Comparative Analysis of Frameworks for Knowledge-Intensive Intelligent Agents

Randolph M Jones
Colby College & Soar Technology
5857 Mayflower Hill
Waterville, ME 04901-8858
Tel: 207.872.3831
rjones@soartech.com

Robert E Wray
Soar Technology
3600 Green Road Suite 600
Ann Arbor, MI 48105
Tel: 734.327.8000
wray@soartech.com

Abstract

This paper discusses representations and processes for agents and behavior models that encode large knowledge bases, are long-lived, and exhibit high degrees of competence and flexibility while interacting with complex environments. There are many different approaches to building such agents, and understanding the important commonalities and differences between approaches is often difficult. We introduce a new approach to comparing approaches based on the notions of deliberate commitment, reconsideration, and a categorization of representations. We review three agent frameworks, concentrating on the major representations and processes each directly supports. By organizing the approaches according to a common nomenclature, the analysis highlights points of similarity and difference and suggests directions for integrating and unifying disparate approaches and for incorporating research results from one approach into alternative ones.

1 Overview

A variety of frameworks exist for designing intelligent agents and behavior models. Although they have different emphases, these frameworks each provide coherent, high-level views of intelligent agency. However, more pragmatically, much of the complexity of building intelligent agents is in the low-level details, especially when building agents that exhibit high degrees of competence while interacting in complex environments. We call such agents "knowledge-intensive", to distinguish them from smaller scale, single-task agents (e.g., service brokers) that are often fielded in multi-agent systems. Good examples of fielded knowledge-intensive agents include a real-time fault diagnosis system on the Space Shuttle [Georgeff and Rao, 1996] and a real-time model of combat pilots [Jones et al., 1999]. Knowledge-intensive agents are also often used in "long-life" situations, where a particular agent needs to behave appropriately and maintain awareness of its environment for a long period of time. Additionally, knowledge-intensive agents must be engineered such that their knowledge can be easily updated as environment and task requirements change during deployment.

Transfer and generalization of results from one framework to others is usually slow and limited. The reasons for such limited transfer include differences in nomenclature and methodology that makes it more difficult to understand and apply results, and to specify low-level details that are not given by the frameworks, but become important in actual implementation. Our goal is to develop techniques that will minimize framework-specific descriptions and that bridge the gap between a framework's theory and the details of its implementation, especially clarifying which details are intrinsic to particular approaches and which are not.

This paper reviews three existing agent frameworks in order to explore what they specify (and do not) about an agent's construction. The chosen frameworks have proven successful for building knowledge-intensive agents or specifically address constraints on agents with high levels of competence (such as human behavior models). We identify the representations and agent processes that the frameworks dictate for agent design. This comparative analysis, to our knowledge, is novel and provides insights into the tradeoffs inherent in these systems for building intelligent agents. The goal is truly comparative. Each system we review arguably has a unique application niche, and we are not seeking to suggest one framework is better than another. Rather, in comparing them, especially in noting convergences and divergences in knowledge-intensive agent applications, we seek to develop a uniform methodology for comparing frameworks and ultimately to speed the development and evaluation of new agent architectures, by making research results more communicable and transparent to researchers not working within the specific subfield of AI or Cognitive Science in which new architecture developments are made.

2 Review of Agent Frameworks

We review three mature frameworks for intelligent agents that represent three different theoretical traditions (philosophical and logical, psychological, and func-

tional). Our intent is to consider the primary representational constructs and processes directly supported by each. We focus on these aspects of agent frameworks because an agent is essentially the sum of a system's knowledge (represented with particular constructs) and the processes that operate on those constructs [Russell and Norvig, 1994]. The goal of focusing on these types of frameworks is that they provide *integrated* platforms for building intelligent systems. They make commitments to various representations and processes that are necessarily dependent on each other. However, the analysis will show that many of these dependencies appear at an architectural, or implementation, level. Using an information-level analysis, we describe a more general framework that specifies a space of potential representations and processes (abstracted away from their specific implementations), together with a space of potential integrations of these components.

We focus on frameworks that have been used to build large-scale, highly capable agent systems. One motivating factor for this analysis was recognition that *implemented* BDI and Soar systems, while originating from different theoretical starting points, have converged on similar solutions for large-scale systems. However, this analysis is not complete. Other frameworks include additional representations and processes that may be important for knowledge-intensive agent applications (e.g., 4D/RCS [Albus, 2001], RETSINA [Payne et al., 2002] and ACT-R [Anderson, 1998]). In the long term, we will extend our analysis to these and other frameworks as well.

2.1 BDI

The BDI (Beliefs-Desires-Intentions) framework provides a methodology for building competent agents that are grounded in logical theories of rationality [Georgeff and Lansky, 1987; Rao and Georgeff, 1995; Wooldridge, 2000]. A basic assumption in BDI is that intelligent agents ought to be rational in a formal sense, meaning rationality (as well as other properties) can be logically proven. However, the framework goes beyond vague notions of proof as rationality by providing an integrated set of elements that serve as first-class representational objects within a logical framework for reasoning. In BDI, actions arise from internal constructs called intentions. An intelligent agent cannot make rational decisions about intentions until it has at least some representation of its beliefs about its situation. Any particular set of beliefs may logically entail many different situations that the agent considers desirable (subject to logical constraints governing desirability, together with preference knowledge about goals). Given limited resources, however, the agent can often only act on some subset of these desires, so the agent commits to subset, its intentions, to pursue.

A BDI agent's actions must be logically consistent with its combination of beliefs and goals (again as specified by various logics for defining consistency in rational agents). This property is not generally true of the other frameworks we examine. BDI has a number of distinct implementations, among them IRMA [Bratman et al, 1988], PRS [Georgeff and Lansky, 1987], dMARS [d'Inverno et al, 1997], and JAM [Huber, 1999].

2.2 GOMS

GOMS (Goals, Operators, Methods, and Selections) was developed from a psychological perspective as an analysis tool mostly for human-computer interaction [Card et al., 1983]. GOMS is not strictly an agent framework, but it formalizes many details of high-level human reasoning and interaction in the same spirit of integration as other knowledge-intensive agent architectures. However, GOMS is particularly interesting because knowledge-intensive agents are often used to simulate human behavior. Although GOMS has not been used to develop large-scale systems, it has been used to represent the human knowledge necessary for performing many tasks, including complex human activity. We include GOMS because the representation and process regularities it has identified are critical for knowledge-intensive agents that will encode this type of knowledge. In addition, improvements in efficiency increasingly allow executable cognitive models to compete with AI architectures in application areas (e.g., [John et al., 1994]).

GOMS systems explicitly encode hierarchical task decompositions, starting with a top-level task goal, plus a number of methods, or plans, for achieving various types of goals and subgoals. Each goal's plan specifies a series of actions (called operators) invoking subgoals or primitive actions to complete the goal. Selection rules provide conditional logic for choosing between plans based on the agent's current set of beliefs. Like BDI, GOMS is a high-level framework, realized in a number of individual implementations, such as GLEAN [Kieras et al., 1995], APEX [Freed and Remington, 2000], CPM-GOMS [Gray et al., 1993], and NGOMSGL [Kieras 1997]. Early implementations of GOMS (as with many early agent architectures) relied on hand-crafted representations of problems and situations, instead of being situated in interactive environments. However, many of the more recent implementations contain integration subsystems to interact with realistic environments, including running user-interface models. These variations move GOMS even closer to agent architectures, by requiring the models to address explicitly issues of perception and action.

2.3 Soar

Soar has roots in cognitive psychology and computer science, but it is primarily a functional approach to encoding intelligent behavior [Laird et al., 1987]. The continuing thread in Soar research has been to find a minimal but sufficient set of mechanisms for producing intelligent behavior. An additional hallmark of efforts with Soar has been to focus on a principled integration of representations and processes. These goals have resulted in

uniform representations of beliefs and knowledge, fixed mechanisms for learning and intention selection, and methods for integrating and interleaving all reasoning.

Like BDI, Soar's principles are based in part on assumed high-level constraints on intelligent behavior. Foremost among these are the problem space hypothesis [Newell, 1982] and the physical symbol systems hypothesis [Newell, 1980]. Problem spaces modularize long-term knowledge so that it can be brought to bear in a goal-directed series of discrete steps. The problem space hypothesis assumes rationality, similar to BDI. However, where BDI frameworks generally provided explicit logical encodings of rationality constraints or principles, this has generally not been the case in Soar systems (although it would certainly be possible to insert similarly encoded constraints into Soar-based agents). The physical symbol-systems hypothesis argues that any entity that exhibits intelligence can be viewed as the physical realization of a formal symbol-processing system. The physical symbol systems hypothesis led to Soar's commitment to uniform representations of knowledge and beliefs. For example, Soar's design encourages the development of models and knowledge representations that allow the same knowledge to be used for planning as for execution. Uniform knowledge representations also allow certain reflective capabilities, where a model can create its own internal representations that mimic the representations it might see from its perceptual systems (or supply to its motor systems).

While Soar imposes strong constraints on fundamental aspects of intelligence, it does not impose functionally inspired high-level constraints (in the spirit of BDI's use of logic, or GOMS' use of hierarchical goal decomposition). Thus, Soar is a lower level framework for reasoning than BDI and GOMS. Either BDI principles [Taylor et al 2004] or GOMS principles [Peck and John 1992] can be followed while using Soar as the implementation architecture.

	Representation	**Commitment**	**Reconsideration**
Inputs			
BDI	Input language		
GOMS	Input language		
Soar	Working memory		
Beliefs			
Entailments			
BDI	Beliefs	Logical inference	Belief revision
GOMS	Working memory		
Soar	Working memory	Match/assert	Reason maintenance
Assumptions			
BDI	Beliefs	Plan language	Plan language
GOMS	Working memory	Operators	Operators
Soar	Working memory	Deliberation/Ops	Operators
Desires			
BDI	Desires	Logic	Logic
GOMS			
Soar	Proposed ops.	Preferences	Preferences
Active Goals			
BDI	Intentions	Deliberation	Decision theory
GOMS	Goals	Operators	
Soar	Beliefs/Impasses	Deliberation	Reason maintenance
Plans			
BDI	Plans	Plan selection	Soundness
GOMS	Methods	Selection	-
Soar			Interleaving
Actions			
BDI	Plan language	Atomic actions	
GOMS	Operators	Operators	
Soar	Primitive Ops	Deliberation	Reason maintenance
Outputs			
BDI	Plan language	Plan language	
GOMS	Primitive ops.	Conditional ops.	
Soar	Working memory	Conditional ops.	

Table 1. Agent framework comparisons. Black, underlined items are specific solutions provided by the framework. Grey items are generally support provided by the framework. No entry means the framework does not explicitly address the element.

3 Analysis of Agent Frameworks

Each of these frameworks provides a coherent view of agency and gives explicit attention to specific representations and processes for intelligent agents. They also reflect different points of emphasis, arising in part from the theoretical traditions that produced them. However, none of the frameworks cover all the points of emphasis.

Table 1 lists the union of a number of base-level representations identified for BDI, GOMS, and Soar. The order of the representations in the table suggests the basic information flow from an external world (through perceptual systems) into agent reasoning (cognition, the primary focus of agent architectures) and then out to the environment (through motor systems). The "representation" column specifies each framework's implementation data structure for the base-level representational element. Each representation also requires a decision point in the reasoning cycle, where an agent must choose from a set of alternatives. We generalize Wooldridge's [2000] concept of intention commitment to specify the process an agent uses to assert some instance of the base-level representation. In Table 1, the "commitment" column identifies the general process used to select among alternatives. Commitments also require maintenance; "reconsideration" is the determination of whether a commitment remains valid (a generalization of intention reconsideration [Schutt and Wooldridge, 2001]).

3.1 Inputs

Any interactive agent must have a perceptual or input system that provides a primitive representation of the agent's environment or situation. Neither BDI nor GOMS specify any particular constraints on input. Soar represents primitive perceptual elements in the same language as beliefs, although it does not dictate the structure

of the perceptual systems that create these elements. However, the constraint that perceptual input must be represented in the same language as beliefs has important implications. Aside from their location in memory, primitive perceptual representations are indistinguishable from beliefs, which is consistent with Soar's principle of uniform knowledge representation. This makes it a relatively simple matter to allow agents to deliberate over potential input situations (or reflect on past or possible future input experiences) and transfer that knowledge directly to actual inputs.

3.2 Beliefs

From primitive perceptual elements, an agent creates interpretations of the environment, or beliefs. Belief representation is perhaps one of the most important distinguishing features between knowledge-intensive agents and other agent systems. Before they can make choices about goals and actions, knowledge-intensive agents must spend significant effort creating stable representations of their understanding and interpretation of the environment.

Using the terminology of reason maintenance systems [Forbus and deKleer, 1993], beliefs can be classified as either entailments (or justified beliefs) or assumptions. Entailments remain in memory only as long as they are (logically) grounded in perceptual representations and assumptions. Assumptions, by definition, receive a high level of commitment and remain in memory, independent of their continuing relevance to–and logical consistency with–the external environment. Assumptions remain asserted until the agent explicitly removes them. Assumptions are necessary because not all beliefs can be grounded in current perception. For example, if an agent needs to remember an object no longer in the field of view, then it must commit to remembering (or "assuming") that the object still exists.

Neither GOMS nor BDI make an explicit distinction between entailments and assumptions. They each provide general mechanisms for maintaining entailments, but not specific solutions. Belief revision [Gardenfors, 1988] is the mechanism of entailment reconsideration in the BDI framework, although details of the process are only defined in various specific implementations. Soar uses a reason maintenance system to assert and retract entailments automatically. Reason maintenance ensures that entailments are logically consistent and follow from their antecedents. All three frameworks support assumptions. Soar requires that assumptions be created as the result of deliberate commitments (operator effects).

Importantly, other frameworks use still other techniques for managing the commitment and reconsideration of beliefs. For example, 4D/RCS [Albus, 2001] uses a limited capacity buffer, allowing only a fixed number of assumptions to be asserted at any one time. ACT-R [Anderson & Lebiere, 1998] employs sub-symbolic activation and decay mechanisms to manage assertions. By making such design decisions explicit in this analysis, we hope to facilitate a discussion of the trade-offs in these decisions among different approaches, and to make it more clear how to incorporate mechanisms from one architecture into another architecture. For example, the activation and decay mechanisms of ACT-R have recently been incorporated into a hybrid architecture integrating elements of ACT-R, Soar, and EPIC (EASE) [Chong & Wray, in press]. This architecture uses Soar's reason maintenance to manage the assertion and retraction of beliefs, but ACT-R's activation and decay mechanisms to manage assumptions.

3.3 Desires

BDI is the only framework that clearly separates desires from "normal" active goals (below). Desires allow an agent to monitor goals that it has chosen not to pursue explicitly. An additional advantage is that, by making its desires explicit, an agent can communicate those desires to another agent that may be able to achieve them [Wooldridge, 2000]. Soar and GOMS do not specify that desires should exist, how they should be represented, or how they should influence reasoning. BDI manages commitment to desires through logical inference.

3.4 Active Goals

A hallmark of intelligent behavior is the ability to commit to a particular set of concerns and then pursue them [Bratman, 1987; Newell, 1990]. Most agent frameworks support explicit representation of the goals an agent has committed to pursue. However, the agent literature is somewhat inconsistent in its use of descriptive terms relevant to goals. Wooldridge [2000] uses the term "intentions" to label what we term *active goals*. In contrast, some implementations of BDI do not represent active goals distinctly from the selected plans that would achieve these goals. In such systems, *selected plans* are "intentions", but there is no explicit representation of an active goal apart from the plan. In Soar, an "intention" is the *next action selected* from a current plan (which may itself directly activate a goal). GOMS does not use the term "intention", but requires the explicit representation of goals. To avoid confusion, we call these commitments "active goals" to distinguish them from "inactive" desires and plans, and we avoid the term "intention" because of its overloaded usage in the literature and various implementations.

Agents require a process for selecting the current active goal (or set of goals). BDI and Soar include explicit processes for deliberate goal commitment, although goals can be implemented in a variety of ways in Soar. In GOMS, goal commitment occurs by invoking the plan associated with the goal. Although this is a deliberative process, it is not divided into separate steps as in the other frameworks. This is similar to some of the BDI implementations that implicitly attach active goals to plans (e.g., Winikoff et al, 2002).

An important area of research explores the question of when an agent should reconsider an active goal (e.g.,

Schutt and Wooldridge, 2001; Wray and Laird, 2003). The BDI framework uses evaluations of soundness to determine when an active goal should be reconsidered, and more recently, has been extended to include decision-theoretic processes for intention reconsideration [Schutt & Wooldridge, 2002]. Soar uses reason maintenance, which is essentially an implementation of the soundness criterion. GOMS uses selection rules to commit to a goal, but does not explicitly address later reconsidering a goal.

3.5 Plans

Once there is an active goal to pursue, the agent must commit to a plan of action. BDI and GOMS assume there is a plan library, or some other method for generating plans outside the basic agent framework. Soar does not require that an agent have an explicit representation of a plan. More commonly, Soar agents associate individual actions with goals (plan knowledge is implicit in the execution knowledge), or interleave planning and execution. Either way, Soar assumes that planning is a deliberative task requiring the same machinery as any other agent activity, and involving the same concerns of commitment and resource usage. However, as with any other unsupported base-level representation, Soar thus forces the agent developer to implement the planning algorithm and the representation of any plans.

GOMS and BDI do not specify plan languages, although their implementations do. Soar has nothing like the relatively rich GOMS and BDI plan languages, instead using its operators to implement simple types of commitment. The tradeoff is that complex plans are much easier for a developer to program in GOMS and BDI, but easier for a Soar agent to learn (because of the simpler, uniform target language). Developers of GOMS and BDI implementations must make decisions about plan languages, leading to non-uniform solutions from one implementation to another.

Plan commitment in BDI can be quite simple: plans can be selected via a lookup table indexed by goal [Wooldridge, 2000] or implied completely by goal selection, as in JAM. In sharp contrast, GOMS treats the choice of which method to choose to pursue a goal as a major element of the framework. Because Soar does not have an architectural notion of a plan, there is no plan-specific commitment mechanism. Soar also does not make an explicit distinction between plan generation/selection and plan execution. Creating (or finding) a plan involves a series of context-sensitive decisions and operations, just as executing a plan does.

An agent can consider abandoning its current plan, even when it has chosen to remain committed to its current goal [Wooldridge, 2000]. The frameworks here do not provide strong advice on when such a commitment should be given up; BDI and Soar at least dictate that any plan should be executed one action at a time, allowing reconsideration of the plan after each step (although the two disagree on how complex a single action can be). Frameworks that use explicit plans may provide support for abandoning a plan reactively (BDI) or ignore this problem completely (GOMS). Soar, because it does not require explicit plans, implicitly supports plan reconsideration, because there is no separate commitment to a plan in the first place. Thus, in Soar, an agent commits to one action at a time, rather than committing to a whole plan. The tradeoff is that Soar agents must include extra knowledge to remain committed to a particular course of action, and the implementation of this knowledge is up to individual agent developers.

Other approaches to plan maintenance include using "completable plans" [Gervasio and DeJong, 1994] and allowing agents to switch back and forth between two or more plans in support of multiple, orthogonal goals. Plan switching is clearly a requirement for knowledge-intensive agents in many complex domains, but none of the frameworks specify how switching must occur. For example, it is not clear whether any current implementations of BDI or GOMS support resumption of a partially executed plan (reentrant plan execution). Many Soar systems implement task switching, but they rely on extra knowledge coded by the agent developer. This area deserves future attention, because reentrant plans provide a key capability for knowledge-intensive intelligent agents over more brittle behavior systems.

3.6 Actions

An agent must eventually commit to some type of action relevant to its active goals. All three frameworks specify three types of actions: execute an output command, update the belief set, or commit to a new goal (or desire). GOMS and Soar define *operators* as the atomic level of action, allowing commitment and/or reconsideration for each plan action. As an alternative, BDI systems generally provide a plan language that is a complete programming language. Such languages provide powerful and flexible means of plan implementation, but may leave them outside the commitment regime of the framework. BDI dictates that reconsideration ought to occur after each plan step, but does not tightly constrain how much processing may occur in a single step. This imposes a tradeoff between ease of programming (BDI) and taking advantage of the uniformity of the framework's built-in processes (GOMS and Soar).

Soar uses actions to create assumptions in the belief set (thus, assumptions can only be the result of deliberate decision making). Tying assumptions to actions is an important issue. Automated, logical reason maintenance is attractive, but there are pragmatic resource limitations for updating an agent's beliefs. Ideally, a rational agent would compute all relevant entailments from any input. But in complex environments, this is simply not computationally feasible (e.g., [Hill, 1999]).

3.7 Outputs

The ultimate level of commitment is to initiate activity in the environment. To accomplish this, an agent invokes

an output system. All three frameworks assume that output has to happen somehow, but do not impose strong constraints on the representation of output. BDI leaves output decisions up to the designer of the plan language. GOMS insists that primitive operators produce any output. As with perception, Soar requires that a motor command be represented in Soar's belief language, which allows the agent to reason about and execute output commands using the same agent knowledge. This is a key aspect of Soar's approach to integrated reasoning and execution.

Systems that use completable plans may include conditional outputs (possibly in addition to other conditional actions). Soar conditionally decodes actions using the same computational processes that it uses for maintaining entailments. The instantiated completion of an action is analogous to the automated elaboration of beliefs. Each framework supports methods for executing completable plans; some depending on plan language choices. Soar specifies what the plan language has to be, and therefore also specifies how plan completion occurs.

4 Conclusions

The research communities that use agent frameworks continue to explore the issues that limit and inform the development of highly competent intelligent agents *within* their integrated frameworks (e.g., Harland and Winikoff, 2001; Jones and Laird, 1997). However, too little attention has been paid to understanding the commonalities and differences across frameworks. We have attempted to contribute to this larger discussion by reviewing the directly supported representations and processes in three broadly differing agent frameworks. Continuing to identify and develop representations and processes for agents is an important research goal. Increasingly, researchers are attending to processes necessary for social agents, including normatives, values, obligations and teamwork. However, there are additional intra-agent representations and processes that the frameworks discussed here do not directly support and that may be so widely necessary that they should be considered base-level representations. Examples include deliberate attention [Hill, 1999], parallel active goals [Jones et al., 1994], and architectural support for managing resource limitations and conflicts [Meyer and Kieras, 1997]. Learning is also important for long-lived knowledge-intensive agents. The migration of knowledge into (and out of) long-term memory can also be studied in terms of representations, commitment, and reconsideration, resulting in a complex space of learning mechanisms (e.g., along dimensions of automatic vs. deliberate learning, or representations of procedural, declarative, and episodic memories). This analysis lays the groundwork for extending and unifying the basic level representations and processes needed for knowledge-intensive intelligent agents.

References

Albus, J. 2001. *Engineering of Mind: An Introduction to the Science of Intelligent Systems*. 2001: John Wiley and Sons.

Anderson, J. R. and Lebiere, C. 1998. *Atomic Components of Thought.* Hillsdale, NJ: Lawrence Erlbaum.

Bratman, M. E. 1987. *Intentions, plans, and practical reason.* Cambridge, MA: Harvard University Press

Bratman, M. E., Israel, D. J., and Pollack, M. E. 1988. Plans and resource-bounded practical reasoning. *Computational Intelligence* 4:349-355.

Card, S., Moran, T., and Newell, A. 1983. *The psychology of human-computer interaction*. Hillsdale, NJ: Lawrence Erlbaum.

Chong, R.S. 2003. *The addition of an activation and decay mechanism to the Soar architecture*. in *Fifth International Conference on Cognitive Modeling*. Bamberg, Germany: University of Bamberg.

Chong, R. S., & Wray, R. E. In press. *Constraints on Architectural Models: Elements of ACT-R, Soar and EPIC in Human Learning and Performance*. In K. Gluck & R. Pew (Eds.), *Modeling Human Behavior with Integrated Cognitive Architectures: Comparison, Evaluation, and Validation*.

d'Inverno, M., Kinny, D., Luck, M., and Wooldridge, M. 1997. A formal specification of dMARS. In Singh, A. Rao, and M. J. Wooldridge (eds.), *Intelligent Agents IV* (LNAI Volume 1365), 155-176. Berlin: Springer Verlag.

Forbus, K. D., and deKleer, J. 1993. *Building problem solvers.* Cambridge, MA: MIT Press.

Freed, M. A., and Remington, R. W. 2000. Making human-machine system simulation a practical engineering tool: An Apex overview. *Proceedings of the 2000 International Conference on Cognitive Modeling.*

Gardenfors, P. 1988. *Knowledge in flux*. Cambridge, MA: MIT Press.

Georgeff, M. P., and Lansky, A. L. 1987. Reactive reasoning and planning. *Proceedings of the Sixth National Conference on Artificial Intelligence,* 677-682. Menlo Park, CA: AAAI Press.

Georgeff M. P. and Rao, A. S. 1996. *A profile of the Australian AI Institute*. IEEE Expert, 11(6): 89-92.

Gervasio, M. T., and DeJong, G. F. 1994. An incremental approach for completable planning. *Machine Learning: Proceedings of the Eleventh National Conference*, 78-86. San Mateo, CA: Morgan Kaufmann.

Gray, W. D., John, B. E., and Atwood, M. E. 1993. Project Ernestine: Validating a GOMS analysis for predicting and explaining real-world performance. *Human-Computer Interaction* 8(3): 237-309.

Harland, J., and Winikoff, M. 2001. Agents via mixed-mode computation in linear logic: A proposal. *Proceedings of the ICLP'01 Workshop on Computational Logic in Multi-Agent Systems*. Paphos, Cyprus.

Hill, R. 1999. Modeling perceptual attention in virtual humans. Proceedings of the Eighth Conference on Computer Generated Forces and Behavior Representation. Orlando, FL.

Huber, M. J. 1999. JAM: A BDI-theoretic mobile agent architecture. *Proceedings of the Third International Conference on Autonomous Agents,* 236-243.

John, B. E., Vera, A. H., and Newell, A. 1994. Toward real-time GOMS: A model of expert behavior in a highly interactive task. *Behavior and Information Technology* 13(4): 255-267.

Jones, R. M., Laird, J. E., Nielsen, P. E. at al. 1999. Automated Intelligent Pilots for Combat Flight Simulation. *AI Magazine.* 20(1): 27-42.

Jones, R. M., and Laird, J. E. 1997. Constraints on the design of a high-level model of cognition. *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*.

Jones, R. M., Laird, J. E., Tambe, M., and Rosenbloom, P. S. 1994. Generating behavior in response to interacting goals. *Proceedings of the Fourth Conference on Computer Generated Forces and Behavior Representation,* 325-332. Orlando, FL.

Kieras, D. E. 1997. A guide to GOMS model usability evaluation using NGOMSL. In M. Helander, T. Landauer, and P. Prabhu (eds.), *Handbook of human-computer interaction,* 733-766. North-Holland.

Kieras, D. E., Wood, S. D., Abotel, K., and Hornof, A. 1995. GLEAN: A computer-based tool for rapid GOMS model usability evaluation of user interface designs. *Proceedings of the ACM symposium on User Interface Software and Technology.*

Laird, J. E., Newell, A., and Rosenbloom, P. S. 1987. Soar: An architecture for general intelligence. *Artificial Intelligence* 33(1): 1-64.

Meyer, D., and Kieras, D. 1997. EPIC: A computational theory of executive cognitive processes and multiple-task performance: Part 1. *Psychological Review* 104:3-65,

Newell, A. 1980. Physical symbol systems. *Cognitive Science* 4:135-183.

Newell, A. 1982. The knowledge level. *Artificial Intelligence* 18(1): 87-127.

Newell, A. 1990. *Unified theories of cognition.* Cambridge, MA: Harvard University Press.

Payne, T. R., Singh, R., & Sycara, K. 2002. Calendar agents on the semantic web. *IEEE Intelligent Systems,* 17(3): 84-86,

Peck, V. A., and John, B. E. 1992. Browser-Soar: A computational model of a highly interactive task. In P. Bauersfeld, J. Bennett, and G. Lynch (eds.), *ACM CHI '92 Conference on Human Factors in Computing Systems*, 165-172. New York: ACM Press.

Rao, A., and Georgeff, M. 1995, BDI agents: From theory to practice. *Proceedings of the First Intl. Conference on Multiagent Systems*. San Francisco.

Schutt, M. C., and Wooldridge, M. 2001. Principles of intention reconsideration. *Proceedings of the Fifth International Conference on Autonomous Agents,* 209-216. New York: ACM Press.

Taylor, G., Frederiksen, R., Vane, R. and Waltz, E. 2004. Agent-based simulation of geo-political conflict. 2004 *Innovative Applications of Artificial Intelligence.* San Jose.

Winikoff, M., Padgham, L., Harland, J.,Thangarajah, J. *Declarative and Procedural Goals in Intelligent Agent Systems*. In proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR2002), April 22-25, 2002, Toulouse, France.

Wooldridge, M. 2000. *Reasoning about Rational Agents.* Cambridge, MA: MIT Press.

Wray, R.E. and J.E. Laird. 2003. *An architectural approach to consistency in hierarchical execution.* Journal of Artificial Intelligence Research. **19**: p. 355-398.