Information Sciences 117 (1999) 1–18

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

The engineering of mind ☆

James S. Albus [1]

Intelligent Systems Division, National Institute of Standards and Technology, Gaithersburg, MD 20895, USA

Received 1 September 1998; accepted 1 November 1998

Abstract

While the mind remains a mysterious and inaccessible phenomenon, many of the components of mind, such as perception, behavior generation, knowledge representation, value judgment, reason, intention, emotion, memory, imagination, recognition, learning, attention, and intelligence are becoming well defined and amenable to analysis. Progress is rapid in the cognitive and neurosciences as well as in artificial intelligence, control theory, and many other fields related to the engineering of mind. A reference model architecture for intelligent systems is suggested to tie together concepts from all these separate fields into a unified framework that includes both biological and machine embodiments of the components of mind. It is argued that such a reference model architecture will facilitate the development of scientific models of mind. Published by Elsevier Science B.V. All rights reserved.

1. Introduction

What is mind? What is the relationship between the mind and the brain? What is thought? What are the mechanisms that give rise to imagination? What is perception and how is it related to the object perceived? What are emotions and why to we have them? What is will and how do we choose what we intend to do? How do we convert intention into action? How do we plan and how do we know what to expect from the future?

☆ Invited – delivered as JCIS'98 Banquet Speech.
[1] E-mail: james.albus@nist.gov

Published by Elsevier Science Inc.
PII: S 0 0 2 0 - 0 2 5 5 (9 8) 1 0 1 0 2 - 0

Until recently such questions could only be addressed indirectly by subjective introspection, or by psychological experiments in which the majority of the critical variables cannot be measured or controlled. Only in the past half century, since the invention of the electronic computer has it become possible to approach these issues directly by building machines and programs that exhibit some of the mind's essential qualities; such as the ability to recognize patterns and relationships, to store and use knowledge, to reason and plan, to learn from experience, and to evaluate what is happening. This is a crucial step in the study of mind, for it makes it possible to build mathematical models, and conduct experiments where, at least in principle, all the variables can be measured.

Research in neural nets, brain modeling, fuzzy systems, and genetic algorithms is providing insight into learning and the similarities and differences between neuronal and electronic computing. Artificial intelligence and linguistics are probing the nature of language. Image understanding has developed into a field of its own. There has been significant progress in rule based reasoning, planning, and problem solving. Game theory and operations research have developed methods for decision making in the face of uncertainty. Autonomous vehicle research has produced advances in real-time sensory processing, world modeling, navigation, and locomotion. Research in robotics and automated manufacturing has produced intelligent hierarchical controls, distributed databases, representations of object geometry, process plans, and material properties. Control theory has developed precise understanding of stability, adaptability, and controllability under various conditions of feedback and noise. Powerful mechanisms have been developed for parallel processing, recursive estimation, and focusing of attention. Engineering solutions exist for fusing sensory input from multiple sources, and assessing the believability of noisy data.

Since the 1950's, a wide variety of robotic systems have been designed and built – from experimental laboratory vehicles that wander about, follow walls, and pick up sundry items, to precision assembly systems that use vision to acquire parts and computer aided design (CAD) models to plan motions. Many approaches have been explored and various architectures designed, from subsumption and neural nets, to SOAR [1] and RCS [2]. Entire factories have been automated and products as complex as the Boeing 777 aircraft containing over three million parts have been designed and engineered entirely in software, without physical mockups.

Progress is also rapid in the cognitive and neurosciences. Neuroanatomy is producing maps of the interconnecting pathways of the brain. Neurophysiology is determining how neurons compute functions and communicate information. Neuropharmacology is discovering many of the transmitter substances that modify value judgments, compute reward and punishment, motivate behavior, and produce learning. Psychophysics provides clues as to how humans and animals perceive objects, events, time, and space, and reason about relationships. Behavioral psychology is creating models of mental and emotional development and behavior.

While the mind itself remains a mysterious and inaccessible phenomena, many of the components of mind, such as perception, behavior generation, knowledge representation, value judgment, reason, intention, emotion, memory imagination, recognition, learning, and intelligence are becoming well defined and amenable to analysis. Progress is rapid, and there exists an enormous and rapidly growing literature in each of the above fields. What is lacking is a general theoretical model which ties all these separate areas into a unified framework that includes both biological and machine embodiments of the components of mind. In 1991, I published an outline for a general theory of intelligence [3]. This theory was expressed in the notation of the real-time control system (RCS) developed at NIST and elsewhere for the design of intelligent control systems [2]. In this paper, I will illustrate how many of the concepts developed for intelligent machines apply to biological intelligence and suggest how engineering principles might be developed for the design and analysis of practical intelligent systems.

2. The fundamental elements

In any scientific endeavor, it is necessary to precisely define concepts and clearly state assumptions. I therefore begin with some axioms and definitions.

Axiom 1. The functional elements of an intelligent system are behavior generation, sensory perception, world modeling, and value judgment.

Df: **behavior generation (BG)** the planning and control of action designed to achieve behavioral goals.

Behavior generation organizes the response of a collection of agents to task commands. Behavior generation accepts task commands with goals, objects, and priorities. It decomposes tasks into jobs and assigns jobs and resources to agents. It formulates and selects plans and develops schedules for possibly coordinated actions by agents. It executes plans and reacts to feedback so as to follow plans in spite of local perturbations and unexpected events. Finally, behavior generation produces output commands that are either decomposed further, or act directly on the environment.

Df: **planning** a process that:

1. assigns responsibility to agents for jobs, and allocates resources to agents for performing their assigned jobs,
2. hypothesizes strings of actions (plans) for agents from a vocabulary of possible actions to accomplish jobs,
3. simulates and predicts the results of executing these hypothesized plans,
4. evaluates the predicted results of the hypothesized plans,

5. selects the hypothesized plan with the most favorable results for execution.

The planning process may use either a heuristic or an exhaustive search strategy for synthesizing hypothesized plans. Heuristic strategies may include the selection of previously generated plans from a library.

Df: **agent** a set of computational elements that plan and control the execution of jobs, correcting for errors and perturbations along the way.

An agent may servo its output to follow a planned trajectory, or may sequence discrete actions and branch on conditions. An agent also assigns jobs and resources to subordinates. The computational elements in an agent may include sensory perception, world modeling, and value judgment functions and a knowledge database.

Df: **sensory perception (SP)** the transformation of data from sensors into meaningful and useful representations of the world.

Sensory perception accepts input data from sensors that measure states of the external world as well as internal states of the system itself. Sensory perception scales and filters data, computes observed features and attributes, and compares observations with expectations generated from internal models. Correlations between sensed observations and internally generated expectations are used to detect events and recognize entities and situations. Differences between sensed observations and internally generated expectations are sent to world modeling to update internal models. Sensory perception also classifies, generalizes, and clusters, or groups, recognized entities and detected events into higher order entities and events, and computes attributes of entities and events.

Df: **value judgment (VJ)**.

(a) the computation of cost, risk, and benefit of actions and plans,

(b)the estimation of the importance and value of objects, events, and situations,

(c) the assessment of reliability of information,

(d) the calculation of reward or punishment resulting from perceived states and events.

Value judgment evaluates perceived and planned situations thereby enabling behavior generation to select advantageous goals and set priorities among competing behavioral possibilities. It computes what is important (for attention), and what is rewarding or punishing (for learning). Value judgment is performed in the brain by the limbic system.

Df: **world modeling (WM)** a process that performs four principal functions:

1. Uses sensory input to construct, update, and maintain a knowledge database, including iconic images, symbolic lists, entity and event frames, and semantic and pragmatic relationships between entities, and links between symbolic and iconic representations. In biological systems, this is the function of short term and long term memory.
2. Answers queries from behavior generation regarding the state of the world. It provides knowledge about the state of the world to be used by behavior

generation as feedback to servo behavior to follow desired plans, and to provide the latest status information on which to base planning operations. This is the function of recall.

3. Simulates results of possible future plans. Simulated results are evaluated by the value judgment system in order to select the best plan for execution. Simulation performed by the brain is what we call thinking. Our ability to plan depends on the fidelity of our internal model of how the world works.
4. Generates sensory expectations based on knowledge in the knowledge database. Expectations are used by sensory perception to configure filters, masks, windows, and templates for correlation, model matching, and recursive estimation; and for clustering. This corresponds to the tendency of biological brains to see and hear what they expect to see and hear.

Axiom 2. The functional elements of an intelligent system are supported by a knowledge database that stores a priori and dynamic information about the world in the form of state variables, symbolic entities, symbolic events, rules and equations, structural and dynamic models, task knowledge, signals, images, and maps.

Df: **knowledge database (KD).** A set of data structures filled with the static and dynamic information that provide a best estimate of the state of the world and the processes and relationships that effect events in the world.

In the knowledge database:

State variables represent the current estimated state of the world.

Entity frames are list data structures that store symbolic representations of features, objects, or groups that exist in the world, or in the imagination. An entity frame consists of a list head with a name as an address, plus a set of attribute-value pairs, and a set of relations to other entities or events. These relationships represent semantic meaning.

Event frames are list data structures that store symbolic representations of state transitions, or situations that occur at particular times and places, or sequences of states or situations that transpire over intervals of time and space in the world. An event frame also consists of a name, a set of attribute-value pairs, and a set of relationships to other events or entities.

Rules and equations such as if/then rules, the predicate calculus, and differential equations can express physical laws that describe the way the world works, as well as mathematical and logical formulae that describe the way things relate to each other.

Images are two-dimensional functions of attribute values that may be sampled by arrays of sensors, or pixels. Images may be generated in a number of ways. For example, an image may be formed by the optical projection of light from a scene in the world through a lens onto an array of photoreceptors (or pixels) such as the retina or a CCD TV camera. An image may also be formed by pressure on an array of tactile sensors on the skin. An image consists of attributes such as brightness, pressure, spatial or temporal gradients, stereo disparity, or computed values such as range, or flow that are derived from other images. An image may also be generated by internal mechanisms (such as a computer graphics engine) from information stored in symbolic entity frames. In biological systems, image generation corresponds to imagination [4].

Maps are also two-dimensional arrays of pixels, wherein icons or symbolic names, in addition to attributes, are attached to pixels.

Task knowledge is knowledge of how to do things. Task knowledge includes information about the goal, the agents, the task objects, parameters, enabling and disabling conditions, tools and resources required, and plans, scripts, or procedures for generating and refining plans. Control laws and plant models can also be represented as task knowledge.

The knowledge database has two parts: long term and short term memory.

(a). *Long term* (static or slowly varying) memory contains symbolic representations of all the entities, events, and rules that are known to the intelligent system. Long term memory consists entirely of symbolic entity and event frames, plus rules and equations. Attributes from long term frame representations may be transferred into short term memory, or vice versa.

(b) *Short term* (dynamic) memory contains both symbolic and iconic representations of entities-of-attention.

Df: **entities-of-attention.** Entities that have either been specified by the current task, or are particularly noteworthy entities observed in current sensory input.

Short term symbolic entity frames include attributes, pointers to iconic images, and pointers to entities stored in long term memory. Short term iconic representations can consist of attribute images generated directly from sensory observations, or filtered through recursive estimation. Short term iconic images can also be generated by internal mechanisms from short term symbolic entity frames. In machine systems, this is done through simulation and animation. In biological systems, it corresponds to imagination. Short term iconic images can be used to mask or window incoming data, or to compare and fuse incoming sensory observations with internally generated images. Short term iconic images persist in memory only so long as they are refreshed by incoming sensory data or by internally generated images.

Axiom 3. The functional elements and knowledge database can be implemented by a set of computational modules that are interconnected to form nodes in a control system architecture.

Df: **node.** A part of a control system that processes sensory information, maintains a world model, computes values, and generates behavior.

A node corresponds to a set of neurons in the brain that close a loop between afferent and efferent neural pathways. In doing so, each node typically performs the functions of behavior generation, sensory perception, world modeling, and value judgment. A typical node from the RCS reference model architecture [2] is shown in Fig. 1.

Within each node, interconnections between behavior generation, world modeling, and value judgment modules enable task decomposition, planning, and reasoning about future actions. Interconnections between sensory perception, world modeling, and value judgment modules enable knowledge acquisition, situation evaluation, and learning. Interactions between sensory perception and world modeling modules enable recursive estimation for optimal filtering and prediction. Interconnections between sensory perception, world modeling, and behavior generation modules close a reactive feedback control loop between the observed input and the commanded action. Input commands convey task goals and priorities from higher level nodes. Output commands convey subtask goals to lower level nodes, or directly to actuators. The downward flow of commands corresponds to efferent pathways in the brain. The upward flow of information through the sensory perception modules correspond to afferent pathways.

Connections to the operator interface have no biological analog. These enable a human operator to insert commands to override or modify system behavior, or to observe the values of state variables and entity attributes. The

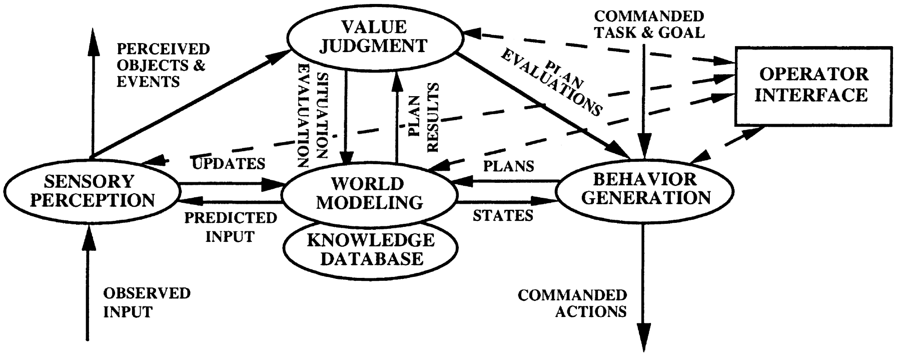

Fig. 1. *A node in the RCS reference model architecture.* The functional elements of an intelligent system are behavior generation (planning and control), sensory perception (filtering, detection, recognition, and interpretation), world modeling (storing and retrieving knowledge and predicting future states), and value judgment (computing cost, benefit, importance, and uncertainty). These are supported by a knowledge database, and a system architecture that interconnects the functional modules and the knowledge database. This collection of modules and their interconnections makes up a generic node in the RCS reference model architecture. Each module in a node may have an operator interface.

operator interface can also be used for system maintenance, programming, and debugging.

Axiom 4. The complexity inherent in intelligent systems can be managed through hierarchical layering.

Intelligent systems are inherently complex. Hierarchical layering is a common method for organizing complex systems that has been used in many different types of organizations throughout history for effectiveness and efficiency of command and control. In a hierarchical control system, higher level nodes have broader scope and longer time horizons, with less concern for detail. Lower level nodes have narrower scope and shorter time horizons, with more focus on detail.

In the RCS reference architecture, behavior generating modules in nodes at the upper levels in the hierarchy make long range strategic plans consisting of major milestones, while lower level behavior generating modules successively refine the long range plans into short term tactical plans with detailed activity goals. Sensory perception modules at lower levels process data over local neighborhoods and short time intervals, while at higher levels, they integrate data over longer time intervals and larger spatial regions.

World model knowledge at low levels, is short term and fine grained, while at higher levels it is broad in scope and generalized. At every level, feedback loops are closed to provide reactive behavior, with high-bandwidth fast-response loops at lower levels, and slower more deliberative reactions at higher levels. RCS thus provides what Brooks calls "coherent behavior from many adaptive processes" [5].

At each level, state variables, entities, events, and maps are maintained at the resolution in space and time that is appropriate to that level. At each successively lower level in the hierarchy, as detail is geometrically increased, the range of computation is geometrically decreased. As temporal resolution is increased, the span of interest decreases. As plans become more detailed, the planning horizon shrinks. This produces a ratio that remains relatively constant throughout the hierarchy. A design goal is for behavior generating functions at each level to generate plans of roughly the same number of steps, and for sensory perception functions to compute entities containing roughly the same number of subentities. At higher levels, plans, perceived entities, and world modeling simulations are more complex, but there is more time available between replanning intervals for processes to run. Thus, hierarchical layering keeps the amount of computing resources needed in each node within manageable limits.

At the top of the hierarchy, strategic objectives and priorities influence the selection of goals and the prioritization of tasks throughout the entire hierarchy. However, the details of execution are left to subordinates.

At intermediate levels, tasks with goals and priorities are received from the level above, and subtasks with subgoals and shorter range attention priorities are output to the level below. Again, the details of execution are left to subordinates.

At each level, global goals from a higher level are refined and focused onto more narrow and finer resolution subgoals. At each level, attention is focused into a more narrow and finer resolution view of the world. The effect of each hierarchical level is thus to geometrically refine the detail of the task and the view of the world, while keeping the computational load at all levels within limits that can be handled by intelligent agents of modest capacity.

3. The RCS reference model architecture

A set of generic nodes such as illustrated in Fig. 1 can be interconnected in a hierarchical control architecture for an intelligent machine system. The diagram shown in Fig. 2 consists of a hierarchy of control nodes wherein each of the nodes contain the set of modules and interconnections illustrated in Fig. 1. Each of the nodes is therefore an intelligent controller capable of planning and control, knowledge representation, value judgment, and sensory perception. Each of the nodes closes a feedback loop between afferent and efferent pathways. An operator interface may access modules in any node at any level.

The example of a reference model architecture in Fig. 2 is designed for a military vehicle system with four subsystems: locomotion, mission package, communication, and attention. Each of the four subsystems has one or more mechanisms, each of which has one or more actuators and sensors. For example, the locomotion subsystem may consist of a navigation and driving controller with several controllers for steering, braking, throttle, and gear shift, plus ignition, lights, horn, and turn signals, each of which has one or more actuators and sensors. The mission package may have loading, aiming, and firing subsystems each of which has numerous sensors and actuators. The communication subsystem might consist of a message encoding subsystem, a protocol syntax generator, and communications bus interface, plus antenna pointing and band selection actuators. The attention subsystem may contain cameras, laser range imagers, infrared cameras, radar, and acoustic sensors. Sensory perception algorithms may detect and track objects, surfaces, edges and points, and compute trajectories for pan, tilt, and focus actuators to point cameras, range finders, and antennae. All of these functions need to be coordinated in order to successfully achieve behavioral goals. The horizontal curved lines between WM modules represent the sharing of state information between nodes within subtrees in order to synchronize related tasks.

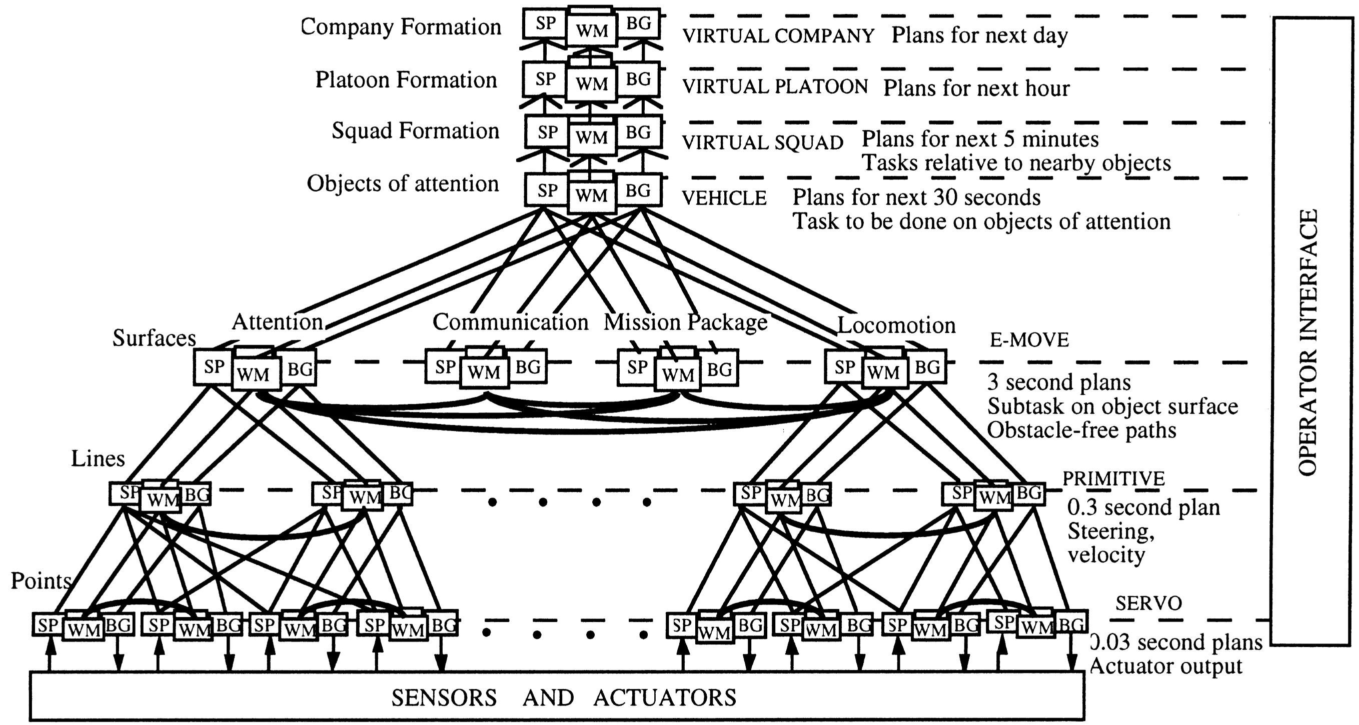

Fig. 2. *A RCS reference model architecture for an individual vehicle.* Processing nodes are organized such that the behavior generation (BG) modules form a command tree. Information in the knowledge database (KD) is shared between world modeling (WM) modules in nodes within the same subtree. KD modules are not shown in this figure. On the right, are examples of the functional characteristics of the behavior generation (BG) modules at each level. On the left, are examples of the type of entities recognized by the sensory perception (SP) modules and stored by the WM in the KD knowledge database at each level. Sensory data paths flowing up the hierarchy typically form a graph, not a tree. Value judgment (VJ) modules are hidden behind WM modules. A control loop is closed at every node. An operator interface may provide input to, and output from, modules in every node.

The operator interface provides the capability for the operator to interact with the system at any time at a number of different levels – to adjust parameters, to change speed, to select or verify targets, or to authorize the use of weapons. The operator interface provides a means to insert commands, change missions, halt the system, alter priorities, perform identification of friend-or-foe, or monitor any of the system functions. The operator interface can send or display information from the communications subsystem, or display any of the state variables in the world model at a rate and latency dictated by the communications bandwidth. Using the operator interface, a human operator can view situational maps with topographic features, with overlays that indicate the position and movement of both friendly and enemy forces. The operator interface may display graphic images of motion paths, or print out control programs (plans), in advance, or while they are being executed. The operator interface also enables the operator to run diagnostics in the case of system malfunctions.

In Fig. 2, three levels of control are shown above the node representing the individual vehicle. These three additional levels represent a virtual chain of command that exists above the individual vehicle. Because each vehicle is semi-autonomous, it carries a copy of the control nodes that contain its superiors in the command chain. This virtual chain of command serves four functions. First, it provides each vehicle with an estimate of what its superiors would command it to do if they were in direct communication. Second, it enables, any vehicle to assume the duties of any of its superiors in the event this should become necessary. Third, it enables each vehicle to dedicate a separate node to handle higher level tasks. In this example, the virtual chain of command consists of three levels with three different planning horizons (five minutes, one hour, and one day). These three levels deal with external objects and maps at three different scales and ranges. Fourth, it provides a natural interface for human commanders at the squad, platoon, or company level to interface with the vehicle at a level relevant to the task being addressed. There, of course, may be more than three levels above the vehicle in the virtual chain of command.

At each layer of the RCS hierarchy, there are both deliberative and reflexive elements. In each node at each level, sensory data are processed, entities are recognized, world model representations are maintained, and tasks are deliberatively decomposed into parallel and sequential subtasks, to be performed by cooperating sets of subordinate agents. Also in each node, feedback from sensors reflexively closes a control loop allowing each agent to respond and react to unexpected events. The result is a system that combines and distributes deliberative and reflexive features throughout the entire hierarchical architecture, with both planned and reactive capabilities tightly integrated at all levels and time frames. Fig. 3 illustrates the internal structure of a behavior generation (BG) module. The BG module represents an opera-

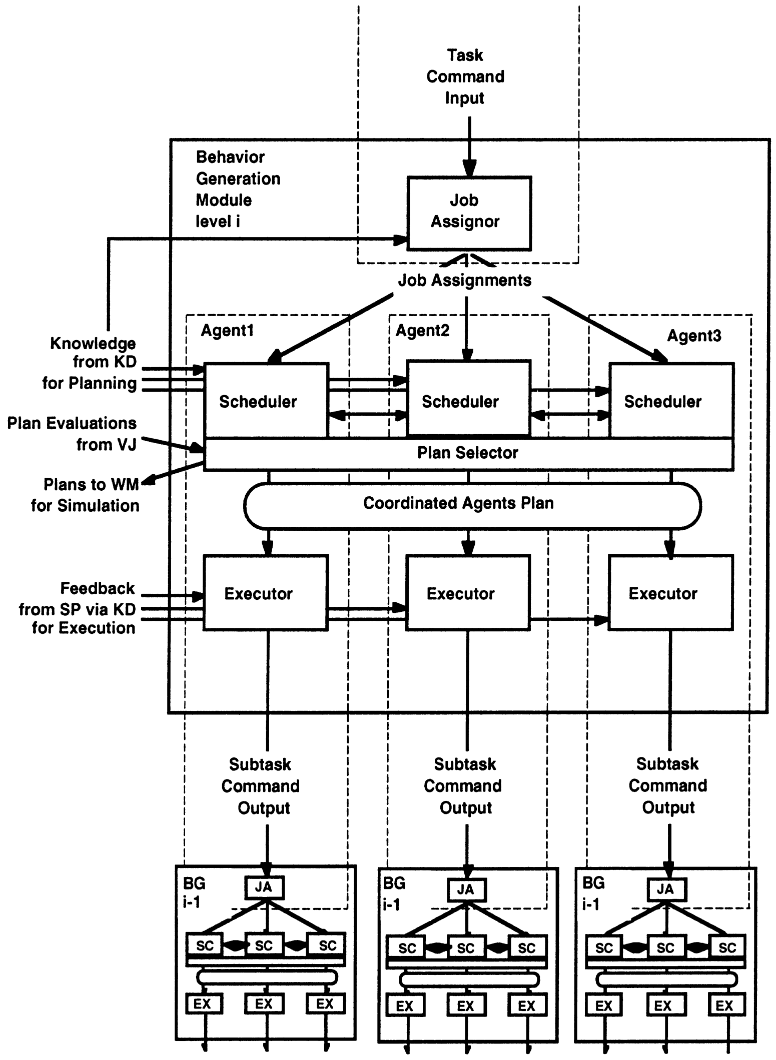

Fig. 3. *Internal structure of a Behavior Generation* (*BG*) *module.* A BG module is an organizational unit consisting of a Job Assignor, a set of Schedulers, a Plan Selector, a plan holding register that contains the coordinated agents plan, and a set of Executors. The Schedulers, Executors, and Job Assignors comprise agents within the BG units. Each agent is part of two BG units at two different levels. At level i, an agent is a team member, or peer. At level i-1, the same agent is a supervisor.

tional unit that typically is comprised of several intelligent agents. It is important to distinguish clearly between the agents and the BG organizational unit to which they belong. An agent is typically part of two BG modules at two different levels. At one level, an agent is a member of a team, and subordinate to the supervisor of the team. At the next lower level, the same agent is a supervisor of a team of agents in the BG module at the next lower level. The job assignor submodule thus is part of the supervisor agent for a BG unit at level *i*. The job assignor assigns jobs to agents in the BG unit, and works with them to develop a set of schedules for each of the subordinate agents within the i-level BG unit. The set of schedules represents a plan for the BG unit. Each of the subordinate agents in the i-level BG unit contains a scheduler and an executor at level *i*, and is a supervisor (i.e., a job assignor) at level *i*-1. Each agent is thus a peer in a BG module at a level, and a supervisor of a BG module at the next lower level.

Fig. 4 illustrates the details of the interactions that take place within and between the BG, WM, SP, and VJ modules in a single node. A task command into a node is of the form ⟨Do action on object to achieve goal x* ⟩. The task name and goal enters a BG module where the task is decomposed into jobs for agents. Each agent has a scheduler that generates a schedule and coordinates with schedulers of other agents. This produces a tentative plan which is a path from the current estimated state $\hat{X}$ to the goal state x*. The tentative plan is submitted to a plan simulator in the WM which predicts results. These results are evaluated by the plan evaluator in the VJ which returns cost-benefit analysis to the plan selector. If the evaluation is satisfactory, the tentative plan is selected for execution. Otherwise replanning is called for, and another tentative plan is generated by the job assignor and scheduler submodules.

The object specified by the task command is sent to the world model knowledge database which looks up a corresponding entity in the long-term memory database. This entity is then entered into the short-term memory list of entities-of-attention. Attributes of entities-of-attention are used to generate a predicted image that can be compared with an observed image. The predicted image defines windows for correlation, comparison, recognition, and clustering. Differences between predicted and observed images can be used to update the estimated state $\hat{X}$, including the estimated attributes of the entity-of-attention. This is a looping process of recursive estimation that can be used to implement a tracking filter, a predictor, or a phase-lock loop. The estimated state $\hat{X}$ is used by the executor to servo the output to follow the selected plan. It is also used by the job assignor and scheduler submodules for planning, and by the WM plan simulator for predicting results of tentative plans. Recognized and clustered entities are evaluated by the VJ and transmitted to higher level sensory perception modules for more global processing.

This recursive estimation procedure has recently been implemented on modest computing hardware for vision based lane following by highway vehicles. Dickmanns [6] has implemented a 4-D model in world coordinates.

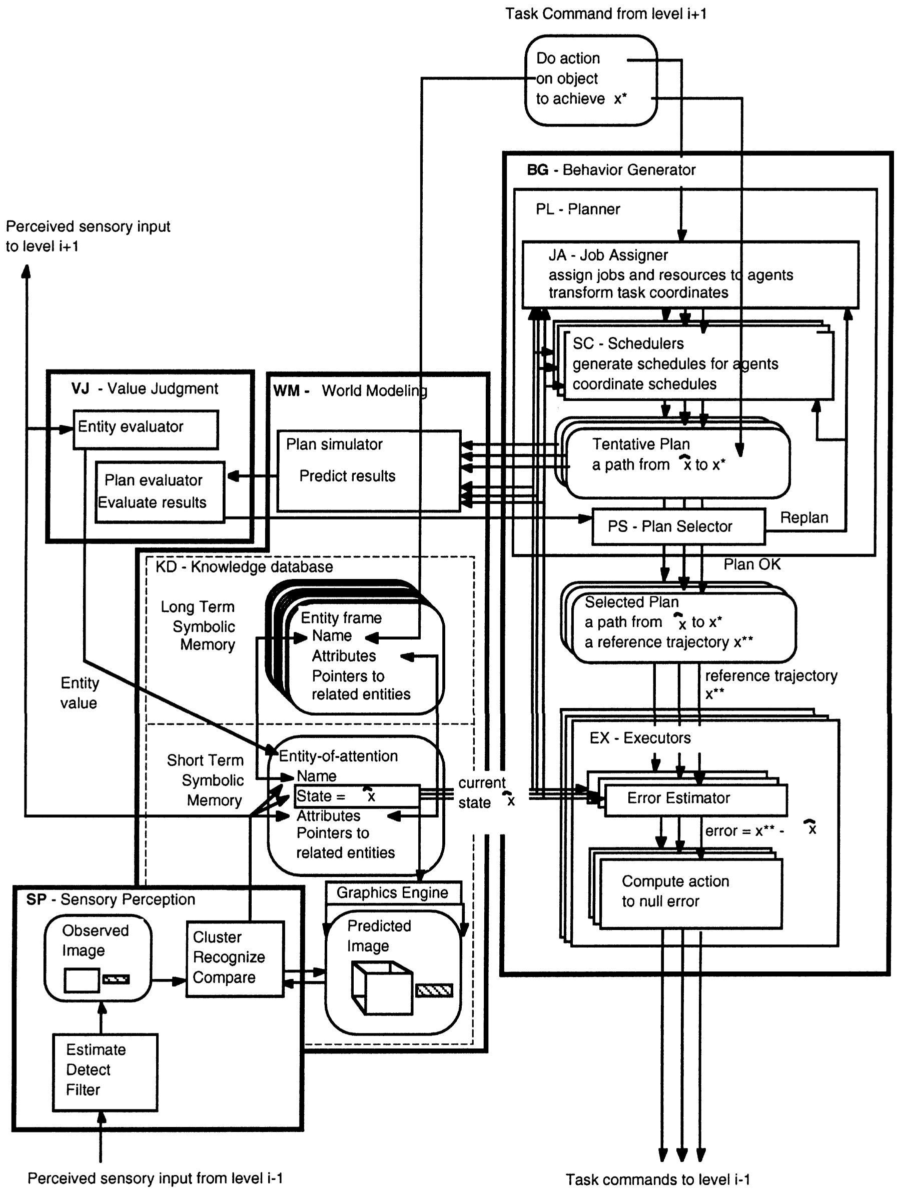

Fig. 4. *Relationships within a single node of the RCS architecture.* The behavior generating (BG) modules contain job assignor (JA), scheduler (SC), plan selector (PS) and executor (EX) submodules. The world modeling (WM) module contains a plan simulator and mechanisms for updating the knowledge database (KD), which contains both long term and short term symbolic representations and short term iconic images. The sensory perception (SP) module contains filtering, detecting, and estimating algorithms, plus mechanisms for comparing predictions generated by the WM module with perceived input from sensors. It has algorithms for recognizing entities and clustering entities into higher-level entities. The value judgment (VJ) module evaluates plans and places values on entities recognized in the observed sensory input.

Schneiderman and Nashman [7] implemented a 2-D model in image coordinates. Speeds of up to 100 kph on highways and 40 kph on a winding test track have been achieved by the 2-D system. The 4-D system has achieved speeds of up to 130 kph in normal highway traffic with automatic lane changing while simultaneously tracking up to five other vehicles [8].

Axiom 5. The complexity of the real world environment can be managed through focusing attention.

Intelligent systems must operate in a real world environment which is infinitely rich with detail. The real world environment contains an unlimited variety of objects, such as ground, rocks, grass, sand, mud, trees, bushes, buildings, posts, ravines, rivers, roads, vehicles, weapons, people, and enemy and friendly positions. The environment also contains elements of nature, such as wind, rain, snow, sunlight, and darkness. All of these objects and elements have an infinite regression of detail, and the world itself extends infinitely far in every direction.

Yet, the computational resources available to any intelligent system are finite. No matter how fast and powerful computers become, the amount of computational resources that can be embedded in any practical system will be limited. Therefore, it is imperative that the intelligent system be able to focus the available computing resources on what is important, and ignore what is irrelevant.

Top down, what is important is defined by behavioral goals. Top down goals and high level perceptions generate expectations of what should be encountered during the evolution of the task. Bottom up, what is important is the unexpected, unexplained, unusual, or dangerous. The lower level sensory perception functions detect variations between what is expected and what is observed. The lower levels also compute attributes of signals or images that may indicate problems or emergency conditions, such as limits being exceeded on position, velocity, acceleration, vibration, pressure, force, current, voltage, or thermal sensor signals.

Focusing of attention can be accomplished by sampling the environment at high resolution at important points, and with low resolution elsewhere. For example, most of the photodetectors in the visual field are concentrated in the foveal region, and lower resolution in the periphery. Similarly, tactile attention is accomplished by high resolution spatial distribution of tactile sensors in the finger tips, lips, and tongue, with much lower resolution in other regions of the skin. Intelligent control points the fovea at points in the visual field that are important, and guides the fingers to touch objects at points important to the task goal. Hierarchical layering focuses computing resources near the present, with exponentially lower resolution for longer time horizons, both for planning the future and analyzing the past.

4. Discussion

In RCS, each node, and each module and submodule within a node, is implemented as an augmented finite state machine which runs asynchronously as a cyclically executing process. The execution cycle typically is triggered by a clock at a fixed repetition rate that is an order of magnitude faster than the time constant of the process being controlled, but may be triggered by events. Each finite-state machine is surrounded by a set of input and output data buffers. At the beginning of each cycle, the submodule reads from its input buffers and processes the inputs into a form suitable for a state-table that encodes a set of state dependent if/then rules of a form that are common in expert systems. The processed input is compared with the rules in the state-table. The rule that matches causes the process to go to a next state, possibly execute a procedure, and compute an appropriate output. Each submodule also computes a set of diagnostic functions, such as the time required for the process to complete, the maximum time the process has taken, and the average time taken. Each submodule has an interface for an operator that provides the operator the ability to halt, single-step, and/or display the value of any variable and the state of any process at any time during execution. A process may be halted, parameters examined by a programmer, variables changed, and execution resumed. Communications between processes in the RCS system are designed so that all processes cycle independently and run completely asynchronously with no protection against messages getting overwritten. In this respect, RCS is similar to Brooks' πβ machine [5].We agree with Brooks that design principles and computational structures should constrain system design and force solutions to maintain biological relevance. To do otherwise allows divergence from approaches that can draw on (or contribute to) concepts from the cognitive and neurosciences. Our differences with Brooks include our insistence on a systematic design of the topology of the computing structure. While we admit that random design choices coupled with natural selection may eventually lead to an optimum topology, we believe that systematic design principles will reach the goal in far fewer iterations.

RCS programming tools and software templates are being built that provide the system developer an easy way to configure an RCS system. The templates automatically generate all the required utilities and diagnostic features, and provides slots in a menu for inputs, outputs, and system parameters. Software templates are implemented in C++. A graphical design tool enables a programmer to define a RCS submodule with the click of a mouse, and interconnect submodules by click and draw techniques. These programming tools have enabled RCS systems with hundreds of submodules to be designed and built in a few months.

RCS has been implemented on a variety of platforms, including Sun workstations, 486 and Pentium class PC computers, VME systems, and

Macintosh machines using a number of different operating systems, including Forth, pSOS, DOS, VxWorks, and Lynx OS [2]. The overhead for a RCS template running on a 486 class machine is about five microseconds. The cycle time for a typical low level RCS submodule is 30 milliseconds. However, for high performance machine tool applications, servo loops may be closed every 300 ms.

5. Conclusions

While many deep theoretical issues regarding the nature of the mind remain, much is known and progress is rapid. Intelligent machines research is beginning to yield to engineering approaches. Intelligent systems are beginning to exhibit impressive performance capabilities in practical applications such as manufacturing systems and highway vehicles. Some of the most promising lines of research appear to be in:

(a) combining goal-driven planning with reflexive behavior,
(b) focusing of attention and active control of sensing,
(c) hierarchical decomposition of tasks and goals,
(d) the use of recursive estimation for sensory perception,
(e) the development of a reference model architecture for intelligent system design.

A reference model architecture paves the way for design principles and software engineering tools that facilitate the building of intelligent machine systems.

Recent work in open system architectures for intelligent controllers is leading toward specification of canonical functional modules and standards for application programming interfaces (APIs) between functional modules in open architecture machine controllers [9]. API standards promise to facilitate system integration and enable incremental upgrades of capability.

It soon may be possible to add new layers or subsystems without modifying system software. Eventually, systems developed in different laboratories might be integrated into large experimental systems consisting of thousands of computing platforms in hundreds of different laboratories. At that point, intelligent machine systems might approach the complexity and computing power of the human brain. Under such circumstances, the prospect is bright for building scientifically valid experimental models of the mind.

[2] Certain commercial equipment, instruments, or materials are identified in this report in order to facilitate understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

It should be noted, in closing, that such models would not only advance the scientific inquiry into of the nature of mind, but would very likely also lead to practical improvements in intelligent machine systems technology for manufacturing, construction, transportation, communication, health care, environmental restoration, waste management, security, and military systems. Such developments would have significant economic, social, and political benefits. But that is a subject for another paper.

References

[1] P.S. Rosenbloom, J.E. Laird A. Newell (Eds.), The SOAR Papers, vols. 1 and 2, MIT Press, Cambridge, MA, 1993.
[2] J.S. Albus, The NIST real-time control system (RCS): An approach to intelligent systems research, J Expt. Theo. Art. Intel., 1996.
[3] J.S. Albus, Outline for a theory of intelligence, IEEE Transactions on Systems, Man and Cybernetics, 21 (3), (1991) 473–509.
[4] S.M. Kosslyn, Image and Brain: The Resolution of the imagery Debate, MIT Press, Cambridge, MA, 1994.
[5] R.A. Brooks, Coherent behavior from many adaptive processes, in: Proceedings of the Third International Conference on Simulation of Adaptive Behavior, 1994, pp. 22–29.
[6] E.D. Dickmanns, A general dynamic vision architecture for UGV and UAV, J. Appl. Intel. 2 (1992) 251–270.
[7] H. Schneiderman, M. Nashman, Visual tracking for autonomous driving, IEEE Trans. Robotics and Automation 10 (6) (1994) 769–775.
[8] E.D. Dickmanns, Parallel use of differential and integral representations for realizing efficient mobile robots,in: Proceedings of the Seventh International Symposium on Robotics Research, Munich, 1995.
[9] Technologies for Enabling Agile Manufacturing, (TEAM), Application Programming Interfaces, Internet location http://isd.cme.nist.gov/info/team, 1996.