo, "A modeling strategy for hybrid systems based on event structures, *DEDS: Theory and Applications vol. 3*, pp. 39–69, 1993.
[21] P. J. Ramadge and W. M. Wonham, "The control of discrete-event systems," in *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, Jan. 1989.
[22] G. N. Saridis, "Analytic formulation of the principle of increasing precision with decreasing intelligence for intelligent machines," *Automatica*, vol. 25, pp. 461–467, 1989.
[23] R. S. Sreenivas and B. H. Krogh, "On condition/event systems with discrete state realizations," *DEDS: Theory and Applications vol. 1*, 1991, pp. 209–236.
[24] S. Young, D. Spanjol, and V. K. Garg, "Control of discrete-event systems modeled with deterministic Buchi automata," in *Proc. Amer. Contr. Conf.*, 1992, pp. 2814–2818.



Humberto E. Garcia was born in Caracas, Venezuela. He received an Engineering Diploma in electrical engineering with distinction from the Universidad de Carabobo, Venezuela, in 1985. He earned the Ph.D. degree and the degree of Master of Science, both in electrical engineering from The Pennsylvania State University, University Park, in 1993 and 1989, respectively.

He joined Argonne National Laboratory (ANL) in May 1994 and is currently a Researcher at this institution. Prior to joining ANL, he held research positions including research graduate posts with Penn State, the Pennsylvania Transportation Institute, and Argonne National Laboratory. His research experience and interests include: supervisory and reconfigurable control, life-extending control, real-time applications for multisensor data fusion and signal processing techniques, control and optimization of continuously varying and discrete-event dynamic processes including automobile, production and process control systems, intelligent instrumentation and control of real-time distributed processes, and design of fault-accommodating and robust control systems as applied to power and processing plants, and autonomous manufacturing and fuel management.

Dr. Garcia is a member of Tau Beta Pi and the Society for Industrial and Applied Mathematics.



Asok Ray (SM'83) received the Ph.D. degree in mechanical engineering from Northeastern University, Boston, MA, in 1976, and also graduate degrees in each of electrical engineering, computer science, and mathematics.

He joined the Pennsylvania State University, University Park, in July 1985, and is currently a Professor of Mechanical Engineering. Prior to joining Penn State, he held research and academic positions at Massachusetts Institute of Technology and Carnegie-Mellon University as well as research and management positions at GTE Strategic Systems Division, Charles Stark Draper Laboratory, and MITRE Corporation. His research experience and interests include: control and optimization of continuously varying and discrete-event dynamic systems in both deterministic and stochastic settings, modeling and analysis of thermo-mechanical fatigue and creep, intelligent instrumentation for real-time distributed processes, and design of fault-accommodating and robust control systems as applied to aeronautics and astronautics, power and processing plants, and autonomous manufacturing.

Dr. Ray has authored or co-authored over 200 research publications including 95 scholarly articles in refereed journals such as transactions of ASME, IEEE and AIAA, and also a research monograph entitled "An Integrated System for Intelligent Seam Tracking in Robotic Welding" by (Springer-Verlag, London, UK.) He is a Fellow of ASME, and an Associate Fellow of AIAA. He is currently serving as an associate editor for three journals, namely, *Journal of Dynamic Systems, Measurement, and Control, Transactions of the American Society of Mechanical Engineers,* and *International Journal of Flexible Manufacturing Systems*. He is registered as a Professional Electrical Engineer in the Commonwealth of Massachusetts.

Robert M. Edwards (M'92) received the B.S. degree from the Pennsylvania State University, University Park, in 1971, the Master's degree from University of Wisconsin in 1972, and the Ph.D. degree from the Pennsylvania Sate University in 1991, all in nuclear engineering.

After receiving the Master's degree in 1972, he was employed by General Atomic, Combustion Engineering, and LeMont Scientific before returning to Penn State in 1987. He is currently an Assistant Professor of Nuclear Engineering at the Pennsylvania State University where he teaches courses in reactor experiments, reactor control, nuclear digital data acquisition processing and control, power plant dynamics and control, and power plant intelligent distributed control. He has conducted research in reactor and power plant operations including optimal, robust, fuzzy logic, neural network, and intelligent reconfigurable control as well as expert systems and artificial intelligent applications.