Intelligent Monitoring and Control

Barbara Hayes-Roth, Richard Washington,
Rattikorn Hewett and Micheal Hewett
Knowledge Systems Laboratory
Stanford University
701 Welch Road
Palo Alto, California 94304

Adam Seiver
Palo Alto Veterans Administration
Medical Center
3801 Miranda Avenue
Palo Alto, California 94304

Abstract

Intelligent monitoring and control involves observing and guiding the behavior of a physical system toward some objective, with real-time constraints on the utility of particular actions. Generic functional requirements for this task include: integration of perception, reasoning, and action; integration of multiple reasoning activities; reasoning about complex, time-varying systems; coordination of multiple response modes; dynamic allocation of limited computational resources. We illustrate these requirements in the domain of patient monitoring in a surgical intensive care unit (SICU). We propose a generic architecture, designed and implemented in layers: top-level system organization; reasoning architecture; generic reasoning skills and knowledge representation; first-principles knowledge of physical systems; domain knowledge. We illustrate the architecture in the "Guardian" system for SICU monitoring and describe Guardian's performance on an illustrative scenario. Finally, we discuss the generality and limitations of the proposed architecture.

1 The Problem

Intelligent monitoring and control involves observing and guiding the behavior of a physical system toward some objective, with real-time constraints on the utility of particular actions. Control theory ([Bollinger and Duffie, 1988],[Hale, 1973]) is useful for tasks that permit a straightforward mapping between sensed data values and appropriate control actions. By contrast, we are concerned with tasks that require a more "intelligent" approach, including: interpretation and prediction of system behavior, diagnosis of exceptional events, explanation of causal mechanisms, reactive response to urgent events, reasoned response to other events, and planning of longer-term courses of action. Such tasks occur in a variety of domains, for example: process control, crisis management, construction management, equipment monitoring, tutoring, and medical monitoring([Fagan, 1980], [Fagan *et* a/., 1980], [Hayes-Roth, 1989],[Pardee and Hayes-Roth, 1987], [Raulefs *et* a/., 1987]).

Our goal is to develop a generic AI architecture for intelligent monitoring and control, suitable for application in multiple domains. In addition to its potential practical utility, this research addresses fundamental AI issues, including: reasoning methods, knowledge representation, resource allocation, distributed intelligence, and real-time performance. In this paper, we propose an architecture and illustrate it in the domain of patient monitoring in a surgical intensive care unit (SICU). Section 2 identifies important generic requirements for intelligent monitoring and control and illustrates them for the SICU monitoring task. Section 3 presents the proposed architecture, illustrates it with an experimental SICU monitoring system called "Guardian", and discusses how the architecture addresses the requirements. Section 4 illustrates Guardian's performance on a characteristic scenario. Sections 5 and 6 discuss the generality and limitations of the architecture.

2 Requirements

We aim to address generic AIS requirements for: integration of perception, reasoning, and action; integration of multiple reasoning activities; reasoning about complex, time-varying systems; coordination of multiple response modes; and dynamic allocation of limited computational resources.

Consider SICU monitoring. Patients in the SICU have had major surgery and suffer temporary failure of one or more organ systems. Therefore, life-support devices assume the fundamental functions of the ailing organ system until it can heal and resume its normal function. For example, a ventilator is an artificial breathing machine. Most life-support devices have sensors that measure relevant parameters of physiological function and settings to determine how much assistance the device provides.

*This research was supported by grants from DARPA and NIH and by gifts from Rockwell International Corp. and FMC Corp. The Palo Alto VAMC supports Adam Seiver's participation in the project. We gratefully acknowledge contributions by Reed Hastings and Nicholas Parlante to an early version of Guardian and to Adnan Darwiche, who recently joined the project. We have benefitted from discussions with Lawrence Fagan and his students, who are working on a related 1CU monitoring system, called 'QQ.' We thank Edward Feigenbaum for sponsoring the work at the Knowledge Systems Laboratory.

The ventilator has sensors that measure gas pressures and flows in the patient-ventilator system. It has settings that determine the number of breaths delivered to the patient per minute, the volume of air blown into the lungs on each breath, and the percent oxygen used to enrich the breathing mixture. In addition to these automatic measurements and controls, SICU monitoring involves other regular observations (e.g., chest xrays, taken once or twice a day), discretionary observations (e.g., laboratory tests of blood gases), and therapeutic actions (e.g., adjustment of a ventilator tube).

Because life-support devices injure as well as sustain patients, one objective of SICU monitoring is to wean the patient from the device as rapidly as is possible and consistent with other therapeutic objectives. Based on a model of the patient's physiological impairment and expected rate of recovery, the physician orders an initial configuration of device settings that substantially augment the patient's own function, followed by a program of modifications to those settings that gradually (over a period of days or weeks) reduce the level of assistance to zero, followed by device withdrawal. Given the complexity of the biological system, the characteristic instability of post-surgical patients, and the duration of the weaning period, however, the patient model may be imprecise or incorrect. Accordingly, SICU staff monitor the patient closely so as to validate the assumptions underlying the weaning plan, refine the plan to coordinate with the detailed progress of the patient's physiological function, modify the plan when the underlying assumptions prove incorrect, and perform additional actions to diagnose and correct other unanticipated problems.

SICU monitoring instantiates the above requirements as follows. It requires integration of perception of patient data; reasoning about the patient's condition, progress, and therapy; and actions to implement or recommend therapeutic interventions. It requires integration of reasoning activities for: interpretation of patient data, diagnosis of observed signs and symptoms, prediction and explanation of the patient's progress, reaction to urgent patient conditions, and planning of longer-term therapy. It requires reasoning about the behavior of an impaired biological organism, partially sustained by physical devices, over a period of days or weeks. It requires coordination of: immediate reactive responses to emergencies, prompt associative responses where clinical knowledge is applicable, and deliberate reasoned responses to complex or evolving patient conditions. It requires dynamic allocation of limited computational resources among competing monitoring activities presenting variable and uncertain resource requirements.

3 Proposed Architecture

Figure 1 illustrates the proposed architecture, as currently implemented for Guardian. The following sections describe the top-level system organization, reasoning architecture, reasoning skills, and knowledge representation.

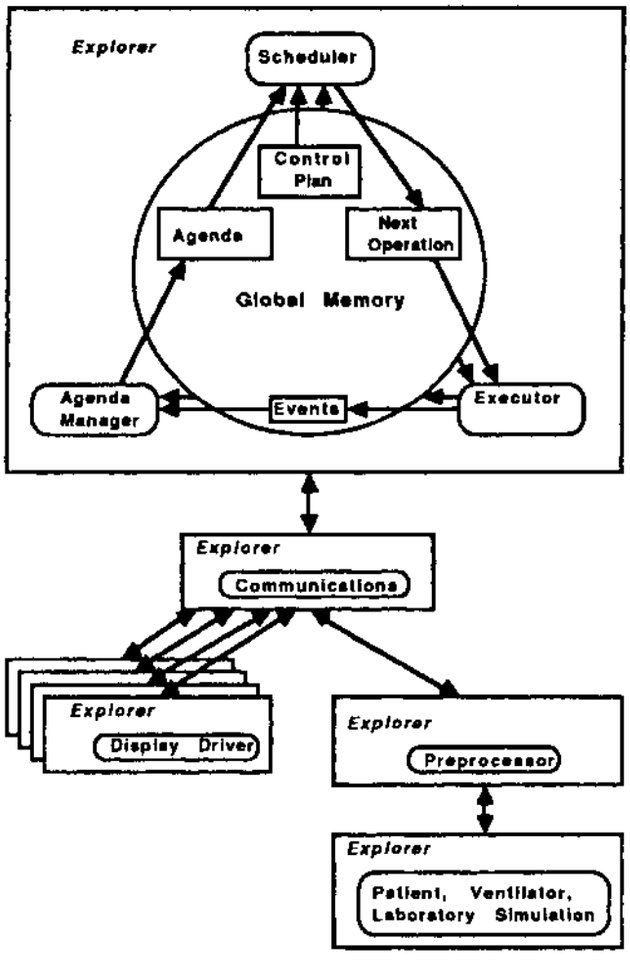

Figure 1: Guardian's Architecture

3.1 Top-Level System Organization

To enable a monitoring system to interact with a dynamic environment, our top-level system organization ([Hayes-Roth, 1987b], [Hayes-Roth, 1989], [Hayes-roth *et* a/., 1989]) provides several types of loosely-coupled subsystems: (a) a central *reasoning system* performs all knowledge-based reasoning; (b) a *communication interface* mediates I/O between the reasoning system and all perception/action subsystems via shared I/O buflers[llewett and Hayes-Roth, 1989]; (c) each of several perceptual subsystems comprises *sensors* and an associated *preprocessor* that interprets and filters sensed data according to instructions from the reasoning system [Washington and Hayes-Roth, 1989]; and (d) each of several action subsystems comprises *effectors* and an associated driver that interprets and executes action programs according to instructions from the reasoning system. For example, in Guardian, a single preprocessor handles inputs from all sensors associated with a patient-ventilator-laboratory simulation. Several drivers handle graphical displays and user interactions on different devices.

Different subsystems and their constituent processes operate concurrently, enabling a monitoring system to

perform each of its three basic functions-perceive events in the environment, perform actions that affect the environment, and reason about its monitoring activities-without interrupting or otherwise interfering with one another. The results of these activities influence one another asynchronously. For example, Guardian simultaneously perceives new patient data, conducts its ongoing reasoning, and performs intended user interactions. It incorporates newly perceived patient data into its reasoning as those data appear in its input buffers and performs newly intended actions determined by its reasoning as they appear in its output buffers.

The proposed organization also provides complexity-management functions. Preprocessors shield the reasoning system from the mass of non-critical data generated in the environment. For example, Guardian's patient simulation produces values for twenty variables, each sensed up to ten times per second, and its laboratory simulation produces values for several other variables, as requested, and returns them with realistic time delays. By contrast, the reasoning system currently incorporates at most one input per second. Guardian's preprocessor protects it from data overload, while insuring that it notices important data, by abstracting sensed data (e.g., as critical values, averages, or trends) and filtering the abstract values according to Guardian's current focus of attention. Conversely, drivers relieve the reasoning system of the details of action execution. For example, Guardian's drivers handle programs that explain reasoning activities and results.

Finally, this organization supports immediate reactive response modes by permitting a tight coupling of sensory input to action execution within perception/action subsystems. For example, Guardian's display drivers conduct low-level user interactions-e.g., menu-driven elaborations and explanations-without mediation by the reasoning system. We discuss architectural support for other response modes below.

3.2 Reasoning Architecture

We adapted the *dynamic control architecture[If Ayes*-Roth, 1985], implemented as BB1 [Hayes-Roth and Hewett, 1988], for the reasoning system. As shown in Figure 1, all reasoning operations occur in a *global memory* that contains all of the facts, inferences, events, plans, etc. known to the system. The architecture iterates a three-step reasoning cycle. First, the *agenda manager* identifies reasoning operations enabled by recent perceptual events appearing in the input buffers and cognitive events produced by prior reasoning operations. Second, the *scheduler* chooses as the next operation the one that best matches the current control plan. Third, the *executor* executes the chosen operation, changing information in the global memory-possibly the control plan itself-and recording a corresponding cognitive event.

This architecture supports the several different reasoning skills required by a monitoring system (discussed below) and enables it to integrate execution of those skills by interleaving their constituent reasoning operations. (We are investigating parallel execution of distributed reasoning tasks [Hayes-roth *et* a/., 1989].) More important, concurrent reasoning activities can influence one another asynchronously by recording and inspecting results in the global memory. For example, Guardian has reasoning skills for classifying input data and diagnosing problems. By performing these tasks concurrently, it can reason about problems in dynamic biological systems, classifying relevant new data as they occur and immediately incorporating them into its ongoing diagnosis.

The architecture also enables a monitoring system to allocate its limited computational resources among competing monitoring activities. By interleaving control planning operations with other reasoning operations, the system incrementally constructs and modifies control plans for its own behavior in real time. Control plans may describe intended behavior at variable levels of specificity and for variable time intervals. At each point in time, the system behaves in accordance with its current control plan. Thus, a system can integrate opportunistic responses to unanticipated events with carefully planned courses of action ([Hayes-Roth, 1987a], [Johnson and Hayes-Roth, 1987]). For example, given an observed problem, Guardian decides which of two diagnostic methods to use. If the problem is not serious or occurs at the same time as many other problems, Guardian responds reactively and moves on to the next problem. However, if the problem is potentially serious or occurs during an uneventful period, Guardian might undertake its more analytic, but computationally expensive model-based reasoning.

3.3 Reasoning Skills

As discussed above, intelligent monitoring and control requires multiple reasoning skills. Guardian has domain-independent skills for: (a) determining appropriate perceptual filters, based on the system's available resources and ongoing tasks[Washington and Hayes-Roth, 1989]; (b) abstracting perceived data as temporal episodes of defined value categories; (c) diagnosing the most likely causes of commonly observed problems based on Bayesian belief networks[Pearl, 1988]; (d) explaining the causal mechanism underlying a diagnosis based on explicit structure/function models [Hayes-Roth *et* a/., 1988]; (e) generating plausible diagnoses for unfamiliar problems or those for which the most likely diagnosis is incorrect based on structure/function models [Hayes-Roth *et* a/., 1988]; and (f) identifying standard actions for commonly diagnosed problems based on association networks.

In developing these reasoning skills, we emphasize a monitoring system's need to coordinate multiple response modes. As discussed above, tight coupling of sensory input to action execution in perception/action subsystems provides an immediate reactive response mode. R,eactive response is appropriate for critical events or those that do not merit more careful situation assessment and planning. Guardian's reactive management of low-level user interactions falls in the latter category. Although some investigators argue for exclusively or primarily reactive response modes([Agre and

Chapman, 1987], [Brooks, 1985], [Firby, 1987], [Nilsson, 1988], [Rosenschein and Kaelbling, 1986], [Schoppers, 1987]), we believe that many monitoring and control tasks require reasoned response modes as well[Hayes-Roth, 1987aj.

For example, our belief-network operations for diagnosing common problems and identifying standard corrective actions are associative. By performing instances of these two operations in succession, Guardian responds quickly, on the order of seconds in the current implementation, performing the standard corrective action for the most likely diagnosis of an immediate problem. We could further reduce the associative response time by combining diagnosis and action in a single operation. However, the current response time is satisfactory in Guardian's domain and the interruptability between diagnosis and action is useful. Associative response is appropriate for situations that require knowledge-based reasoning-though not very deep-and prompt response.

By contrast, our model-based methods for diagnosis and planning (the latter is not yet implemented) are analytic. By exploiting these skills, guardian responds more slowly, on the order of minutes in the current implementation. It develops an explicit model of the causal mechanism underlying the observed problem, plans a course of action-in some cases, coordinated with the patient's changing condition-based on that model, and instantiates that planned course of action in real time. Model-based response is appropriate for unfamiliar problems, problems that require a precise or longer-term course of action, or problems that require actions synchronized with external events.

The explicit control provided by our reasoning architecture allows a system to choose among alternative reasoning activities and to interleave multiple activites. Thus, Guardian can instruct perception/action subsystems to react in specified ways to specified input data. Given a problem that requires reasoning, Guardian can decide between associative and model-based responses. Alternatively, it can combine associative diagnosis with model-based action or combine model-based diagnosis with associative action. In some cases, it may respond to a problem reactively or associatively, and then plan a longer-term course of action.

3.4 Knowledge Representation

As discussed above, our reasoning architecture presumes a globally accessible memory. For this purpose, we use a conceptual network representation[Sowa, 1984], with predefined architectural concepts (for example, perceptual and cognitive events, control plans, reasoning operations). We have begun to develop an ontology of generic monitoring concepts and conceptual respresentations in three areas. *Reasoning knowledge* includes generic operations and control strategies for component reasoning skills, such as those discussed above. *Domain knowledge* includes both associative knowledge and factual knowledge. For example, Guardian's associative knowledge currently includes: networks relating semantic categories to value ranges for each measured patient variable; Bayesian belief networks[Pearl, 1988] relating commonly observed signs to likely underlying faults; and association networks relating likely faults to standard treatments. Its factual knowledge currently includes explicit structure/function models of the normal anatomy and physiology of the respiratory, circulatory, pulmonary exchange, tissue exchange, and tissue metablism systems and a similar model of the ventilator. *First-principles knowledge* includes domain-indendent knowledge of the physical world. Guardian currently has knowledge of the normal and abnormal structure and function of generic flow, diffusion, and metabolic systems, which it uses to diagnose or explain problems in particular organ systems[Hayes-Roth *et a*/., 1988].

4 Illustrative Performance

4.1 Guardian Implementation

Guardian is a prototype SICU monitoring system. It is designed within the architecture described above, implemented in CommonLisp, and runs on a configuration of TI Explorers, (see Figure 1). Guardian can handle a growing variety of SICU scenarios. In this section, we discuss its performance on a scenario that illustrates how the underlying architecture enables Guardian to meet the generic requirements introduced in section 2.

Before discussing the scenario, we make a few observations. First, all of the capabilities discussed below are implemented except as noted in the text. Second, the references to real time are approximately correct, with accumulated error on the order of minutes, due to uncontrolled variability in network communications times. Third, although Guardian can exercise closed-loop control over the patient-ventilator simulation, it would be allowed to act only in an advisory capacity in the hospital. In this scenario, we model the hospital situation, with the assumption that SICU personnel promptly execute all actions recommended by Guardian. Fourth, for brevity, we omit the details of communications between the reasoning system and perception/action subsystems. However, in every case of perception, the perceptual subsystem relays input data to the communication interface, which inserts it in the appropriate input buffer in the reasoning system. Conversely, in every case of action, the reasoning system places an action description in an appropriate output buffer, from which the communication interface retrieves it and relays it to the appropriate action subsystem. Finally, also for brevity, we omit discussion of Guardian's extensive graphical displays and user interactions. In fact, Guardian displays all of its observations, inferences, and recommendations as they occur.

4.2 Overview of the Scenario

In this scenario, Guardian monitors a ventilator-assisted SICU patient who has just returned from the operating room. The physician has ordered standard ventilator settings, including a breathing rate of 8 breaths per minute and a tidal volume of 1200cc.

As we shall see, the scenario includes two errors. The first is the physician's error in ordering standard ventilator settings. Post-surgical patients typically are cold

upon their return from the operating room. Because a low temperature slows down metabolism, patients need to breath more slowly-lower rate or tidal volume or both-than patients with normal temperatures. Otherwise, they will exhale too much CO_2, causing a low partial pressure of CO_2 in the arterial blood, a condition known as 'hypocapnia.' Of course, a post-operative patient will warm up to normal temperature during the next two hours. Therefore, the physician should order a low initial breathing rate (say 6 breaths per minute), gradually raised to the normal 8 breaths per second as the patient's temperature rises to normal. Although physicians understand such relationships very well, it is not unusual for this kind of error to occur in communications with SICU staff. The second error is Guardian's error in failing to notice that the standard ventilator settings are inappropriate, given the patient's temperature. Again, it is not unusual for SICU staff to make this kind of error, either by oversight or because they are busy elsewhere. Although Guardian could easily avoid making this particular error, we allow it to occur in the scenario to illustrate how Guardian can recover from its own errors or those made by others.

4.3 Getting Started

Beginning at 1:00, Guardian decides to follow standard set-up procedures for monitoring a newly post-surgical patient. It sends its preprocessor a standard initial filter for each patient variable: 'send a newly sensed value only when (a) it differs by at least p% from the previously sent value; or (b) at least m minutes have elapsed since the last value was sent.' Guardian begins classifying the filtered patient data as they arrive and organizing them in temporal episodes of known value categories. During the first few minutes, Guardian adjusts filters for individual variables to approximate the maximum data rate it can handle in real time without overflowing its input buffers. Guardian also requests an initial analysis of blood gases from the laboratory simulation. It decides to inspect the lab results when they return in approximately 20 minutes. During the remainder of this time period, Guardian interprets input data. All twenty variables remain normal, except for the patient's temperature, which is low. Because this is not critical, Guardian ignores it.

4.4 Reactive Response

At 1:20 when the requested lab result returns, Guardian classifies it as Mow PaC02-hypocapnia.' Although both associative and model-based response modes are possible, Guardian chooses the associative mode by default. It diagnoses 'hyperventilation' as the most likely cause of the observed hypocapnia and determines that 'decrease the breathing rate to 6' is the most appropriate action. Guardian advises SICU staff to take that action and decides to request another lab test in twenty minutes to confirm the effect of its action. It continues to interpret input data.

At 1:40 Guardian requests another lab test and decides to evaluate it-expecting a normal result when it returns. It continues to interpret input data.

At 2:00 the lab test returns and Guardian confirms the expected normal $PaC0_2$- It decides to request another lab test and to evaluate the result when it returns. It continues to interpret input data.

At 2:20 when the lab test returns, Guardian classifies it as 'high PaC02~hypercapnia.' Responding associatively, it diagnoses 'hypoventilation' as the most likely cause and determines that 'increase the breathing rate to 7' is the most appropriate action. It advises SICU personnel to take that action and continues to interpret input data.

4.5 Model-Based Reasoning

At 2:22, given the patient's two successive $PaC0_2$ problems, Guardian decides to analyze the causal mechanism underlying these problems more carefully. Because its knowledge of model-based reasoning methods specifies high computational requirements, Guardian conserves resources by adopting stronger perceptual filters: 'send sensed values only when (a) they differ from previously sent values by at least twice the current change threshold; or (b) at least twice the current allowed time interval has elapsed since the last value was sent.' As a consequence, Guardian continues to interpret new input data, but spends less of its computational resources doing so. In a distributed implementation, Guardian could assign its model-based reasoning task to a remote processor. However, any system having limited computational resources will encounter situations in which it has a reduced capability for reasoning about new inputs and, therefore, should adopt different perceptual filters.

Guardian uses its domain knowledge and first-principles knowledge to construct models of alternative hypothetical faults that could cause the observed problems. Among its several hypotheses are these two. 'Hypothesis A: Decreased PaC02 is caused by high breathing rate. Decreased partial pressure of C02 in the arteries is caused by decreased partial pressure of C02 in the pulmonary exchange system, which is caused by decreased amount of C02 in the respiratory system, which is caused by increased delivery of other gases from the ventilator, which is caused by high breathing rate set at the ventilator.' 'Hypothesis B: Decreased PAC02 is caused by low temperature. Decreased partial pressure of C02 in the arteries is caused by decreased delivery of C02 from the tissue exchange system, which is caused by decreased delivery of C02 from the tissue metabolism system, which is caused by decreased production of C02 by the tissue metabolism system, which is caused by decreased consumption of 02 by the tissue metabolism system, which is caused by low temperature.'

Guardian distinguishes these two hypotheses from others it constructs for different reasons. Hypothesis A provides a causal rationale for Guardian's reactive diagnosis, 'hyperventilation,' and corrective action, 'lower the breating rate to 6.' Hypothesis B identifies a previously observed (and ignored) problem, 'low temperature,' as a causal factor in hypocapnia.

Guardian constructs and distinguishes the corresponding two hypotheses to explain the patient's subsequent hypercapnia. 'Hypothesis C: Increased PaC02 is caused

by low breathing rate.' 'Hypothesis D: Increased PaC02 is caused by high temperature.'

Now Guardian observes that the patient's temperature was lower at 1:00 (when the blood sample showing hypocapnia was taken) than it was at 1:40 (when the blood sample showing normal $PaCO_2$ was taken, and lower then than it was at 2:00 (when the blood sample showing hypercapnia was taken). Given these observations, Guardian should build a more encompassing model relating the patient's temperature, its own settings of breathing rate, and the resulting variations in arterial CO_2 throughout the monitoring period. Given its knowledge that the patient's temperature will return to normal, Guardian should predict that its recent increase in the breathing rate will eliminate the patient's present hypercapnia only temporarily. Indeed, Guardian should have predicted that its original lowering of the patient's breathing rate would eliminate the patient's original hypocapnia only temporarily, leading eventually to the present hypercapnia. (Note: We are now implementing this prediction skill.)

4.6 Planned Response

Although Guardian could eventually stabilize the patient's arterial CO_2 at a normal level with a normal breathing rate and temperature by continuing to respond associatively, the resulting alternation of normal $PaCO_2$ and hypercapnia episodes would be a suboptimal route to that goal. Instead, Guardian should plan to coordinate changes in breathing rate with changes in temperature, so as to maintain a normal $PaCO_2$ throughout: 'Increase breathing rate by 10 percent for every 1 degree increase in temperature, expected to occur every half hour.' Having done so, it should follow this plan, reinstate its original perceptual filters, and resume its original rate of interpreting input data. It should interleave planned operations to confirm expected increases in temperature and make associated increases in breathing rate. (Note: We are now implementing this planning skill.)

5 Generality

To maximize the generality, extensibility, and reusability of the proposed architecture, we designed it in layers. The system organization and reasoning architecture are application independent. The reasoning skills, first principles, and domain knowledge apply to successively narrower classes of applications. By modifying their contents, we can extend a given application or develop a new one. For example, we are extending Guardian's expertise incrementally at all three levels. With its original reasoning skills (data classification, associative diagnosis, and model-based diagnosis), first-principles knowledge (generic flow systems), and domain knowledge (the respiratory system), Guardian classified respiratory data and performed associative and model-based diagnosis of certain respiratory problems. With new knowledge of the circulatory system, Guardian handled similar problems in the circulatory system. Given a new reasoning skill for model-based explanation, Guardian explained the causal mechanisms underlying diagnosed problems in both systems. With addititional reasoning skills (reactive response), first-principles knowledge (diffusion and metabolic systems), and domain knowledge (the tissue exchange and metabolic systems), the current version of Guardian handles a range of problems arising within each system or from interactions among systems. By replacing Guardian's knowledge at particular levels, we could create new monitoring systems. For example, a power plant monitor might combine Guardian's reasoning skills and generic flow models with new knowledge of the structure and function of power plants. Similarly, a materials processing monitor might incorporate Guardian's reasoning skills with new generic models for heating, cooling, and compaction processes and new domain knowledge of particular materials processes.

6 Limitations

In addition to obvious limitations on the amount of knowledge currently implemented, the proposed architecture has (at least) two more fundamental limitations.

First, like all AI techniques, the proposed architecture must pass the test of scalability. There are many relevant scaling factors, including: number of data variables, sensed data rates, number of alternative diagnoses, number of co-occurring and interacting problems, complexity of physical models, and number and complexity of alternative responses. We need to investigate how a system's performance degrades as we increase its complexity along these dimensions.

Second, intelligent monitoring and control is a real-time task, imposing hard and soft time constraints on the utility of observations, conclusions, and responses. The proposed architecture includes features designed to give a monitoring system knowledge-based control over its resource allocations in order to address real-time constraints. These features are its programmable perceptual preprocessors and action drivers and its dynamic control of reasoning behavior. However, we have not yet provided an explicit representation of time or a quantitative basis for making time-constrained resource allocations. In addition, the BB1 execution cycle involves unbounded computation times, which can undermine even the most "intelligent" resource allocation behavior. We are attempting to address both of these limitations in ongoing research.

References

[Agre and Chapman, 1987] P. Agre and C. Chapman. Pengi: An implementation of a theory of activity. In *Proceedings of the National Conference on Artificial Intelligence, AAAI-87,* 1987.

[Bollinger and Duffie, 1988] J. Bollinger and N. Duffie. *Computer Control of Machines and Processes.* 1988.

[Brooks, 1985] R.A. Brooks. A robust layered control system for a mobile robot. Technical report, M.I.T. AI Laboratory, 1985.

[Fagan *et al.*, 1980] L. Fagan, E.H. Shortliffe, and B.G. Buchanan. Computer-based medical decision making: From mycin to vm. *Automedica,* 3:97-106, 1980.

[Fagan, 1980] L. Fagan. Vm: Representing time-dependent relations in a medical setting. Technical report, Stanford University, 1980.

[Firby, 1987] R. Firby. An investigation into reactive planning in complex domains. In Proceedings of the National Conference on Artificial Intelligence, AAAI-87, 1987.

[Hale, 1973] F. Hale. Introduction to control system analysis and design. Prentice-Hall, 1973.

[Hayes-Roth and Hewett, 1988] B. Hayes-Roth and M. Hewett. Building systems in the bb1 architecture. In R. Engelmore and A. Morgan, editors, Blackboard Systems. London: Addison-Wesley, 1988.

[Hayes-Roth et a/., 1988] B. Hayes-Roth, R. Hewett, and A. Seiver. Diagnostic explanation using generic models. Technical Report KSL-88-20, Stanford, Ca.: Stanford University, 1988.

[Hayes-roth et al, 1989] B. Hayes-roth, M. Hewett, R. Washington, R. Hewett, and A. Seiver. Distributing intelligence within an individual. In L. Gasser and M.N. Huhns, editors, Distributed Artificial Intelligence, Volume 2. Morgan Kaufmann, 1989.

[Hayes-Roth, 1985] B. Hayes-Roth. A blackboard architecture for control. Artificial Intelligence Journal, 26:251-321, 1985.

[Hayes-Roth, 1987a] B. Hayes-Roth. Dynamic control planning in adaptive intelligent systems. Proceedings of the DARPA Knowledge-Based Planning Workshop, 1987.

[Hayes-Roth, 1987b] B. Hayes-Roth. A multi-processor interrupt-driven architecture for adaptive intelligent systems. Technical Report KSL-87-31, Stanford, Ca.: Stanford University, 1987.

[Hayes-Roth, 1989] B. Hayes-Roth. Making intelligent systems adaptive. In K. VanLehn, editor, Architectures for Intelligence. Lawrence Erlbaum, 1989.

[Hewett and Hayes-Roth, 1989] M. Hewett and B. Hayes-Roth. Real-time i/o in knowledge-based systems. In V. Jagannathan and R.T. Dodhiawala, editors, Current Trends in Blackboard Systems. 1989.

[Johnson and Hayes-Roth, 1987] M.V. Johnson and B. Hayes-Roth. Integrating diverse reasoning methods in the bbl blackboard control architecture. Proceedings of the National Conference on Artificial Intelligence, AAA 1-87, 1987.

[Nilsson, 1988] N. Nilsson. Action networks. Technical report, Stanford University, 1988.

[Pardee and Hayes-Roth, 1987] W. Pardee and B. Hayes-Roth. An intelligent materials processor. Technical report, Rockwell Science Center, 1987.

[Pearl, 1988] J. Pearl. Probabilistic reasoning in intelligent systems: Networks of plausible inferences. Morgan Kaufmann Publishers, Inc., 1988.

[Raulefs et al, 1987] P. Raulefs, B. D'Ambrosio, M.R. Fehling, S. Forrest, and M. Wilber. Real-time process management for materials composition. Proceedings of the 3rd IEEE Artificial Intelligence Applications Conference, 1987.

[Rosenschein and Kaelbling, 1986] S.J. Rosenschein and L.P. Kaelbling. The synthesis of machines with provably epistemic properties. In Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning about Knowledge, 1986.

[Schoppers, 1987] M.J. Schoppers. Universal plans for reactive robots in unpredictable environments. In Proceedings of the International Conference on Artificial Intelligence, IJCAI-87, 1987.

[Sowa, 1984] J.F. Sowa. Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, 1984.

[Washington and Hayes-Roth, 1989] R. Washington and B. Hayes-Roth. Input data management in real-time ai systems. In Proceedings of the International Conference on Artificial Intelligence, UCAI-89. 1989.