DTIC FILE COPY

October 1988

Report No. STAN-CS-88-1226

Also Numbered KSL-88-64

AD-A200 913

Making Intelligent Systems Adaptive

by

Barbara Hayes-Roth

Department of Computer Science

Stanford University

Stanford, California 94305

DTIC ELECTE NOV 0 9 1988 S D D

DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

LELAND STANFORD JUNIOR UNIVERSITY ORGANIZED 1891

88 11 08 007

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188
Exp Date Jun 30, 1986

1a REPORT SECURITY CLASSIFICATION

1b. RESTRICTIVE MARKINGS

2a SECURITY CLASSIFICATION AUTHORITY

3. DISTRIBUTION/AVAILABILITY OF REPORT

2b DECLASSIFICATION/DOWNGRADING SCHEDULE

4 PERFORMING ORGANIZATION REPORT NUMBER(S)
STAN-CS-88-1226

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a NAME OF PERFORMING ORGANIZATION
Computer Science Department

6b OFFICE SYMBOL (If applicable)

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS (City, State, and ZIP Code)
Stanford University
Stanford, CA 94305

7b. ADDRESS (City, State, and ZIP Code)

8a NAME OF FUNDING/SPONSORING ORGANIZATION
DARPA

8b OFFICE SYMBOL (If applicable)

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
00039-83-C-0136

8c. ADDRESS (City, State, and ZIP Code)
Arlington, Virginia

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11 TITLE (Include Security Classification)
Making Intelligent Systems Adaptive

12 PERSONAL AUTHOR(S)
Barbara Hayes-Roth

13a TYPE OF REPORT	13b TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

Contemporary intelligent systems are isolated problem-solvers. They accept particular classes of problems, reason about them, perhaps request additional information, and eventually produce solutions. By contrast, human beings and other intelligent animals continuously adapt to the demands and opportunities presented by a dynamic environment. Adaptation plays a critical role in everyday behaviors, such as conducting a conversation, as well as in sophisticated professional behaviors, such as monitoring critically ill medical patients. To make intelligent systems similarly adaptive, we must augment their reasoning capabilities with capabilities for perception and action. Equally important, we must endow them with an attentional mechanism to allocate their limited computational resources among competing perceptions, actions, and cognitions, in real time. In this paper, we discuss functional objectives for "adaptive intelligent systems," an architecture designed to achieve those objectives, and our continuing study of these objectives and architecture in the context of particular tasks.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT
☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION

22a NAME OF RESPONSIBLE INDIVIDUAL

22b. TELEPHONE (Include Area Code)

22c OFFICE SYMBOL

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

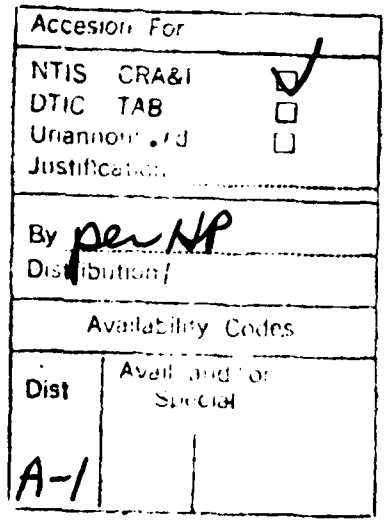
Knowledge Systems Laboratory
Report No. KSL 88-64

April, 1988 -- Revised July, 1988

Making Intelligent Systems Adaptive

by

Barbara Hayes-Roth



KNOWLEDGE SYSTEMS LABORATORY
Computer Science Department
Stanford University
Stanford, California 94305

To appear as a chapter in the book, Architectures for Intelligence, edited by K. Van Lehn and published by Lawrence Erlbaum, 1988. The research was supported by DARPA Contract #00039-83-C-0136 and gifts from Rockwell International Corporation and FMC Corporation.

Abstract

Contemporary intelligent systems are isolated problem-solvers. They accept particular classes of problems, reason about them, perhaps request additional information, and eventually produce solutions. By contrast, human beings and other intelligent animals continuously adapt to the demands and opportunities presented by a dynamic environment. Adaptation plays a critical role in everyday behaviors, such as conducting a conversation, as well as in sophisticated professional behaviors, such as monitoring critically ill medical patients. To make intelligent systems similarly adaptive, we must augment their reasoning capabilities with capabilities for perception and action. Equally important, we must endow them with an attentional mechanism to allocate their limited computational resources among competing perceptions, actions, and cognitions, in real time. In this paper, we discuss functional objectives for "adaptive intelligent systems," an architecture designed to achieve those objectives, and our continuing study of these objectives and architecture in the context of particular tasks.

1 Introduction

Contemporary artificial intelligence systems exhibit important aspects of intelligence. They possess knowledge and heuristic problem-solving skills. They solve problems typically requiring sophisticated human expertise and they do so in a manner that is evocative of human problem-solving behavior. On the other hand, most contemporary AI systems are static, isolated problem solvers. They accept particular classes of problems, reason about them, perhaps request additional information, and eventually produce solutions. They perform a narrow range of reasoning functions to produce stereotypic responses to a predetermined set of situations. They are oblivious to real-time constraints on the utility of their behavior.

By contrast, human beings are versatile and flexible problem solvers that continuously adapt to the demands and opportunities presented by a dynamic environment. They encounter a great variety of unanticipated situations, decide whether and how to respond to them, and opportunistically adjust their behavior as those situations evolve. They focus attention on the most critical and most urgent aspects of the current situation and synchronize their behavior with important external events. Adaptivity figures prominently in everyday human skills, such as conducting a conversation or playing a game of tennis, as well as esoteric skills, such as monitoring critically ill medical patients or controlling a manufacturing process.

Following the model set by human intelligence, we define an *adaptive intelligent system (AIS)* as: a *knowledge-based system that reasons about and interacts with other dynamic entities in real time.* The present research involves building and experimenting with adaptive intelligent systems in particular task domains. Our goal is to develop a generic AIS architecture to support adaptive intelligent systems in a variety of task domains.

2 An Illustrative Adaptive Intelligent System: GUARDIAN

To illustrate the kind of behavior a generic AIS architecture must support, let us consider the task of monitoring patients in a surgical intensive-care unit (SICU).

Surgical intensive-care patients are critically ill individuals who require life-support devices, such as respirators or dialysis machines, to perform some of their vital functions (see Figure 1). These devices also measure certain physiological parameters. For example, parameters measured by the respirator include the *tidal volume* of air inhaled by the patient on each breath and the *peak inspiratory pressure.*

During a patient's stay in the SICU, medical staff gradually reduce the amount of life

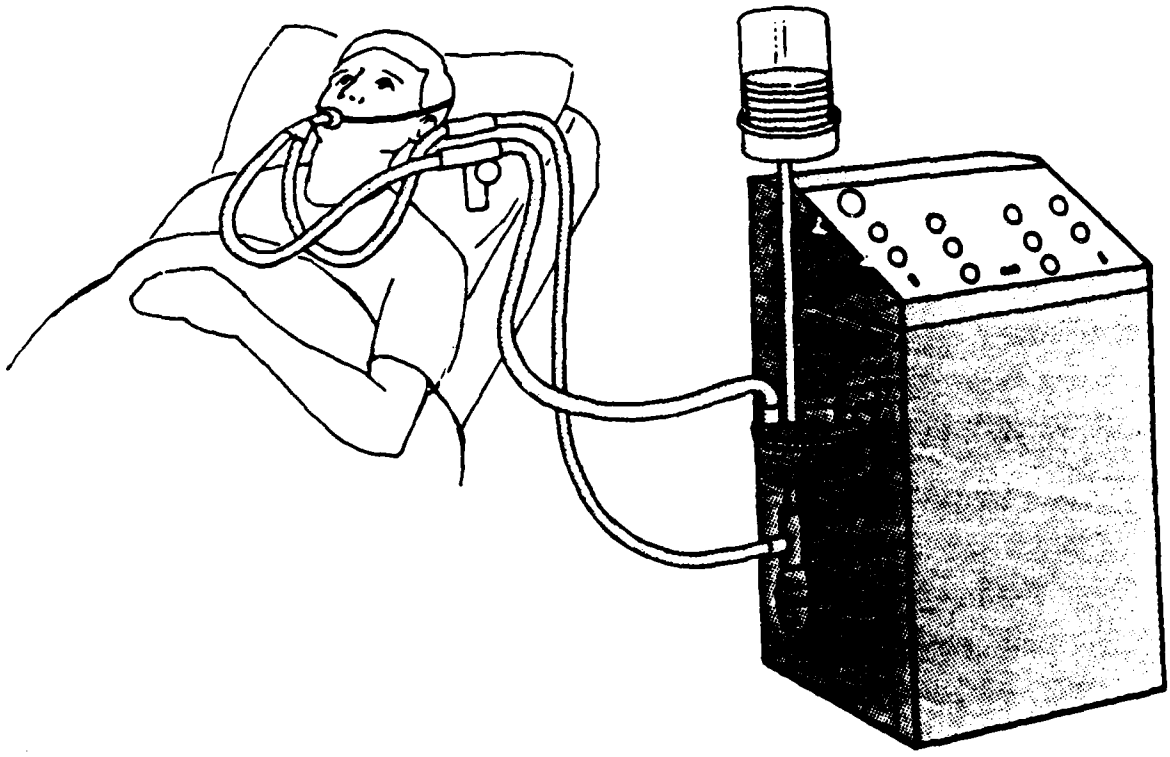


Figure 1: SICU Patient with Respirator-Assisted Breathing.

support provided to the patient and eventually withdraw life-support devices in accordance with a therapeutic plan. Along the way, the staff closely monitor and interpret available measurements of the patient's physiological function, detect deviations from expected progress, and diagnose observed signs and symptoms. If necessary, they adjust or modify the therapeutic plan and perform other therapeutic interventions.

The SICU situation typically presents great quantities of patient data and simultaneous demands for multiple interpretation, diagnosis, prognosis, and treatment activities. Because these demands exceed human cognitive capabilities, the SICU staff must selectively attend to the most important information and perform the most urgent and important activities.

GUARDIAN [14] is an experimental system for SICU patient monitoring being developed in collaboration with Dr. Adam Seiver, of the Palo Alto Veterans Administration Medical Center, and Mr. Micheal Hewett, Dr. Rattikorn Hewett, and Mr. Richard Washington, all of Stanford University. Mr. Reed Hastings and Mr. Nicholas Parlante worked on the original implementation of GUARDIAN.

For ethical and legal reasons, GUARDIAN is not intended to perform closed-loop control. That is, it will not actually change settings on life-support devices or carry out other

therapeutic interventions. However, we intend that GUARDIAN eventually will perform all of the reasoning necessary for closed-loop control and exploit that reasoning in an advisory capacity. Thus, in addition to the multiple tasks performed by SICU staff, GUARDIAN will report, summarize, and explain the SICU situation and its reasoning about that situation to physicians, nurses, and other SICU staff members.

Here is a simple scenario of the sort GUARDIAN must handle:

1. GUARDIAN is sensing several different respiratory parameters, including tidal volume and peak inspiratory pressure, many times per second. To insure that its interpretation keeps pace with the data, GUARDIAN samples sensed values of each parameter once per second and bases its interpretation of the patient's condition on the sampled values.

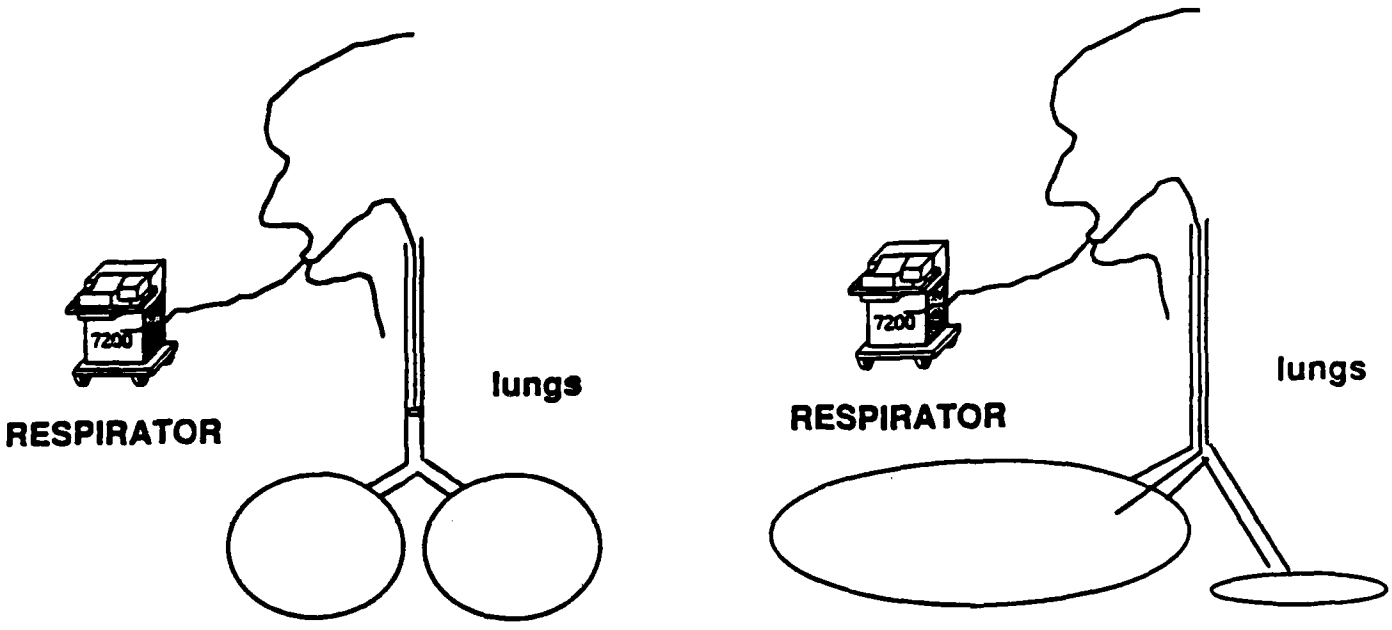
2. Following an initial interval of normal data, GUARDIAN detects an abnormal increase in peak inspiratory pressure and begins reasoning about probable causes. To allocate cognitive resources for this diagnosis task and avoid falling behind in its interpretation task, GUARDIAN reduces its sampling rate to once every ten seconds for all sensed parameters except peak pressure. It maintains its once per second sampling rate for peak pressure because that is the focus of its diagnostic reasoning.

3. GUARDIAN hypothesizes "one-sided intubation" as the most likely cause of the increase in peak inspiratory pressure. It reports this hypothesis to SICU personnel, along with a diagrammatic explanation of how one-sided intubation causes increased peak pressure (see Figure 2). GUARDIAN then recommends a corrective action, "reposition the tube," and confirms the resulting resumption of normal pressure.

4. Having completed its diagnosis, explanation, recommendation, and confirmation tasks, GUARDIAN continues to perform only its interpretation task, resuming its original once per second sampling of all respiratory parameters.

3 The Class of Adaptive Intelligent Systems

We have studied several tasks requiring an adaptive intelligent system: intensive-care monitoring, materials processing, aircraft tactical planning and control, and tutorial instruction. Despite differences among the domains of these several tasks, they share fundamental requirements for: (a) *Perception*--interpretation of sensed data to gain knowledge of other entities; (b) *Action*--controlled actuation of effectors to influence other entities; (c) *Cognition*--symbolic reasoning to draw inferences from perceptions, solve problems, make



The respirator tube is positioned correctly above the bifurcation of the endotracheal tube. It delivers a constant volume of air to the two lungs on each breath.

The respirator tube has slipped into the right main stem. The respirator must apply greater pressure to deliver the same volume of air to a single lung.

Figure 2: One-sided Intubation Causes Increased Peak Inspiratory Pressure.

decisions, and plan actions; and (d) *Attention*--allocation of computational resources among competing perceptions, actions, and cognitions in real time.

In the case of intensive-care monitoring, the system must continually sense and interpret important features of the patient's condition, diagnose observed signs and symptoms, predict the course of the patient's condition, plan appropriate therapies, control life-support device settings, take other therapeutic actions, and explain its reasoning to medical personnel. (We are investigating intensive-care monitoring in collaboration with Dr. A. Seiver and Dr. L. Fagan at Stanford University.)

In the case of intelligent processing of materials, the system must continually sense and interpret important material properties, diagnose exceptional properties, predict the impact of exceptional properties on the overall process outcome, revise the process plan to achieve process goals, control process environment parameter settings, and explain its reasoning and behavior to the operator [22]. (This project is directed by Dr. W. Pardee at Rockwell International Corporation.)

In the case of aircraft tactical planning and control, the system must continually sense and interpret important environmental circumstances, diagnose exceptional events, predict the impact of events on tactical success, revise the tactical plan to achieve tactical goals, and explain its reasoning to the pilot. (This project is directed by Dr. N.S. Sridharan at FMC Corporation.)

In the case of tutorial instruction, the system must continually sense and interpret important features of the student's learning state, diagnose errors and limitations in the student's knowledge, predict the ramifications of current learning state on subsequent tutorial activities, revise the tutorial plan to achieve tutorial goals, perform tutorial actions, and explain its reasoning and behavior [20]. (This project is directed by Dr. W. Murray at FMC Corporation.)

In each of these tasks, the AIS faces a continuing stream of demands and opportunities for potential perceptions, actions, and cognitions in real time. It generally cannot perform all potential operations in a timely fashion. Further, performing all potential operations as soon as possible is not always a system's primary objective or even a desirable one. Under certain circumstances, for example, it may be more desirable to perform only operations that meet a specified criterion or to delay performance of certain operations until specified preconditions occur. While more efficient algorithms or faster computers may enable some application systems to achieve particular real-time objectives, they will not solve the general problem of limited resources or obviate its concomitant resource-allocation requirement. For a computer of any speed, we can define tasks whose computational demands--for multiplicity of operations, computational complexity of operations, temporal responsiveness, and synchronization--exceed its computational resources. For these reasons, we view attentional power and flexibility, rather than speed *per se*, as the primary scientific challenge in developing a generic architecture for adaptive intelligent systems.

4 Generic Requirements for Adaptive Intelligent Systems

More specifically, adaptive intelligent systems functioning in diverse task environments share generic requirements for: cognitive versatility, interaction with a dynamic environment, management of complexity, and real-time performance. We discuss each of these different categories of requirements below, with illustrations from GUARDIAN's patient-monitoring task.

4.1 Cognitive Versatility.

Multi-Faceted Expertise. An AIS must perform multiple reasoning tasks, involving different problems, problem domains, and problem-solving methods. For example, GUARDIAN must know how to interpret patient data, diagnose observed signs and symptoms, and plan appropriate therapeutic actions. It must know how to perform these tasks for different biological systems, such as the respiratory system and the circulatory system. It must know, for example, how to diagnose observed signs probabilistically, using a belief network, as well as from first principles, using explicit models of system structure and function.

Concurrent Reasoning Activities. An AIS must simultaneously conduct multiple reasoning activities. For example, GUARDIAN must continue to interpret newly perceived patient parameters while diagnosing an observed sign so that it can incorporate interpretations of relevant data into its diagnostic reasoning and notice when other incidental data suggest other situations requiring its attention.

Incremental Reasoning. An AIS must reason incrementally about situations observable over time. For example, in the SICU, a great variety of patient data occur asynchronously over a long period of time. GUARDIAN must perceive and integrate relevant data, as they occur, to form a coherent, "up-to-the-minute" model of the patient's dynamic condition.

Explanation. An AIS must explain its knowledge, reasoning, and behavior. As indicated above, GUARDIAN will not act directly upon the patient or the ventilator. It can only advise SICU staff. To maximize the utility of its advice, GUARDIAN must justify its recommendations as well as it can.

4.2 Interaction with a Dynamic Environment.

Functional Asynchrony and Parallelism. An AIS must perceive, reason, and act asynchronously and in parallel. For example, GUARDIAN cannot ignore a patient, whose condition can change at any time, while interpreting previously perceived patient data. It must perceive important new patient data when they occur. Similarly, GUARDIAN must perform planned actions at appropriate times regardless of the amount of non-relevant perception and cognition it performs during overlapping intervals.

Continuous Operation.

An AIS must function continuously over long time intervals. For example, a practical version of GUARDIAN would operate continuously over periods of several weeks,

accumulating patient information, building an increasingly complete and accurate patient model, and recommending therapeutic actions tailored to the patient's evolving condition.

Functional Integration. An AIS must integrate perception, action, and cognition within a coherent point of view. For example, GUARDIAN's model of a patient's condition must incorporate its perceptions of the patient's physiological parameters and its interpretation must influence its therapeutic actions. In some cases, GUARDIAN's patient model also should influence its perceptions of particular physiological parameters. For example, since post-operative patients typically have lowered body temperatures, GUARDIAN should adjust its perception of "normal," "high," and "low" temperatures accordingly.

4.3 Management of Complexity.

Selective Attention. An AIS must differentially process sensed data in accordance with cognitive objectives and the external situation. The SICU situation presents vast quantities of data, more than a human being or GUARDIAN could interpret in real time. It must choose among them. If, for example, GUARDIAN perceives that the patient is suffering from hypocapnia (low CO2 in the blood), it should focus on patient data relevant to its diagnosis of that problem (e.g., blood gases, temperature, respiration rate, tidal volume). It should temporarily ignore extraneous patient data whose interpretation would distract it from solving the problem at hand. At the same time, however, GUARDIAN must remain sensitive to the possibility that extraneous data might signal a new emergency.

Automatic Performance. An AIS must perform potentially complex actions without impeding ongoing perception and cognition. For example, given a decision to report recent patient data, GUARDIAN should be able to select, format, and display those data, while continuing to perceive and interpret new patient data, diagnose new signs and symptoms, plan or replan therapeutic or other actions, and carry out unrelated actions in an appropriate and timely fashion. At the same time, it must be prepared to interrupt its reporting to perform a more important competing action, such as to alert SICU staff to a newly observed patient sign.

Focused Reasoning An AIS must dynamically control its reasoning in accordance with the current situation and its strategic objectives. The SICU presents many more "problems" than GUARDIAN could solve in real time. It must choose among them. For example, if GUARDIAN has decided to explore alternative diagnoses for observed hypocapnia, it should go about that task in a deliberate fashion without being distracted unecessarily by other potential reasoning activities--e.g., reviewing the long-range therapeutic plan or preparing the day's

summary report. At the same time, it must be prepared to interrupt its diagnostic reasoning to attend to a more serious emergency, should one arise.

4.4 Real-Time Performance.

Guaranteed Inter-Operation Latencies. An AIS must guarantee that it will begin successive reasoning operations after a specified absolute or relative latency. Conversely, an AIS must not allocate its cognitive resources to uninterruptable processes of arbitrary durations. All real-time performance rests ultimately on an AIS's ability to redirect its cognitive resources appropriately and quickly in response to a dynamic situation. The minimum inter-operation latency required of an AIS depends upon its domain. For example, GUARDIAN must guarantee that it will sound an alarm within a few seconds after perceiving that a patient has stopped breathing.

Time-Stress Responsivity. An AIS must respond to increased time stress by reducing its response latency. For example, if GUARDIAN is diagnosing a slight reduction in the patient's tidal volume (amount of air per breath), a few extra minutes of elapsed time will not affect the utility of its diagnosis. On the other hand, if GUARDIAN is diagnosing a complete interruption of tidal volume (the patient is not breathing), it must complete the diagnosis, as well as recommend corrective action, within four minutes, to help save the patient's life.

Graceful Degradation. An AIS must reduce response latency, in accordance with increased time stress, by gradually compromising the quality of its performance. For example, GUARDIAN could reduce diagnosis time by exploring only the most likely possibilities and thereby reducing the certainty of its conclusions. It could further reduce diagnosis time by exploring a smaller subset of the possibilities and further reducing the certainty of its conclusions.

Speed-Knowledge Independence. An AIS must produce stable response latencies despite increases in knowledge. For example, GUARDIAN must continue to guarantee a stable diagnosis time for exploring a given subset of possibilities, even as it acquires knowledge of many other diseases. In fact, relevant new knowledge should have the potential to speed up GUARDIAN's performance.

5 A Generic AIS Architecture

Figure 3 illustrates a generic AIS architecture that addresses the above requirements. This section describes the elements of the AIS architecture--a *dynamic control architecture*, an *asynchronous I/O subsystem*, *dynamic I/O channels*, and a *satisficing reasoning cycle*--and illustrates them with examples from the GUARDIAN system introduced above.

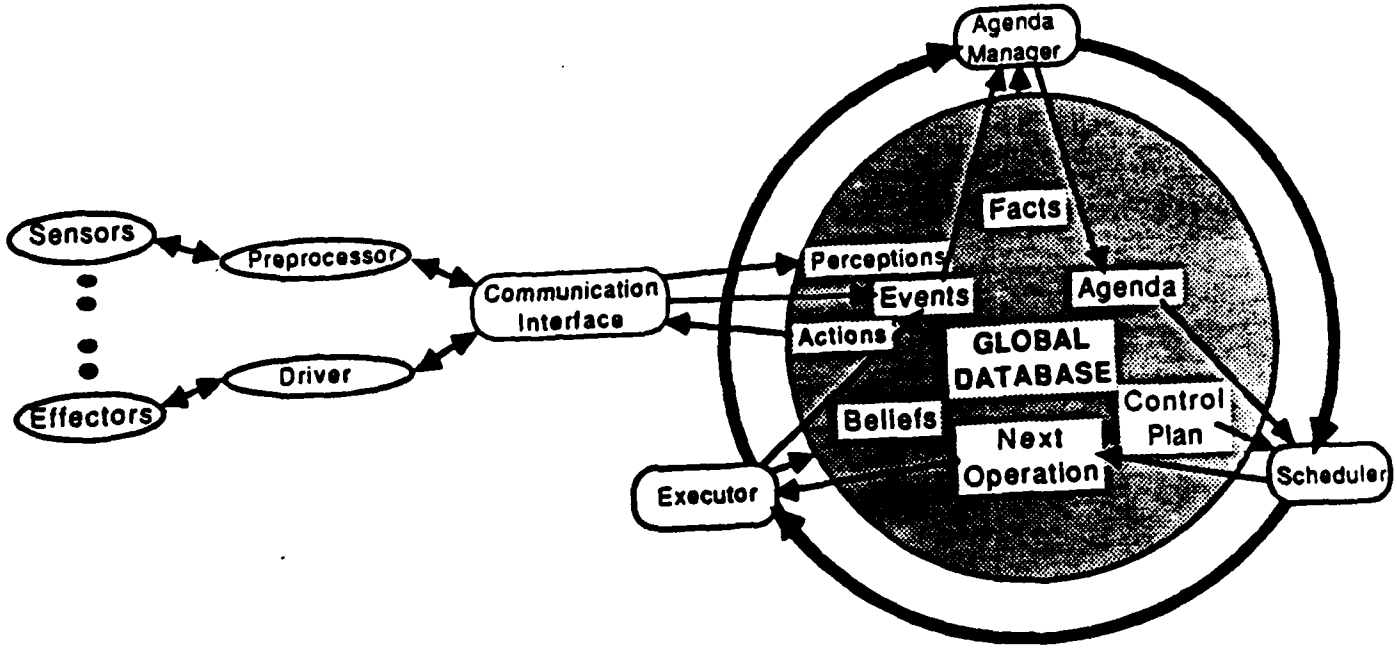


Figure 3: The Generic AIS Architecture.

5.1 A Dynamic Control Architecture

The rightmost section of Figure 3 schematizes the *dynamic control architecture* underlying the proposed AIS architecture. The architecture is implemented as the BB1 system [10, 16]) and we will use the terms "dynamic control architecture" and BB1 interchangeably. Cognitive operations take place in the context of a *global database* that contains all of the facts, beliefs, events, plans, etc. known to the system. The architecture iterates a three-step reasoning cycle. First, the *agenda manager* produces an agenda of reasoning operations suggested by recent cognitive events. Then, the *scheduler* chooses as the next operation the one that best serves the current control plan. Finally, the *executor* executes the chosen operation, changing information in the global database and recording a corresponding cognitive event.

The BB1 knowledge base is implemented in the BB* conceptual network representation [13]

BB* provides subnetworks representing architecturally defined entities, such as actions, events, control plans, and cognitive skills. It also provides an editor for building subnetworks representing application-specific strategies factual knowledge and cognitive skills (discussed below). For example, GUARDIAN's factual knowledge covers aspects of normal and abnormal anatomy and physiology, probable causes of certain signs and symptoms, and the normal and abnormal structure and function of generic flow and exchange systems. Figure 4 excerpts GUARDIAN's knowledge of normal respiratory anatomy and physiology.

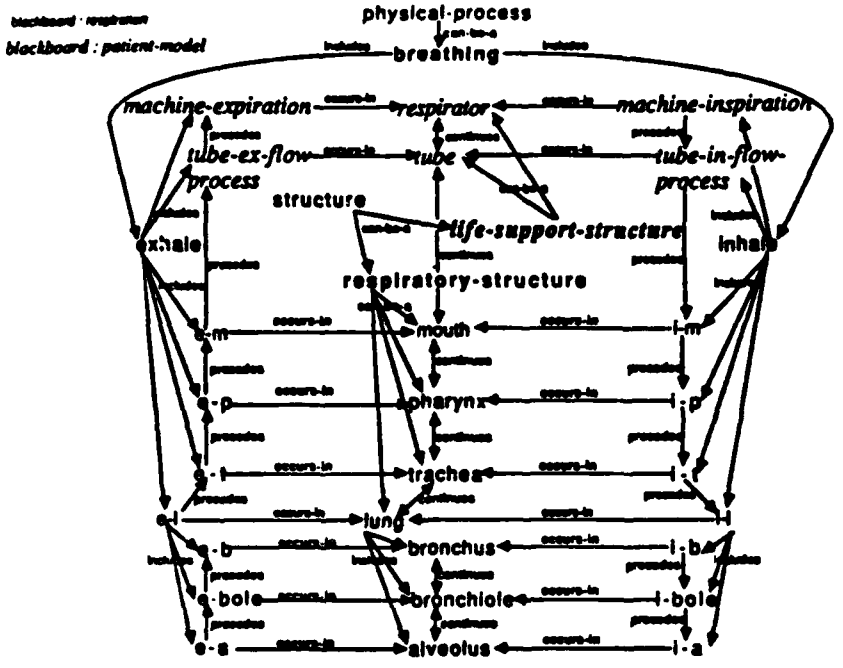


Figure 4: Excerpt of GUARDIAN's Knowledge of Respiratory Anatomy and Physiology.

The dynamic control architecture provides a general-purpose reasoning environment that can support the multiple reasoning activities required of a typical AIS. For example, BB1 has been used to build systems for design [12, 26], planning and plan recognition [3, 9, 20], signal interpretation [2, 5], explanation [24], and analogical inference [4, 18]. Moreover, BB1 allows an AIS to incorporate several cognitive skills. For example, GUARDIAN currently incorporates skills for: classification of perceptual observations into temporal episdoes of known semantic categories (e.g., normal, high, very high); diagnosis of observed signs based on belief networks; diagnosis and explanation of observed signs based on generic system models [15]. Finally, BB1 allows an AIS to perform multiple tasks concurrently by interleaving their constituent operations. For example, GUARDIAN typically continues to classify newly sensed data while

diagnosing previously observed signs. In fact, if GUARDIAN happens to classify new data relevant to an ongoing diagnosis, it incorporates those results in its diagnostic reasoning.

The dynamic control architecture provides the strategic control required of an AIS. In general, an AIS must balance efforts to: (a) respond promptly to urgent situations; and (b) plan effective patterns of future behavior. BB1 supports this range of behavior by enabling a system to incrementally construct and modify explicit plans for its own behavior. These control plans may be short-term or long-term, abstract or specific. The system may augment or modify its plan at any time. On each cycle, the scheduler chooses an operation that best matches the current control plan. Thus, the system always behaves in accordance with plans it has previously constructed. Whenever the scheduler chooses operations that change the plan, the system's subsequent behavior changes accordingly.

For example, Figure 5 shows GUARDIAN's control reasoning for the scenario described in section 2 above. The horizontal dimension in Figure 5 represents scenario time, partitioned into units corresponding to reasoning cycles. The top panel of the figure shows the dynamic control plan GUARDIAN constructs during the scenario. The middle panel shows the agenda of potential operations on each cycle. The bottom panel shows the operations GUARDIAN chooses to execute on each cycle. The actual results of GUARDIAN's reasoning (e.g., its data classifications, diagnostic conclusions, and diagnostic explanations) appear elsewhere in the knowledge base. The episode and associated control reasoning unfold, left to right, as follows:

1. At the start of the episode, GUARDIAN is following a long-term plan to monitor all respiratory parameters, sampling each one once per second. On subsequent cycles, it ignores potential operations of other types (symbolized as K, J, L) and executes monitoring operations (symbolized as M).

2. Upon observing an abnormal increase in peak inspiratory pressure, GUARDIAN decides to correct this problem. To free up computational resources for this task, it decides to reduce its sampling of all respiratory parameters to once every ten seconds, except for pressure, which it continues to sample once per second. On subsequent cycles, GUARDIAN occasionally interleves these types of operations (symbolized as M and P, respectively) with its reasoning about the elevated peak pressure.

3. GUARDIAN begins its effort to correct the elevated pressure by deciding to diagnose it. On subsequent cycles, GUARDIAN executes diagnostic operations (symbolized as D) and eventually hypothesizes that the problem is one-sided intubation.

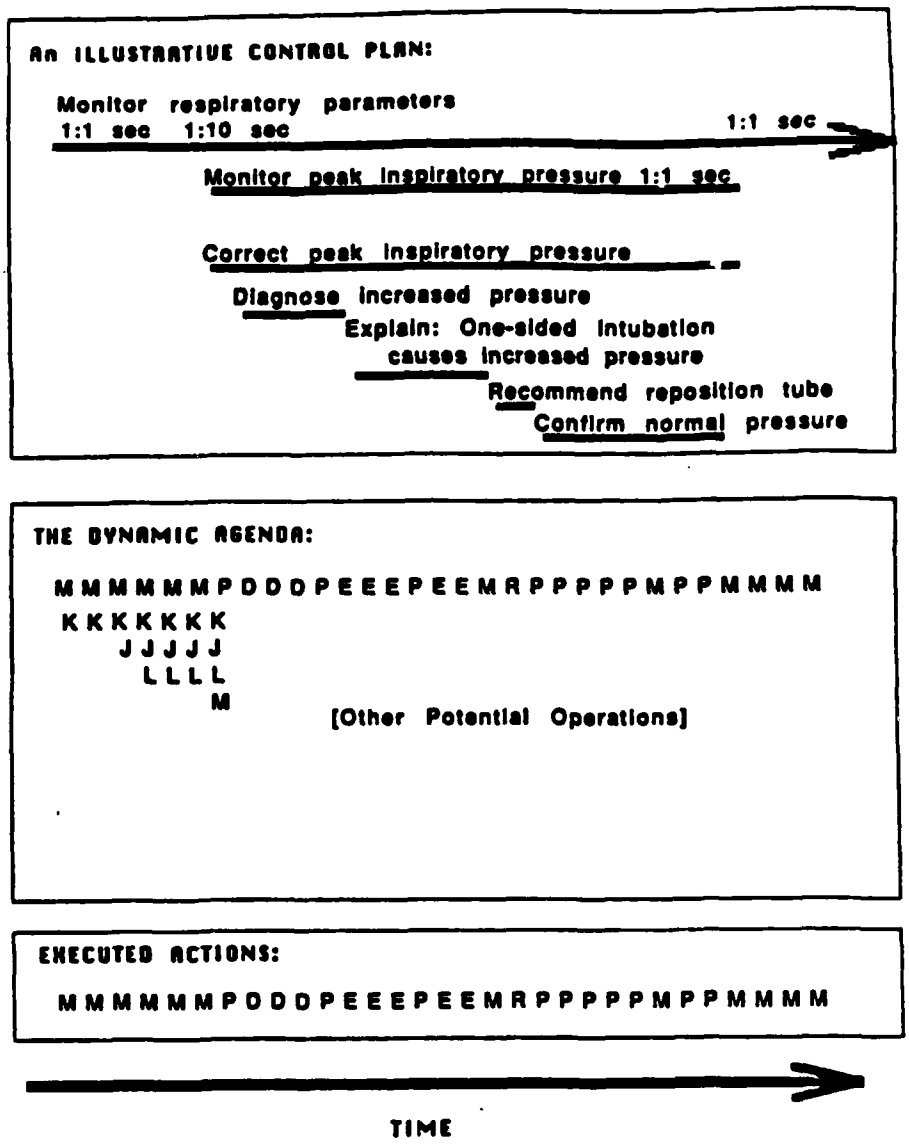


Figure 5: Excerpt from a Simple Control Plan for GUARDIAN.

3. Having completed the diagnosis, GUARDIAN decides to explain how one-sided intubation would cause elevated peak pressure. On subsequent cycles, it executes corresponding explanation operations (symbolized as E), producing the diagrammatic explanation in Figure 2 above.

4. Having completed the explanation, GUARDIAN decides to recommend a corrective action, repositioning of the tube. On the next cycle, it makes that recommendation (symbolized as R).

5. Having made its recommendation, GUARDIAN decides to monitor peak pressure in order to confirm that the tube has been repositioned and peak pressure has returned to normal. On subsequence cycles, it executes corresponding monitoring actions (symbolized as P).

6. Finally, having completed all tasks necessary to correct the abnormal peak pressure, GUARDIAN continues only its monitoring activities, resuming its original once per second sampling of all respiratory parameters.

5.2 An Asynchronous I/O Subsystem

To support an integrated approach to perception, action, and cognition the AIS architecture extends the dynamic control architecture with an *asynchronous I/O subsystem* comprising *logical I/O buffers* and a *communications interface (CI)* [17], as illustrated in Figure 3.

To integrate perception with cognition, the CI continuously monitors physical streams from remote sensors and records perceptual events representing sensed data in appropriate logical input buffers. In the case of GUARDIAN, for example, the CI monitors streams from sensors attached to the respirator. The agenda-manager uses perceptual events, along with cognitive events, to update the agenda of potential reasoning operations, thereby introducing them into the reasoning process. Since input buffers are part of the global database, it is also possible for other reasoning operations to inspect them at any time. Input buffers have fixed capacities, with first-in-first-out (FIFO) overflow. Thus, if the CI relays newly sensed data faster than the reasoning system can use them, the reasoning system "forgets" old perceptions, rather than falling increasingly behind.

To integrate action with cognition, the CI continuously monitors logical output buffers for intended actions placed there by reasoning operations. The CI relays intended actions to physical streams for appropriate remote effectors. In the case of GUARDIAN, for example, the CI monitors buffers associated with various display devices. Output buffers have fixed capacities, with FIFO overflow. Thus, in the unlikely case that the reasoning system places intended actions into the buffer faster than the CI can service them, the system "forgets" old intended actions, rather than falling increasingly behind.

The CI can run either as a background process on the host machine or on a separate machine connected to the host by Ethernet, providing concurrent and asynchronous perception, action, and cognition. The AIS architecture need not complete an entire reasoning cycle before noticing intervening perceptions or executing intended actions. It gives reasoning operations immediate access to new perceptions and it immediately executes intended actions determined by reasoning operations.

5.3 Dynamic I/O Channels

Given real-time constraints on the utility of its behavior, an AIS must manage the computational complexity of perception and action. The environment in which an AIS operates continuously bombards its sensors with data that are more or less relevant to its current task. Interpreting these data and selecting those that are relevant or otherwise interesting is computationally intensive. Attempting to process all such data in real time could easily swamp the reasoning mechanism. Similarly, although an AIS has discretion over which actions to initiate, correctly executing actions--especially those that need to be coordinated with external events--is computationally intensive. Attempting to control the execution of several complex action programs in real time could easily swamp the reasoning mechanism. Finally, reasoning operations are, themselves, computationally intensive. Attempting to reason effectively could easily distract an AIS from perception of important events or timely performance of important actions.

To facilitate management of the computational complexity of real-time perception and action, the AIS architecture incorporates *dynamic I/O channels*. As illustrated in Figure 3, each channel comprises one or more processes that mediate communications between an AIS's remote sensors and effectors and its reasoning mechanism. To provide selective attention, *preprocessors* continuously interpret and filter asynchronously arriving sensor data in accordance with their current *perceptual filters*. Preprocessors relay only task-relevant or otherwise important data to the reasoning system. To provide automatic performance, *drivers* filter and interpret asynchronously arriving actions in accordance with their current *performance filters*. They give priority to the most important and urgent actions and handle all details of action execution, including synchronization with external events. Both preprocessors and drivers apply filters determined by reasoning operations and sent to them via the communications interface.

Of course, the effectiveness of dynamic I/O channels depends on the effectiveness of the filters they apply. Our approach builds directly on the dynamic control architecture and its I/O subsystem. First, we use activity in I/O buffers as an indicator of the dynamic balance between reasoning and perception. When overflow of input buffers indicates that the system's current reasoning activities cannot keep pace with its current perceptual activities, stronger perceptual filters reduce the rate of perceptual input. When underflow of input buffers indicates that the system's current perceptual activities are underutilizing its reasoning capacity, weaker perceptual filters increase the rate of perceptual input. GUARDIAN currently exploits this mechanism.

Second, we plan to use changes in the control plan to signal changes in attentional focus and the control plan itself to characterize attentional focus. For example, having decided to monitor peak inspiratory pressure more closely than other respiratory parameters (see Figure 5), GUARDIAN should send perceptual filters favoring sensor data representing that parameter over other respiratory parameters. Both of these mechanisms enable an AIS to dynamically adapt its perceptual activities to current reasoning activities in order to balance overall resource utilization.

5.4 A Satisficing Reasoning Cycle

An AIS must satisfy real-time constraints on its performance--that is, it must perform the right operations at the right times. The AIS architecture in Figure 3 provides a good foundation for real-time performance, but it is vulnerable to open-ended computation times in each step of its reasoning cycle. In particular, the AIS architecture cannot rely upon the "best-next" version of this cycle, in which each step successively acquires control of the processor and runs to normal termination. We call this the "best-next" version because, on each cycle, it identifies, schedules, and executes the best available operation. A system that uses the best-next cycle will fail to perform important operations in a timely fashion whenever it happens to have begun execution of a time-consuming instance of one of its three steps at a critical time. Because most AI architectures use the best-next reasoning cycle, its vulnerabilities are well understood. However, efforts to address these vulnerabilities focus entirely on improving cycle speed through the use of efficient matching algorithms, parallelism, or compilation [7, 8, 19, 21]. In aiming to place an "acceptable" upper bound on computation time for each step of the reasoning cycle, these approaches produce special-purpose solutions to limited classes of AIS applications. They will not work for systems that exceed their specifications for knowledge base size, response latency, or synchronization with external entities. Thus, they ignore the fundamental challenge of real-time computation: to guarantee a dynamically specifiable maximum latency of operations.

We are developing a *satisficing reasoning cycle* to provide the guaranteed latencies required for real-time performance. By "latency," we mean the elapsed time prior to beginning each successive operation. Figure 6 illustrates one such cycle, which differs from the traditional best-next cycle in each of its three steps. First, instead of exhaustively identifying all possible operations on each cycle, the agenda manager identifies as many operations as it can, best first, until any of its dynamic interrupt conditions occurs. Second, instead of choosing the optimal operation from a complete agenda on each cycle, the scheduler does the best it can with an

incomplete agenda. Third, instead of exhaustively executing the scheduled operation on each cycle, the executor partially executes the operation until any of its dynamic interrupt conditions occurs. Upon interruption, the executor saves the state of the interrupted operation in a form suitable for resumption and places a pointer to the ready-to-resume operation on the agenda. Depending upon subsequent events, the scheduler may or may not choose to resume execution of the interrupted operation on a subsequent cycle. All three steps in the satisficing cycle operate in accordance with dynamic *cycle parameters*, determined by reasoning operations.

```
1. Update the agenda of potential operations best first until:
        (a) a criterial operation is identified; or
        (b) a criterial event occurs; or
        (c) the agenda updating deadline occurs; or
        (d) adenda updating terminates.

2. Schedule the best criterial executable operation until;
        (a) a criterial event occurs; or
        (b) scheduling terminates.

3. Execute the scheduled operation until:
        (a) a criterial event occurs; or
        (b) the interpretation deadline occurs.
        (c) execution terminates.
```

Figure 6: Satisficing Cycle.

Again, the effectiveness of the satisficing cycle depends upon the effectiveness of the cycle parameters it obeys and our approach builds directly on the dynamic control architecture and its I/O subsystem. First, as discussed above, we use activity in I/O buffers as an indicator of the dynamic balance between reasoning and perception. When overflow of input buffers indicates that reasoning activities cannot keep pace with perceptual activities, stricter cycle parameters will decrease the time spent on agenda management and select a smaller number of reasoning operations for execution. When underflow of input buffers indicates that current perceptual activities underutilize reasoning capacity, more lenient cycle parameters will increase the time spent on agenda management and select a larger number of reasoning operations for execution. Second, as discussed above, we plan to use changes in the control plan to signal changes in attentional focus and the control plan itself to characterize attentional focus. For example, having decided to diagnose an observed increase in peak inspiratory pressure (see Figure 5, GUARDIAN should adopt cycle parameters favoring these kinds of operations over other kinds of operations. Both of these mechanisms enable an AIS to dynamically adapt its reasoning activities to current perceptual activities in order to balance overall resource utilization.

In contrast to best-next cycles, which invariably identify and perform the best possible operation regardless of temporal considerations, satisficing cycles enable systems to realize, combine, and alternate among different real-time reasoning policies, such as:

- Perform any operation that is "good enough" as soon as possible.
- Perform any urgent operation immediately.
- Perform the "best available" operation whenever necessary.
- Perform only operations that are "good enough."
- Perform the best possible operation regardless of the time required.

On the other hand, satisficing cycles make systems vulnerable to errors that do not occur under conventional best-next reasoning cycles. By definition, satisficing cycles allow systems to perform sub-optimal operations. In extreme cases, a system could decide prematurely to perform costly or ineffective operations or fail to notice highly desirable operations that are well within its capabilities. However, if we wish to build powerful systems that function well in dynamic environments, we must forego optimality in favor of effective management of complexity [25]. Allowing the possibility of error is one concession we can make toward this end. Formulating execution-cycle algorithms that meet the performance requirements of adaptive intelligent systems while minimizing the impact of errors is a primary objective of our research.

6 Satisfaction of AIS Requirements

Table 1 summarizes the relationships between the generic requirements for an AIS set forth in section 4 and the architectural components proposed in section 5. Let us briefly review these relationships.

6.1 Cognitive Versatility

Multi-Faceted Expertise. The dynamic control architecture provides a general reasoning framework and knowledge representation scheme. It can integrate knowledge of multiple problem classes, multiple problem-solving methods and multiple domains of factual knowledge.

Concurrent Reasoning Activities. The dynamic control architecture formulates reasoning as a sequence of discrete cognitive operations that incrementally generate and modify explicit solution representations. It can interleave component operations for concurrent reasoning activities.

Incremental Reasoning. The dynamic control architecture formulates reasoning as a sequence

Table 1: Relationships between AIS Requirements and the Proposed AIS Architecture.

	Dynamic Control Architecture	Asynchronous I/O Subsystem	Dynamic I/O Channels	Satisficing Cycle
COGNITIVE VERSATILITY				
Multi-Faceted Expertise	X			
Concurrent Reasoning Activities	X			
Incremental Reasoning	X			
Explanation	X			
INTERACTION WITH A DYNAMIC ENVIRONMENT				
Functional Asynchrony and Parallelism	X	X	X	X
Continuous Operation	X	X	X	X
Functional Integration	X	X	X	X
MANAGEMENT OF COMPLEXITY				
Selective Attention	X		X	
Automatic Performance	X		X	
Focused Reasoning	X			X
Real-Time Performance				
Guaranteed Inter-Operation Latencies	X			X
Time-Stress Responsivity	X		X	X
Graceful Degradation	X		X	X
Speed-Knowledge Independence	X			X

of discrete cognitive operations that incrementally generate and modify explicit solution representations. It can incorporate information about dynamic external situations as that information becomes available.

Explanation. The dynamic control architecture allows a system to generate and record explicit control plans, which it uses to determine its subsequent actions. It also can use these plans retrospectively to explain its actions and their consequences.

6.2 Interaction with a Dynamic Environment

Functional Asynchrony and Parallelism. The AIS architecture allocates independent processes for the cognitive system, the communications interface, and individual I/O channels, sensors, and effectors. It thus supports asynchronous and parallel perception, action, and cognition.

Continuous Operation. The AIS architecture conceptualizes each functional component (the cognitive system, the communications interface, and individual I/O channels, sensors, and effectors) as a cyclical, non-terminating process. This approach orients a system away from the traditional goal-directed problem solving and toward continuous operation.

Functional Integration. The dynamic control architecture defines perception and action buffers as standard data structures within its global knowledge base. Its I/O subsystem automatically transfers perceptions and actions between those buffers and appropriate I/O channels. Its satisficing cycle treats perception and action events in the same fashion as internally generated cognitive events.

6.3 Management of Complexity.

Selective Attention. The dynamic control architecture provides an explicit representation of a system's own control decisions and cognitive state. Dynamic I/O channels allow the system to use that knowledge to instruct perceptual preprocessors to transform and filter sensed data accordingly before relaying them to perceptual buffers in the knowledge base.

Automatic Performance. The dynamic control architecture allows a system to determine intended actions at an abstract level. Dynamic I/O channels allow the system to "download" computations for controlling the execution of those actions to action drivers.

Focused Reasoning. The dynamic control architecture allows a system to decide what kinds of problems it prefers to address, what kinds of reasoning operations it prefers to perform, and what kinds of knowledge it prefers to apply. The satisficing cycle uses these preferences to identify and schedule potential reasoning operations.

6.4 Real-Time Performance.

Guaranteed Inter-Operation Latencies. The dynamic control architecture allows a system to decide what kinds of operations it prefers to perform and what kinds of events require immediate attention. The satisficing cycle uses these criteria to focus, limit, and interrupt processing between successive operations.

Time-Stress Responsivity. The architecture allows a system to respond to time stress in several ways. First, the dynamic control architecture allows the system to modify its reasoning strategy to focus on urgent reasoning tasks and efficient reasoning methods. Second, the dynamic control architecture allows the system to modify its preferences and interrupt conditions, so that the satisficing reduces the amount of processing between successive operations. Third, dynamic I/O channels allow the system to adopt stricter perceptual filters to reduce the amount of sensed data relayed to and processed by the cognitive system.

Graceful Degradation. All of the above responses to time stress permit graceful degradation. First, the dynamic control architecture allows the system to postpone or discontinue individual reasoning tasks individually as required by the situation. It also permits the system to choose among alternative reasoning methods that vary in efficiency and quality of results. Second, the dynamic control architecture allows the system to modify its preferences and interrupt conditions qualitatively and quantitatively, so that the satisficing cycle can reduce inter-operation processing--with associated reductions in quality of performance--by variable amounts. Third, dyanmic I/O channels allow the system to vary perceptual filters qualitatively and quantitatively to reduce the relay of sensed data by variable amounts.

Speed-Knowledge Independence. The dynamic control architecture allows a system to decide what kinds of operations it prefers to perform, what kinds of knowledge it prefers to apply, and what kinds of events should interrupt its search for operations and knowledge. The satisficing cycle enforces these preferences, regardless of the system's total amount of knowledge.

7 On Architectures for Intelligence

In the present context, a volume on "Architectures for Intelligence," we may ask: Is the AIS architecture a theoretical contribution to our understanding of intelligence? The answer to this question depends upon what we mean by "intelligence," for example: (a) human intelligence; (b) lower forms of biological intelligence; or (c) abstract concepts of intelligence. In principle, each of these ideals can be specified further as a distinctive, although possibly overlapping, set of component functions. For example, symbolic reasoning is prominent in human intelligence, while sensory-motor adaptation is more prominent in lower forms of biological intelligence. Abstract concepts of intelligence vary widely, but a large number of them favor rational decision making. In practice, functional specifications of intelligence are, themselves, objects of research and considerable debate in fields such as psychology, biology, and decision analysis.

For these reasons, to evaluate a given architecture, we must evaluate the stated functional objectives as well as the architecture's achievement of those objectives.

The AIS architecture is directed toward an evolving abstract definition of intelligence. That definition is motivated by important and challenging computational tasks and it is inspired by human intelligence as the driving metaphor. Thus, we have tried to show that our functional definition of adaptive intelligent systems is required for effective computational performance for an important class of tasks. And we would argue that our definition embodies the functionality that enables human beings to perform these tasks. Indeed, we have shown that the dynamic control architecture, which is the foundation for the AIS architecture, effectively models the details of human problem-solving protocols for an everyday planning task [11]. Similarly, the architecture's use of dynamic I/O channels to achieve selective attention and automatic performance corresponds roughly to biological and information-processing models of human behavior [1, 6, 23, 27].

At the same time our definition obviously ignores many equally important elements of human intelligence, such as those related to: sensory-motor performance, a large dynamic memory, linguistic capabilities, analogue processing capabilities, and the emotional and motivational forces affecting human behavior. Moreover, given the metaphorical role of human intelligence in our research, we make no claim to model the actual psychological or biological mechanisms underlying any of the specified functionality. Nor do we claim that it is the only mechanism that can produce the specified functionality. At this stage in our research, we prefer to evaluate the AIS architecture in terms of its sufficiency to produce the specified functionality and its resulting adequacy to support a variety of adaptive intelligent systems. We reserve judgment regarding the architecture's applicability to the substantially greater scope of human intelligence.

References

[1] Broadbent, D.E.
Perception and communication.
London: Pergamon, 1958.

[2] Brugge, J., and Buchanan, B.G.
The ABC System.
Technical Report, Stanford, Ca.: Stanford University, 1987.

[3] Darwiche, A., Levitt, R., and Hayes-Roth, B.
Planning construction tasks from structural designs.
Technical Report, Stanford University, 1988.

[4] Daube, F., and Hayes-Roth, B.
A case-based approach to beam design.
In *Proceedings of the AAAI88 Workshop on Case-Based Reasoning.* 1988.

[5] Delaney, J.
BB1-AIRTRAC.
Technical Report, Stanford University, 1987.

[6] Fitts, P.M.
Perceptual-motor skill learning.
In *Categories of human learning,* . New York: Academic Press, 1964.

[7] Forgy, C.L.
RETE: A fast algorithm for the many pattern/many object pattern matching problem.
Artificial Intelligence 19:17-37, 1982.

[8] Gupta, A., Forgy, C., and Newell, A.
High-speed implementations of rule-based systems.
Technical Report, Carnegie-Mellon University, 1987.

[9] Harvey, J., and Hayes-Roth, B.
WATCH: Inductive abstraction of control knowledge.
Technical Report, Stanford, Ca.: Stanford University, 1987.

[10] Hayes-Roth, B. and Hewett, M.
Building systems in the BB1 architecture.
In R. Engelmore and A. Morgan (editor), *Blackboard Systems,* . London: Addison-Wesley, 1988.

[11] Hayes-Roth, B., and Hayes-Roth, F.
A cognitive model of planning.
Cognitive Science 3:275-310, 1979.

[12] Hayes-Roth, B., Buchanan, B.G., Lichtarge, O., Hewett, M., Altman, R., Brinkley, J., Cornelius, C., Duncan, B., and Jardetzky, O.
PROTEAN: Deriving protein structure from constraints.
Proceedings of the AAAI , 1986.

[13] Hayes-Roth, B., Garvey, A., Johnson, M.V., and Hewett, M.
A layered environment for reasoning about action.
Technical Report KSL-86-38, Stanford, Ca.: Stanford University, 1986.

[14] Hayes-Roth, B., Hewett, M., Hewett, R., Washington, R., Hastings, R., Parlante, N., and Seiver, A.
Monitoring Surgical Intensive Care Patients: A case-study in adaptive intelligent systems.
Technical Report, Stanford, Ca.: Stanford University, 1988.

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

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

[17] Hewett, M., and Hayes-Roth, B.
Real-Time I/O in Knowledge-Based Systems.
In *Proceedings of the AAAI88 Workshop on Blackboard Systems.* 1988.

[18] Johnson,M.V., and Hayes-Roth, B.
Learning to Solve Problems by Analogy.
Technical Report KSL-88-01, Stanford, CA: Stanford University, 1988.

[19] Miranker, D.P.
Performance estimates for the DADO machine: A comparison of treat and rete.
Proceedings of Fifth Generation Computer Systems , 1984.

[20] Murray, W. R.
Dynamic instructional planning in the BBI Blackboard Architecture.
Technical Report, FMC Corporation Artificial Intelligence Center, 1988.

[21] Oflazer, K.
Parallel execution of production systems.
Proceedings of the IEEE International Conference on Parallel Processing , 1984.

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

[23] Posner, M.I.
Chronometric explorations of mind.
Hillsdale, N.J.: Lawrence Erlbaum Associates, 1978.

[24] Schulman, R., and Hayes-Roth, B.
Plan-Based Construction of Strategic Explanations.
In *Proceedings of the AAAI88 Workshop on Explanation.* 1988.

[25] Simon, H.A.
The Sciences of the Artificial.
Cambridge, Ma.: The M.I.T. Press, 1969.

[26] Tommelein, I. D., Levitt, R.E., and Hayes-Roth, B.
Using Expert Systems for the Layout of Temporary Facilities on Construction Sites.
CIB W-65 Symposium, Organization and Management of Construction, Birkshire, U.K , 1987.

[27] Triesman, A.
Strategies and models of selective attention.
Psychological Review 76:282-299, 1969.