Chapter 9.

A "Conciousness"-Based Architecture for a Functioning Mind

Stan Franklin
The University of Memphis, USA

Abstract

Here we describe an architecture for an autonomous software agent designed to model a broad spectrum of human cognitive and affective functioning. In addition to featuring "consciousness," the architecture accommodates perception, several forms of memory, emotions, action-selection, deliberation, ersatz language generation, several forms of learning, and metacognition. One such software agent, IDA, embodying much of this architecture, is up and running. IDA's "consciousness" module is based on global workspace theory, allowing it to select relevant resources with which to deal flexibly with both exogenous and endogenous stimuli. Within this architecture, emotions implement IDA's drives, her[1] primary motivations. Offering one possible architecture for a fully functioning artificial mind, IDA constitutes an early attempt at the exploration of design space and niche space. The design of the IDA architecture spawns hypotheses concerning human cognition and affect that can serve to guide the research of cognitive scientists and neuroscientists. One such hypothesis is that consciousness is discrete.

Introduction

What is a mind? I have maintained for years, and still do, that the most useful way to look at a mind is as a control structure for an autonomous agent (see the next section). The continuing task of a mind is to produce the agent's next action, to answer the only really significant question there is—what shall I do next (Franklin, 1995). Any theory specifying how to go about answering this question is a theory of mind.[2] A theory is computationally plausible if it can be implemented or modeled on a computer, very likely on a virtual machine running on a computer (Sloman & Chrisley, 2003). If our theory is to be implemented or modeled on a computer, we must have in hand a computationally plausible architecture with which to implement or model it. If we have succeeded in implementing our architecture on a computer so that it supports our theory of mind on an autonomous agent, we have produced an artificial mind.

This chapter is devoted primarily to the description of one such, complex, functioning artificial mind, and to some of the hypotheses about human affect and cognition that are derived from it. This artificial mind is the control structure of an autonomous software agent, IDA (Franklin, Kelemen, & McCauley, 1998; Franklin, 2001). IDA's architecture implements global workspace theory, a theory of mind (Baars, 1988, 1997, 2002). It can be seen as an early contribution to the exploration of design space and niche space (Sloman, 1998).

Autonomous Agents

Artificial intelligence pursues the twin goals of understanding human intelligence and of producing intelligent software and artifacts. Designing, implementing, and experimenting with autonomous agents furthers both of these goals in a synergistic way (Franklin, 1997). An *autonomous agent* (Franklin & Graesser, 1997) is a system situated in, and part of, an environment, which senses that environment, and acts on it, over time, in pursuit of its own agenda. In biological agents, this agenda arises from evolved drives and their associated goals; in artificial agents from drives and goals built in by its designer. Such drives that act as motive generators (Sloman, 1987) must be present, whether explicitly represented or expressed causally. The agent also acts in such a way as to possibly influence what it senses at a later time. In other words, it is structurally coupled to its environment (Maturana, 1975; Maturana et al., 1980). Biological examples of autonomous agents include humans and most animals. Nonbiological examples include some mobile robots and various computational agents, includ-

ing artificial life agents, software agents, and many computer viruses. We will be concerned with autonomous software agents designed for specific tasks and "living" in real-world computing systems, such as operating systems, databases, or networks.

Such autonomous software agents serve to spawn hypotheses about human cognition and affect that can serve to guide the research of cognitive scientists and neuroscientists. Each design decision taken for the agent translates into such a hypothesis about human cognitive or affective functioning (Franklin, 1997). Thus, in addition to their practical function, such agents can further the interests of science.

Roboticists often claim that autonomous software agents are not embodied in that they typically do not have to deal with the physics of the real world (Prem, 1997). However, some software agents, including our IDA, negotiate with humans in a real-world environment. They both causally affect the real, physical world and are affected by it. For this to happen, they, in some sense, must be embodied.

Global Workspace Theory

The material in this section relates to Baars' two books (1988, 1997) and superficially describes his global workspace theory of consciousness. In this theory, Baars, along with many others (for example, Minsky, 1985; Ornstein, 1986; Edelman, 1987; Jackson, 1987), postulates that human cognition is largely implemented by a multitude of relatively small, special-purpose processes, almost always subconscious. (It is a multiagent system.) Communication between them is rare and over a narrow bandwidth. Coalitions of such processes find their way into a global workspace (and thus into consciousness). This limited capacity workspace serves to broadcast the message (contents) of the coalition to all the subconscious processors in order to recruit other processors to join in the handling of the current novel situation or in solving the current problem. Thus, consciousness in this theory allows us to deal with novelty or problematic situations that cannot be dealt with efficiently, or at all, by automatized, subconscious processes. In particular, consciousness provides access to appropriately useful resources, thereby solving the *relevance problem*, that is, the problem of identifying those resources that are relevant to the current situation.

All of this takes place under the auspices of contexts: goal contexts, perceptual contexts, conceptual contexts, and cultural contexts. Baars uses goal hierarchies, dominant goal contexts, a dominant goal hierarchy, dominant context

hierarchies, and lower level context hierarchies. Each context is a coalition of processes. Though contexts are typically subconscious, they strongly influence conscious processes.

Baars postulated that learning results simply from conscious attention; that is, that consciousness is sufficient for learning. There is much more to the theory, including attention, action selection, emotion, voluntary action, metacognition, and a sense of self. I think of it as a high-level theory of cognition and of affect.

"Conscious" Software Agents

A "*conscious"* software agent is defined to be an autonomous software agent that implements global workspace theory. (No claim of sentience or phenomenal consciousness is being made, hence, the scare quotes.) I believe that "conscious" software agents have the potential to play a synergistic role in both cognitive theory and intelligent software. Minds can be viewed as control structures for autonomous agents (Franklin, 1995). A theory of mind constrains the design of a "conscious" agent that implements that theory. While a theory is typically abstract and only broadly sketches an architecture, an implemented, computational design provides a fully articulated architecture and a complete set of mechanisms. This architecture and set of mechanisms provides a richer, more concrete, and more decisive theory. Moreover, every design decision taken during an implementation furnishes a hypothesis about how human minds work. These hypotheses may motivate experiments with humans and other forms of empirical tests, thereby providing direction to research in cognitive science and neuroscience. Conversely, the results of such experiments motivate corresponding modifications of the architecture and mechanisms of the software agent. In this way, the concepts and methodologies of cognitive science and of computer science will work synergistically to enhance our understanding of mechanisms of mind (Franklin, 1997).

"Conscious" Mattie

"Conscious" Mattie (CMattie) was our first "conscious" software agent. She is a clerical agent devoted to publicizing mostly weekly seminars on various subjects in a Mathematical Sciences Department (McCauley & Franklin, 1998; Ramamurthy et al., 1998; Zhang et al., 1998; Bogner et al., 2000). She composes and e-mails weekly seminar announcements, having communicated by e-mail

with seminar organizers and announcement recipients in natural language. She maintains her mailing list, reminds organizers who are late with their information, and warns of space and time conflicts. There is no human involvement other than these e-mail messages. CMattie's cognitive modules include perception, learning, action selection, associative memory, "consciousness," emotion, and metacognition. Her emotions influence her action selection. Her mechanisms include variants and extensions of Maes' behavior nets (1989), Hofstadter and Mitchell's Copycat architecture (1994), Jackson's pandemonium theory (1987), Kanerva's sparse distributed memory (1988), and Holland's classifier systems (Holland, 1986).

CMattie will play only a minor role in what follows. The brief description above is included for two reasons. Several of the references given in the context of IDA's modules will, in fact, describe similar modules in CMattie rather than IDA modules. In these cases, the descriptions therein will be mostly accurate when applied to the corresponding IDA module. Second, CMattie constitutes an instructive example relating to the exploration of design space (Sloman, 1998). CMattie's cognitive processes are reactive and metacognitive without being deliberative, demonstrating that this combination is possible if only in an impoverished way.

IDA

IDA (Intelligent Distribution Agent) is a "conscious" software agent that was developed for the U.S. Navy (Franklin et al., 1998). At the end of each sailor's tour of duty, he or she is assigned to a new billet. This assignment process is called distribution. The Navy employs some 280 people, called detailers, full time, to effect these new assignments. IDA's task is to facilitate this process by automating the role of detailer.

IDA's task presents both communication problems and action selection problems involving constraint satisfaction. It must communicate with sailors via e-mail and in natural language, understanding the content and producing lifelike responses. Sometimes IDA will initiate conversations. She must access a number of databases, again understanding the content. She must see that the Navy's needs are satisfied, for example, the required number of sonar technicians on a destroyer with the required types of training. In doing so, IDA must adhere to a number of Navy policies. She must hold down moving costs. And, IDA must cater to the needs and desires of sailors as well as is possible. This includes negotiating with the sailor via an e-mail correspondence in natural language. IDA's architecture and mechanisms are largely modeled after those of CMattie,

though they are more complex. In particular, IDA employs deliberative reasoning in the service of action selection, where CMattie was able to do without.

Before going further, it is important that we distinguish between IDA as a computational model and as a conceptual model. The computational IDA is a running piece of Java code, an actual software agent. The conceptual IDA model includes everything in the computational model with relatively minor changes. It also includes, however, additional functionality that has been designed but not yet implemented. In what follows, I will try to carefully note capabilities not yet implemented. Unless otherwise stated, descriptions will be of the computational model.

As we hypothesize that humans also do, the IDA model runs in a rapidly continuing sequence of partially overlapping cycles, called cognitive cycles (Baars & Franklin, 2003). These cycles will be discussed in detail below, after the IDA architecture and its mechanisms are described.

"Conscious" Software Architecture and Mechanisms

The IDA architecture is partly symbolic and partly connectionist, at least in spirit. Although there are no artificial neural networks as such, spreading activation abounds. The mechanisms used in implementing the several modules have been inspired by a number of different new AI techniques (Hofstadter & Mitchell, 1994; Holland, 1986; Jackson, 1987; Kanerva, 1988; Maes, 1989; Minsky, 1985; Valenzuela-Rendon, 1991). The architecture is partly composed of entities at a relatively high level of abstraction, such as behaviors, message-type nodes, emotions, etc. (all discussed in this chapter), and partly of low-level codelets.

Each codelet is a small piece of code performing a simple, specialized task. They correspond to Baars' processors in global workspace theory (1988). Most codelets are, like demons in an operating system, always watching for a situation to arise, making it appropriate to act. Codelets come in many varieties: perceptual codelets, information codelets, attention codelets, behavior codelets, expectation codelets, etc. (all described in this chapter). Though most codelets subserve some high-level entity, many codelets work independently. Codelets do almost all the work. IDA can almost be viewed as a multiagent system, though not in the usual sense of the term.

As noted above, most of IDA's various entities, both high-level entities and codelets, carry and spread some activation. Such activation typically hopes to measure some sort of strength or relevance. Unless told otherwise, it is safe to

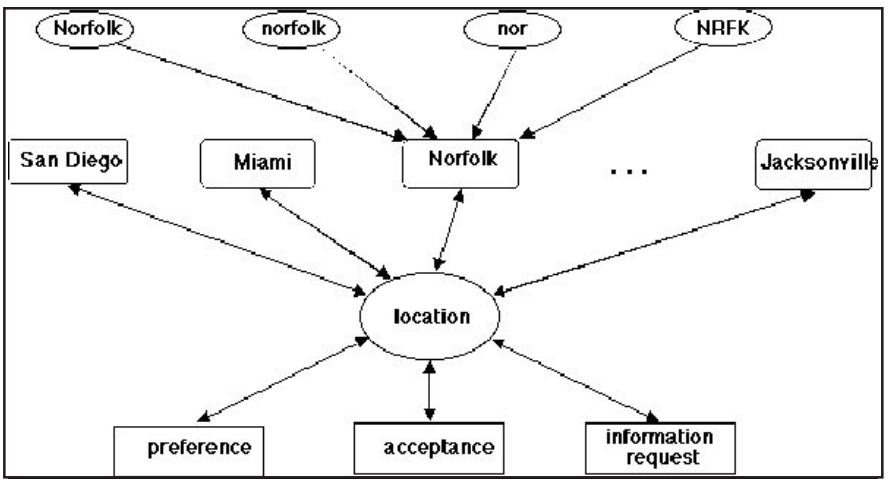
assume that every activation decays over time. Finally, note that though the IDA architecture is conveniently described in terms of modules, it is, in fact, tightly connected. Like the brain, the IDA architecture is both modular and highly interconnected.

Perception

IDA perceives both exogenously and endogenously (Zhang et al., 1998). The stimuli of her single sense are strings of characters. We use Barsalou's perceptual symbol systems as a guide (1999). The perceptual knowledge base of this agent, called perceptual memory, takes the form of a semantic net with activation called the slipnet. The name is taken from the Copycat architecture that employs a similar construct (Hofstadter & Mitchell, 1994). Nodes of the slipnet constitute the agent's perceptual symbols, representing individuals, categories, and higher-level ideas and concepts. A link of the slipnet represents a relation between its source node and its sink node (see Figure 1).

An incoming stimulus, say an e-mail message, is descended upon by a hoard of perceptual codelets. Each of these codelets is looking for some particular string or strings of characters, say one of the various forms of the name of the city of Norfolk. Upon finding an appropriate character string, the codelet will activate an appropriate node or node in the slipnet. The slipnet will eventually settle down. Nodes with activations over threshold and their links are taken to be the constructed meaning of the stimulus. Pieces of the slipnet containing nodes and

Figure 1. A portion of the Slipnet



links, together with perceptual codelets with the task of copying the piece to working memory constitute Barsalou's perceptual symbol simulators.

Memory

Both CMattie and IDA employ sparse distributed memory (SDM) as their major associative memories (Anwar, Dasgupta, & Franklin, 1999; Anwar & Franklin, 2003). SDM is a content addressable memory that, in many ways, is an ideal computational mechanism for use as a long-term associative memory (Kanerva, 1988). Content addressable means that items in memory can be retrieved by using part of their contents as a cue, rather than having to know the item's address in memory.

The inner workings of SDM rely on large binary spaces, that is, spaces of vectors containing only zeros and ones, called bits. These binary vectors, called words, serve as both the addresses and the contents of the memory. The dimension of the space determines the richness of each word. These spaces are typically far too large to implement in any conceivable computer. Approximating the space uniformly with some manageable number of actually implemented, hard locations surmounts this difficulty. The number of such hard locations determines the carrying capacity of the memory. Features are represented as one or more bits. Groups of features are concatenated to form a word. When writing a word to memory, a copy of the word is placed in all close enough hard locations. When reading a word, a close enough cue would reach all close enough hard locations and get some sort of aggregate or average out of them. As mentioned above, reading is not always successful. Depending on the cue and the previously written information, among other factors, convergence or divergence during a reading operation may occur. If convergence occurs, the pooled word will be the closest match (with abstraction) of the input reading cue. On the other hand, when divergence occurs, there is no relation, in general, between the input cue and what is retrieved from memory.

SDM is much like human long-term declarative memory. A human often knows what he or she does or does not know. If asked for a telephone number I have once known, I may search for it. When asked for one I have never known, an immediate "I don't know" response ensues. SDM makes such decisions based on the speed of initial convergence. The reading of memory in SDM is an iterative process. The cue is used as an address. The content at that address is read as a second cue, and so on, until convergence, that is, until subsequent contents look alike. If it does not quickly converge, an "I don't know" is the response. The "on the tip of my tongue phenomenon" corresponds to the cue having content just at the threshold of convergence. Yet another similarity is the power of rehearsal, during which an item would be written many times and, at

each of these, to a thousand locations—that is the *distributed* part of sparse distributed memory. A well-rehearsed item can be retrieved with smaller cues. Another similarity is interference, which would tend to increase over time as a result of other similar writes to memory. The IDA conceptual model uses variants of SDM to implement both transient episodic memory and declarative memory (Franklin et al., in review; Ramamurthy, D'Mello, & Franklin, to appear).

"Consciousness"

The "consciousness" modules in CMattie and IDA are almost identical. In both architectures, the processors postulated by global workspace theory are implemented by codelets, small pieces of code. These are specialized for some simple task and often play the role of a demon waiting for an appropriate condition under which to act. The apparatus for producing "consciousness" consists of a coalition manager, a spotlight controller, a broadcast manager, and a collection of attention codelets that recognize novel or problematic situations (Bogner, 1999; Bogner et al., 2000). Each attention codelet keeps a watchful eye out for some particular situation to occur that might call for "conscious" intervention. Upon encountering such a situation, the appropriate attention codelet will be associated with the small number of information codelets that carry the information describing the situation. This association should lead to the collection of this small number of codelets, together with the attention codelet that collected them, becoming a coalition. Codelets also have activations. Upon noting a suitable situation, an attention codelet will increase its activation as a function of the match between the situation and its preferences. This allows the coalition, if one is formed, to compete for "consciousness."

The coalition manager is responsible for forming and tracking coalitions of codelets. Such coalitions are initiated on the basis of the mutual associations between the member codelets. During any given cognitive cycle, one of these coalitions finds its way to "consciousness," chosen by the spotlight controller, who picks the coalition with the highest average activation among its member codelets. Global workspace theory calls for the contents of "consciousness" to be broadcast to each of the codelets. The broadcast manager accomplishes this.

Action Selection

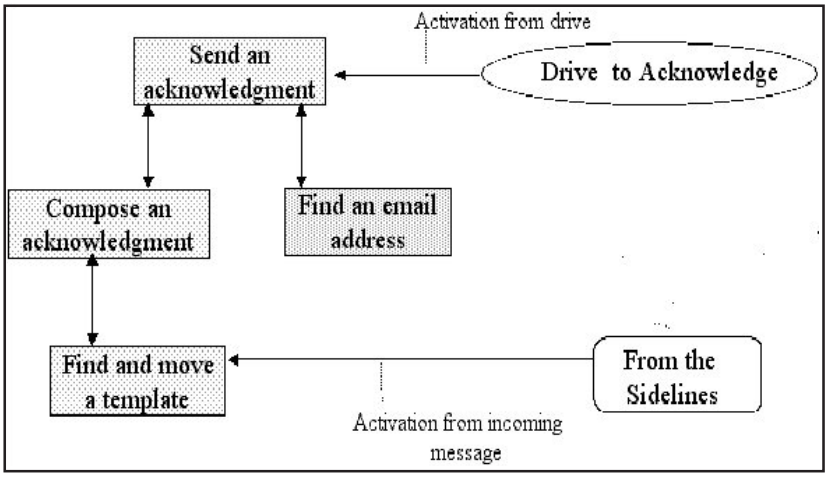
CMattie and IDA depend on an enhancement of Maes' behavior net (1989) for high-level action selection in the service of built-in drives (Song & Franklin, 2000; Negatu & Franklin, 2002). Each has several distinct drives operating in parallel

and implemented in the IDA conceptual model by feelings and emotions. These drives vary in urgency as time passes and the environment changes. The goal contexts of global workspace theory are implemented as *behaviors* in the IDA model. Behaviors are typically mid-level actions, many depending on several behavior codelets for their execution. A behavior net is composed of behaviors and their various links. A behavior looks very much like a production rule, having preconditions as well as additions and deletions. A behavior is distinguished from a production rule by the presence of an activation, which is a number indicating some kind of strength level. Each behavior occupies a node in a digraph (directed graph). The three types of links of the digraph are completely determined by the behaviors. If a behavior *X* will add a proposition *b*, which is on behavior *Y*'s precondition list, then put a successor link from *X* to *Y*. There may be several such propositions, resulting in several links between the same nodes. Next, whenever you put in a successor going one way, put in a predecessor link going the other. Finally, suppose you have a proposition *m* on behavior *Y*'s delete list that is also a precondition for behavior *X*. In such a case, draw a conflictor link from *X* to *Y*, which is to be inhibitory rather than excitatory.

As in connectionist models, this digraph spreads activation. The activation comes from four sources: from activation stored in the behaviors, from the environment, from drives (through feelings and emotions in the IDA conceptual model), and from internal states. The environment awards activation to a behavior for each of its true preconditions. The more relevant it is to the current situation, the more activation it is going to receive from the environment. This source of activation tends to make the system opportunistic. Each drive awards activation to every behavior that, by being active, will satisfy that drive. This source of activation tends to make the system goal directed. Certain internal states of the agent can also send activation to the behavior net. This activation, for example, might come from a coalition of behavior codelets responding to a "conscious" broadcast. Finally, activation spreads from behavior to behavior along links. Along successor links, one behavior strengthens those behaviors with preconditions that it can help fulfill by sending them activation. Along predecessor links, one behavior strengthens any other behavior with an add list that fulfills one of its own preconditions. A behavior sends inhibition along a conflictor link to any other behavior that can delete one of its true preconditions, thereby weakening it. Every conflictor link is inhibitory. A behavior is *executable* if all of its preconditions are satisfied. To be acted upon, a behavior must be executable, must have activation over threshold, and must have the highest such activation. Behavior nets produce flexible, tunable action selection for these agents.

Behaviors in these agents almost always operate as part of behavior streams, which correspond to goal context hierarchies in global workspace theory. Visualize a behavior stream as a subgraph of the behavior net, with its nodes connected by predecessor links (Figure 2). A behavior stream is sometimes a

Figure 2. Behavior stream



sequence, but not always. It can fork in either a forward or backward direction. A behavior stream can be usefully thought of as a partial plan of action.

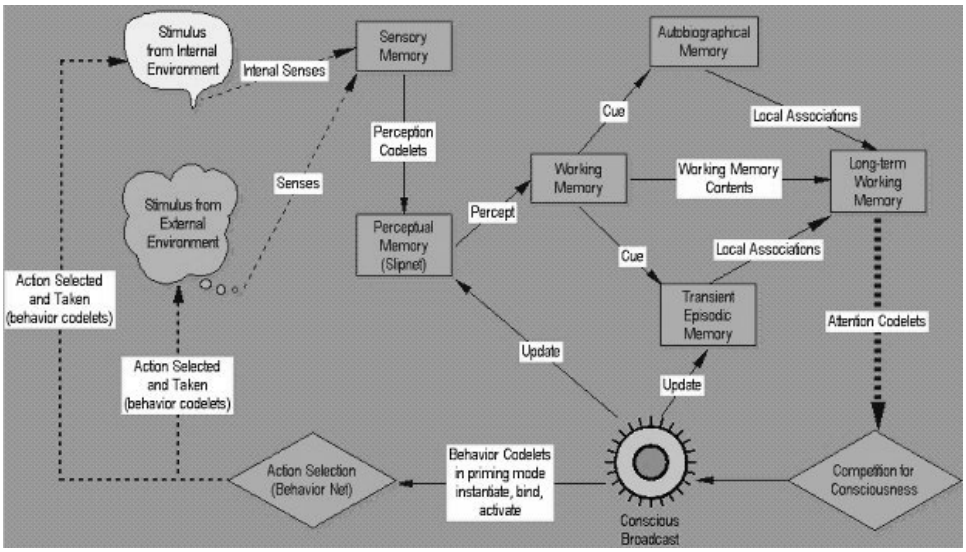
Constraint Satisfaction

At the heart of IDA's task of finding new jobs for sailors lies the issue of constraint satisfaction. Not only must IDA consider the preferences of the sailor, she must also see that the requirements of an individual job are met, and simultaneously adhere to the policies of the Navy. Taking such issues into consideration, IDA's constraint satisfaction module is designed to provide a numerical measure of the fitness for a particular job for a particular sailor.

Given a specified issue such as a sailor preference, a particular Navy policy, or specific job requirement, a function is defined that provides a numerical measure of the fitness of this job for this sailor with respect to this particular issue. Computationally, these functions are diverse and often nonlinear. Most take their input from information from the sailor's personnel record or from the job requisition list that has already been written to IDA's working memory.

To find a common currency for these various issues, we laboriously found a set of weights for the issues, measuring their relative importance (Keleman et al., 2002). With these weights in hand, the weighted sum over the various issues forms a linear functional that provides the desired numerical measure of fitness. The calculating of such a fitness value requires several of IDA's cognitive cycles (see the following) to process each term in the linear functional.

Figure 3. IDA's cognitive cycle



This is an oversimplified account; for example, there are multiplicative terms dealing with hard constraints. I chose not to describe IDA's constraint satisfaction module in more detail, because it is so restricted to her particular, practical domain. In some other application of the IDA architecture, the constraint satisfaction module would not likely appear.

Deliberation

Action selection via the behavior net suffices for CMattie due to her relatively constrained domain. IDA's domain is more complex and requires deliberation in the sense of creating possible scenarios and partial plans of actions and then choosing between them. For example, suppose IDA is considering a ranked list of several possible jobs for a given sailor produced by her constraint satisfaction module, all seemingly suitable. IDA must construct a scenario for at least one of these possible billets. In each scenario, the sailor leaves his or her current position during a certain time interval, spends a specified length of time on leave, possibly reports to a training facility on a certain date, and arrives at the new billet within a given time window. Such scenarios are judged on how well they fit the temporal constraints and on moving and training costs.

As in humans, deliberation is mediated by the "consciousness" mechanism. Imagine IDA in the context of a behavior stream with a goal to construct a

scenario to help evaluate a particular job for a particular sailor. She must first decide on a departure date within an allowable window, the first event of the scenario. Then, events for travel time (often in more than one segment), leave time (occasionally in several segments), training time (with specified dates), and arrival date must be decided upon, again within an appropriate window. If the first try does not work, IDA typically starts over with a suitably adjusted departure date. If still unsuccessful after several tries, IDA will give up on that particular job and go on to another. When successful, the job in question is so marked in working memory and becomes a candidate for voluntary selection (see below) to be offered to the sailor. Each step in this process will require several cognitive cycles, as described below. Thus, IDA is capable of temporal deliberation.

Voluntary Action

Deliberation is also used in IDA to implement voluntary action in the form of William James' ideomotor theory as prescribed by global workspace theory (Baars, 1988, Chapter VII). Suppose scenarios have been constructed for several of the more suitable jobs. An attention codelet spots one that it likes, possibly due to this codelet's predilection for, say, low moving costs. The act of an attention codelet's bringing one of these candidates to consciousness serves to propose it. This is James' idea popping into mind. If no other attention codelet brings an objection to consciousness or proposes a different job, a timekeeper codelet assigned the particular task of deciding will conclude, after a suitable time having passed, that the proposed job will be offered, and starts the process by which it will be so marked in working memory. Objections and proposals can continue to come to consciousness, but the patience of the timekeeper codelet dampens as time passes. Also, the activation of a given attention codelet tends to diminish after winning a competition for consciousness in any given cognitive cycle. This lessening makes it less likely that this particular attention codelet will be successful in proposing, objecting, or supporting in the near future. This diminishing of patience and activation serve to prevent continuing oscillations in the voluntary action selection process.

Language Generation

IDA's language generation module follows the same back and forth to "consciousness" routine carried out over a number of cognitive cycles. For example, in composing a message offering a sailor a choice of two billets, an attention codelet would bring to "consciousness" the information that this type of message

was to be composed and the sailor's name, pay grade, and job description. After the "conscious" broadcast and the involvement of the behavior net as described above, a script containing the salutation appropriate to a sailor of that pay grade and job description would be written to the working memory. Another attention codelet would bring this salutation to "consciousness," along with the number of jobs to be offered. The same process would result in an appropriate introductory script being written below the salutation. Continuing in this manner, filled in scripts describing the jobs would be written, and the message would be completed. Note that different jobs may require different scripts. The appeal to "consciousness" results in some version of a correct script being written.

The mediation by the "consciousness" mechanism, as described in the previous paragraphs, is characteristic of IDA. The principle is that IDA should use "consciousness" whenever a human detailer would be conscious in the same situation. For example, IDA could readily recover all the needed items from a sailor's personnel record subconsciously with a single behavior stream. But, a human detailer would be conscious of each item individually. Hence, according to our principle, so must IDA be "conscious" of each retrieved personnel data item.

This completes our description of the IDA computational model, which implements small parts of what is described below as part of the IDA conceptual model. We now have more than enough of the architecture in hand to discuss the cognitive cycle that derives from the IDA model.

IDA Cognitive Cycle

As noted above, the primary question facing the control system (mind) of any autonomous agent at any moment is, "What do I do now?" IDA answers this question in a moment-to-moment fashion by means of a continuing repetition of her cognitive cycle.[3] One can usefully decompose this cognitive cycle into the nine steps described just below. We choose to follow the computer science fashion (input, processing, output) and the psychological fashion (stimulus, cognition, response) by beginning the cycle with perception and ending it with an action. Note that many, if not most, of our actions aim at choosing the next sensory experience, suggesting that the cycle might well be regarded a beginning with an action. This is also true of some of IDA's actions, particularly her consultation of various Navy databases. Because we hypothesize that we humans employ just such a cognitive cycle (Franklin et al., In review), our description will include elements postulated for humans that are not present in IDA. These will be noted. The description will also mention as yet unimplemented

capabilities from the IDA conceptual model (see the following). Here is the description of IDA's cognitive cycle (Baars & Franklin, 2003):

1. **Perception.** Sensory stimuli, external or internal, are received and interpreted by perception assigning meaning. Note that this stage is subconscious.
   a) *Early perception*. Input arrives through senses. Specialized perceptual codelets descend on the input. Those that find features relevant to their specialty activate appropriate nodes in the slipnet (Perceptual Memory), a semantic net with activation. (For example, a codelet finding the string Norfolk would activate a Norfolk node.)
   b) *Chunk perception*. Activation passes from node to node in the slipnet (for example, from Norfolk to Location). The slipnet stabilizes, bringing about the convergence of streams from different senses (in humans) and chunking bits of meaning into larger chunks. These larger chunks, represented by meaning nodes in the slipnet, constitute the percept. (For example, Preference for a Particular Location.)
2. **Percept to preconscious buffer.** The percept, including some of the data plus the meaning, is stored in preconscious buffers of IDA's working memory. In humans, these buffers may involve visuospatial, phonological, and other kinds of information. (For example, in the computational IDA, a percept might include a nine-digit number tagged as a social security number, or a text string tagged as a location, or the recognition of a stated preference for a particular location.)
3. **Local associations.** Using the incoming percept and the residual contents of the preconscious buffers as cues, local associations are automatically retrieved from transient episodic memory (Taylor, 1999; Conway, 2001; Donald, 2001, p. 137) and from long-term declarative memory. The contents of the preconscious buffers together with the retrieved local associations from transient episodic memory and long-term associative memory. Together, these roughly correspond to Ericsson and Kintsch's long-term working memory (1995) and Baddeley's episodic buffer (Baddeley, 1993).
4. **Competition for consciousness.** Attention codelets, with the job of bringing relevant, urgent, or insistent events to consciousness, view long-term working memory. Some of them gather information, form coalitions, and actively compete for access to consciousness. (For example, in the computational IDA, an attention codelet may gather into a coalition an information codelet carrying the rate and name AS1 Kevin Adams, another carrying the location Norfolk, and yet another, the idea Preference for a

Particular Location.) The competition may also include attention codelets from a recent previous cycle. The activation of unsuccessful attention codelets decays, making it more difficult for them to compete with newer arrivals. However, the contents of unsuccessful coalitions remain in the preconscious buffer and can serve to prime ambiguous future incoming percepts. The same is true of contents of long-term working memory that are not picked up by an attention codelet.

5. **Conscious broadcast.** A coalition of codelets, typically an attention codelet and its covey of related information codelets carrying content, gains access to the global workspace and has its contents broadcast. (For example, IDA may become "conscious" of AS1 Kevin Adams preference for being stationed in Norfolk.) This broadcast is hypothesized to correspond to phenomenal consciousness[4] in humans. The current contents of consciousness are also stored in transient episodic memory. At recurring times, not part of a cognitive cycle, the contents of transient episodic memory are consolidated into long-term associative memory (Shastri, 2001, 2002). Transient episodic memory is an associative memory with a relatively fast decay rate (Conway, 2001). It is to be distinguished from autobiographical memory, a part of long-term declarative memory.
6. **Recruitment of resources.** Relevant behavior codelets respond to the conscious broadcast. These are typically codelets with variables that can be bound from information in the conscious broadcast. If the successful attention codelet was an expectation codelet calling attention to an unexpected result from a previous action, the responding codelets may be those that can help to rectify the unexpected situation. Thus, consciousness solves the relevancy problem in recruiting resources.
7. **Setting goal context hierarchy.** Some responding behavior codelets instantiate an appropriate behavior stream, if a suitable one is not already in place. Using information from the conscious broadcast, they also bind variables and send activation to behaviors. (In our running example, a behavior stream to find jobs to offer Kevin Adams might be instantiated. His preference for Norfolk might be bound to a variable.) Here, we assume that there is such a behavior codelet and behavior stream. If not, then nonroutine problem solving using additional mechanisms is called for.[5]
8. **Action chosen.** The behavior net chooses a single behavior (goal context) and executes it. This choice may come from the just instantiated behavior stream or from a previously active stream. The choice is affected by internal motivation (activation from goals), by the current situation, by external and internal conditions, by the relationship between the behaviors, and by the activation values of various behaviors. (In our example, IDA would likely choose to begin extracting useful data from Kevin Adam's personnel record in the Navy's database.)

9. **Action taken.** The execution of a behavior (goal context) results in the behavior codelets performing their specialized tasks, which may have external or internal consequences. This is IDA taking an action. The acting codelets also include an expectation codelet (see Step 6) with the task of monitoring the action and trying to bring to consciousness any failure in the expected results.

The IDA Conceptual Model

In addition to the IDA computational model that is now a successfully running software agent, several different additional capabilities for IDA have been designed and, sometimes, described. One felicitous feature of the IDA architecture is that it has been possible to design and add one new capability after another with literally no change in the basic processing structure of the system as outlined in the description of IDA's cognitive cycle just above. This makes us think we must be doing something right.

Some of the new capabilities are in the process of being implemented, while others await sufficient time, energy, and funding. This section is devoted to these new capabilities, including feelings and emotions, nonroutine problem solving, automization, transient episodic memory, metacognition, and several varieties of learning.

Feelings and Emotions

We view feelings and emotions as often suitable mechanisms for primary motivations (drives) in autonomous agents, including humans and many other animals (Sloman, 1987). Following Johnston, we conceive of emotions as special kinds of feelings—those with a necessary cognitive component (1999). While the computational IDA has directly implemented drives, IDA's predecessor, CMattie, had an implemented emotional apparatus (McCauley & Franklin, 1998). The newly designed feelings and emotions for IDA play a pervasive role in her entire cognitive process, as they do in humans (Damasio, 1999; Panksepp, 1998; Rolls, 1999). Here we will trace the many roles of feelings and emotions in the various steps of the cognitive cycle (Franklin & McCaulley, 2004).

In Step 1, IDA's perceptual memory, her slipnet, will have nodes for the various feelings, including emotions, as well as links connecting them to and from other nodes. Thus, the percept written to working memory in Step 2 may contain affective content. This affective content will serve as part of the cue used in Step

3 to retrieve local associations from transient episodic (see below) and declarative memories. The local associations so retrieved may contain accounts of affect associated with past events, as well as details of the event and of the action taken.

During the competition for "consciousness" in Step 4, attention codelets gathering information from long-term working memory will be influenced by affect. The stronger the affect, the higher the average activation of the coalition, and the more likely it is to win the competition. During the broadcast in Step 5, the various memories are updated, including the storing of affective information. In Step 6, various behavior codelets respond to the broadcast, and in Step 7, they instantiate behavior streams, bind variables, and activate behaviors. In addition to environmental activation, and based on feelings and emotions, these codelets will also activate behaviors that satisfy drives, thus implementing the drives. Those actions selected in Step 8 and performed in Step 9 are heavily influenced by the cumulative affect in the system.

Nonroutine Problem Solving

As humans, we have the ability to devise unexpected, and often clever, solutions to problems we have never before encountered. Sometimes they are even creative. According to global workspace theory, one principal function of consciousness is to recruit the resources needed for dealing with novel situations and for solving nonroutine problems. Though IDA's "consciousness" module is designed to deal intelligently with novel, unexpected, and problematic situations, the computational IDA is currently expected to handle only novel instances of routine situations. One message from a sailor asking that a job be found is much like another in content, even when received in natural language with no agreed upon protocol. Similarly, finding a new billet for one sailor will generally require much the same process as for another. Even the negotiation process between IDA and a sailor promises to be most frequently a relatively routine process. However, we expect IDA to occasionally receive messages outside of this expected group. Can IDA handle such a message intelligently by virtue of her "consciousness" mechanism alone? We do not think so. Additional mechanisms will be required.

One typical way for a nonroutine problem to arise is for some expectation to be unmet. I flip the switch, and the light does not turn on. In the IDA model, such an unexpected situation would likely be brought to "consciousness" by a particular type of attention codelet, an expectation codelet. In such a situation, behavior codelets responding in Step 6 may have no suitable behavior streams to instantiate (or one or more may be tried over several cycles and all fail). In

this case, a behavior stream is instantiated that implements a variant of a partial-order planner (Gerevin & Schuber, 1996). This planner behavior stream operates in a deliberative (virtual) mode.

The start-state for the plan described here is the current state as reported by the contents of consciousness. The goal-state is chosen so as to satisfy the failed expectation. The backward-chaining planner uses as its initial set of operators the behavior codelets that responded to the broadcast during the cycle in which the planner behavior stream was implemented. The first (backward from the goal-state) step in the plan is chosen and is then written to working memory as IDA's action during the current cycle. On subsequent cycles, additional steps are added to the plan until the start-state is reached. The completed plan becomes a behavior stream that is saved and that is likely instantiated on some forthcoming cycle for trial. This process may be repeated as necessary. Nonroutine problem solving in IDA is a type of procedural learning. The collection of behavior stream templates, together with the behavior codelets, constitutes IDA's long-term procedural memory.

Automization

In this subsection, we briefly describe a mechanism by means of which a software agent can learn to automatize a skill that is a sequence of actions so as to perform it without "conscious" intervention (Negatu, McCauley, & Franklin, in review). Typically, when an action plan is learned for the first time, consciousness plays a major role. As the action plan is executed repeatedly, experience accumulates, and parts of the action plan eventually are automatized.

Motivated by pandemonium theory (Jackson, 1987), whenever any two codelets are active together in the IDA computational model, the association between them becomes stronger (or weaker if things are not going well). This is Hebbian learning (Hebb, 1949). Suppose an attention codelet, say AC1, belongs to the winning coalition, bringing with it the information that will eventually trigger the start of a given task. When the content of the coalition is broadcast to all behavior codelets, suppose a behavior codelet, say BC1, that responded instantiates a behavior stream appropriate for the task. IDA's behavior net eventually picks a particular behavior, say B1, for execution, and its underlying behavior codelets become active. For simplicity's sake, let us assume that each behavior in the stream has only one behavior codelet, and that B1 has its codelet, BC1, active. If the action of BC1 attracts attention codelet AC2 and its coalition to "consciousness," then its content is broadcast. Similarly, suppose some behavior codelet, say BC2, under behavior B2 responds to this broadcast, and that B2 is chosen for execution, and BC2 becomes active.

Note that the codelets in the sequence BC1–AC2–BC2 are all overlappingly active together during a relatively short time period. Suppose our hypothetical task is executed repeatedly, producing the sequence BC1–AC2–BC2 repetitively. As automatization builds, the associations BC1–AC2, BC2–AC2, and BC1–BC2 increase. When the association BC1–BC2 is over threshold, the automatization mechanism allows BC1 to spread activation directly to BC2, causing it to become active without the intermediary of AC2. At the same time, the strong associations BC1–AC2 and BC2–AC2 diminish the attention codelet AC2's activation so that it has less probability to make it to "consciousness." Thus, the sequence BC1–AC2–BC2, which involves "consciousness," is transformed by IDA's automization mechanism into the subconscious action sequence BC1–BC2.

Transient Episodic Memory

Transient episodic memory is an unusual aspect of the IDA conceptual model. It is an episodic memory with a decay rate measured in hours. Though a "transient memory store" is often assumed (Panksepp, 1998, p. 129), the existence of such a memory has rarely been explicitly asserted (Donald, 2001; Conway, 2001; Baars & Franklin, 2003, Franklin et al., in review). In the IDA conceptual model, transient episodic memory is updated during Step 5 of each cognitive cycle with the contents of "consciousness." At this writing, we are in the process of expanding and testing our implementation of an experimental transient episodic memory using a ternary revision of sparse distributed memory allowing for an "I don't care" symbol (Ramamurthy, D'Mello, & Franklin, to appear).

Perceptual Memory

As described above, IDA's perceptual memory, including the slipnet and the perceptual codelets, is a fully implemented part of the running IDA computational model. The IDA conceptual model adds learning to this perceptual memory with updating during the broadcast (Step 5) of each cognitive cycle (Franklin et al., in review). New nodes and links are added to the slipnet as needed, while existing node and links have their base-level activations and weights updated, respectively.

Metacognition

IDA's predecessor, CMattie, had an impoverished metacognition module that prevented oscillations in her processing and tuned the parameters of her behavior net to make her more or less goal oriented or more or less opportunistic, etc. Metacognition in CMattie was implemented as a separate B-brain with its own decidedly different mechanism (Zhang, Dasgupta, & Franklin, 1998) that looked down on what the rest of CMattie was doing (Minsky, 1985) and interfered as needed.

Being dissatisfied with metacognition in CMattie, none was implemented in IDA. However, we have a back-burner project in mind to add a more powerful metacognitive capability to IDA using only her current architecture. This would be part of adding a reporting self to IDA, the subject of a different article. Metacognition would be implemented by a set of appropriate behavior codelets, behaviors, and behavior streams, together with suitable attention codelets.

Learning

The IDA model is also intended to learn in several different ways. In addition to learning via transient episodic and declarative memory as described above, IDA also learns via Hebbian temporal association, as discussed in the section on automization. A coalition that comes to "consciousness" substantially increases the associations between the codelets that form the coalition. The same is true, to a lesser extent, when they are simply active together. Recall that these associations provide the basis for coalition formation.

Step 5 of the cognitive cycle, the broadcast step, also describes the updating of IDA's perceptual memory, the slipnet, using the contents of "consciousness." Procedural learning in the IDA conceptual model also occurs during Step 5, with "conscious" contents providing reinforcement to actions of behaviors. These two forms of learning use a similar mechanism, a base-level activation, the first for nodes (and weights on links), and the second for primitive behavior codelets.

Yet another form of learning in the IDA conceptual model is chunking. The chunking manager gathers highly associated coalitions of codelets into a single supercodelet in the manner of concept demons from pandemonium theory (Jackson, 1987) or of chunking in SOAR (Laird et al., 1987).

Hypotheses

Each design decision as to architecture and mechanisms taken in constructing the IDA conceptual model translates directly into a hypothesis about human cognition for cognitive scientists and neuroscientists (Franklin, 1997). In this section, we highlight a few of these hypotheses (Franklin et al., in review):

1. **The cognitive cycle.** Much of human cognition functions by means of cognitive cycles, continual interactions between conscious content, various memory systems, and the action selection mechanism. The IDA model suggests that consciousness occurs as a sequence of discrete, coherent episodes separated by short periods of no conscious content (see also, VanRullen & Koch, 2003).
2. **Transient episodic memory.** Humans have a content-addressable, associative, transient episodic memory with a decay rate measured in hours (Conway, 2001). In our theory, a conscious event is stored in transient episodic memory by a conscious broadcast. A corollary to this hypothesis says that conscious contents can only be encoded (consolidated) in long-term declarative memory via transient episodic memory.
3. **Perceptual memory.** A perceptual memory, distinct from semantic memory but storing much the same contents, exists in humans and plays a central role in the assigning of interpretations to incoming stimuli. The conscious broadcast begins and updates the processes of learning to recognize, to categorize, and to form concepts, all employing perceptual memory.
4. **Procedural memory.** Procedural skills are shaped by reinforcement learning operating through consciousness over more than one cognitive cycle.
5. **Voluntary and automatic attention.** In the Global Workspace/IDA model, attention is the process of bringing contents to consciousness. Automatic attention occurs subconsciously and without effort during a single cognitive cycle. Attention may also occur voluntarily in a consciously goal-directed way, over multiple cycles.

Conclusion

Here I hope to have described an architecture capable of implementing many human cognitive, including affective, functions within the domain of an autono-

mous software agent. I would hesitate to claim that this architecture, as is, is fully functioning by human standards. Even the conceptual model lacks, for instance, the typical human senses of vision, olfaction, audition, etc. Its contact with the world is only through strings of characters. There is only the most rudimentary sensory fusion by the agents. They lack selves, self-awareness, and the ability to report internal events. There is much work left to be done.

Nonetheless, the IDA conceptual model, together with its architecture and mechanisms, does answer, for the agent, the question of what to do next. Thus, it constitutes a theory of mind. The IDA model seems to satisfy the requirements of an artificial mind as outlined in the introduction above. Something of this view may also be ascribed to Owen Holland, who wrote as follows (2003):

In many ways, Stan Franklin's work on "conscious software" offers a real challenge to functionalists. If consciousness is what consciousness does, then his systems may well exceed the requirements, in that they not only mimic successfully the outcomes of conscious processes in some humans (naval dispatchers [sic]) but they do it in the way that the conscious human brain appears to do it, since their functional components are explicitly modeled on the elements of Baars' global workspace theory.... (p. 3)

Acknowledgments

This research was supported in part by ONR grant N00014-98-1-0332 and was produced with essential contributions from the "Conscious" Software Research Group, including Bernard Baars, Art Graesser, Lee McCauley, Satish Ambati, Ashraf Anwar, Myles Bogner, Sidney D'Mello, Arpad Kelemen, Ravikumar Kondadadi, Irina Makkaveeva, Aregahegn Negatu, Hongjun Song, Allexei Stoliartchouk, Uma Ramamurthy, Matthew Ventura, and Zhaohua Zhang.

Endnotes

[1] My use of feminine pronouns when speaking of IDA simply seems more comfortable given that she seems independent, intelligent, and capable, and that the acronym is a common English feminine name. Nothing more should be read into it.

2 Not to be confused with the phrase "theory of mind" as used by ethologists.

3 The term "cognitive" is used here in a broad sense so as to include affect.

4 We make no claim that IDA is phenomenally conscious. On the other hand, we know of no convincing argument that she is not.

5 Though part of the IDA conceptual model, the material on nonroutine problem solving has not yet been published.

References

Allen, J. J. (1995). *Natural language understanding*. Redwood City, CA: Benjamin/Cummings.

Anwar, A., Dasgupta, D., & Franklin, S. (1999). Using genetic algorithms for sparse distributed memory initialization. *International Conference Genetic and Evolutionary Computation (GECCO)*. July 6–9.

Anwar, A., & Franklin, S. (2003). Sparse distributed memory for "conscious" software agents. *Cognitive Systems Research, 4*, 339–354.

Baars, B. J. (1988). *A cognitive theory of consciousness*. London; Oxford: Cambridge University Press.

Baars, B. J. (1997). *In the theater of consciousness*. Oxford: Oxford University Press.

Baars, B. J. (2002). The conscious access hypothesis: Origins and recent evidence. *Trends in Cognitive Science, 6*, 47–52.

Baars, B. J., & Franklin, S. (2003). How conscious experience and working memory interact. *Trends in Cognitive Science, 7*, 166–172.

Baddeley, A. D. (1993). Working memory and conscious awareness. In A. Collins, S. Gathercole, M. A. Conway, & P. Morris (Eds.), *Theories of memory*. Mahweh, NJ: Lawrence Erlbaum.

Barsalou, L. W. (1999). Perceptual symbol systems. *Behavioral and Brain Sciences, 22* 577–609.

Bogner, M. (1999). *Realizing "consciousness" in software agents*. PhD dissertation. University of Memphis, TN.

Bogner, M., Ramamurthy, U., & Franklin, S. (2000). "Consciousness" and conceptual learning in a socially situated agent. In K. Dautenhahn (Ed.), *Human cognition and social agent technology*. Amsterdam: John Benjamins.

Conway, M. A. (2001). Sensory-perceptual episodic memory and its context: Autobiographical memory. In A. Baddeley, M. Conway, & J. Aggleton (Eds.), *Episodic memory*. Oxford: Oxford University Press.

Damasio, A. R. (1999). *The feeling of what happens*. New York: Harcourt Brace.

Donald, M. (2001). *A mind so rare*. New York: Norton.

Edelman, G. M. (1987). *Neural Darwinism*. New York: Basic Books.

Ericsson, K. A., & Kintsch, W. (1995). Long-term working memory. *Psychological Review, 102*, 21–245.

Franklin, S. (1995). *Artificial minds*. Cambridge, MA: MIT Press.

Franklin, S. (1997). Autonomous agents as embodied AI. *Cybernetics and Systems, 28*, 499–520.

Franklin, S. (2001). Conscious software: A computational view of mind. In V. Loia & S. Sessa (Eds.), *Soft computing agents: New trends for designing autonomous systems*. Berlin: Springer (Physica-Verlag).

Franklin, S., Baars, B. J., Ramamurthy, U., & Ventura, M. (In review). The role of consciousness in memory.

Franklin, S., & Graesser, A. C. (1997). Is it an agent, or just a program?: A taxonomy for autonomous agents. In *Intelligent Agents III*. Berlin: Springer Verlag.

Franklin, S., & Graesser, A. (1999). A software agent model of consciousness. *Consciousness and Cognition, 8*, 285–305.

Franklin, S., & Graesser, A. (2001). Modeling cognition with software agents. In J. D. Moore & K. Stenning (Eds.), *CogSci2001*: *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, August 1–4. Mahwah, NJ: Lawrence Erlbaum Associates.

Franklin, S., Kelemen, A., & McCauley, L. (1998). IDA: A cognitive agent architecture. In *IEEE Conference on Systems, Man and Cybernetics*. Washington, DC: IEEE Press.

Franklin, S., & McCaulley, L. (2004). Feelings and emotions as motivators and learning facilitators. *Architectures for Modeling Emotions. AAAI Spring Symposia Technical Series* [Technical Report SS-04-02].

Gerevin, A., & Schuber, L. (1996). Accelerating partial-order planners: Some techniques for effective search control and pruning. *Journal of Artificial Intelligence Research, 5*, 95–137.

Hebb, D. O. (1949). *Organization of behavior*. New York: John Wiley.

Hofstadter, D. R., & Mitchell, M. (1994). The Copycat Project: A model of mental fluidity and analogy-making. In K. J. Holyoak & J. A. Barnden

(Eds.), *Advances in connectionist and neural computation theory, Vol. 2: Logical connections*. Norwood NJ: Ablex.

Holland, J. H. (1986). A mathematical framework for studying learning in classifier systems. *Physica, 22 D*, 307–317. (Also in J. D. Farmer, A. Lapedes, N. H. Packard, & B. Wendroff [Eds.], *Evolution, games and learning*. Amsterdam: Elsevier/North Holland.)

Holland, O. (Ed.). (2003). Special issue on machine consciousness. *Journal of Consciousness Studies, 10*(4–5).

Jackson, J. V. (1987). Idea for a mind. *Siggart Newsletter, 181*, 23–26.

Johnson, V. S. (1999). *Why we feel: The science of human emotions*. Reading, MA: Perseus Books.

Kanerva, P. (1988). *Sparse distributed memory*. Cambridge, MA: The MIT Press.

Kelemen, A., Liang, Y., Kozma, R., & Franklin, S. (2002). Optimizing intelligent agent's constraint satisfaction with neural networks. In A. Abraham & B. Nath (Eds.), *Innovations in intelligent systems*. Heidelberg: Springer-Verlag.

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

Maes, P. (1989). How to do the right thing. *Connection Science, 1*, 291–323.

Maturana, H. R. (1975). The organization of the living: A theory of the living organization. *International Journal of Man–Machine Studies, 7*, 313–332.

Maturana, R. H., & Varela, F. J. (1980). *Autopoiesis and cognition: The realization of the living, Dordrecht*. Netherlands: Reidel.

McCauley, T. L., & Franklin, S. (1998). An architecture for emotion. In *AAAI Fall Symposium Emotional and Intelligent: The Tangled Knot of Cognition*. Menlo Park, CA: AAAI Press.

Minsky, M. (1985). *The society of mind*. New York: Simon & Schuster.

Negatu, A., & Franklin, S. (2002). An action selection mechanism for "conscious" software agents. *Cognitive Science Quarterly, 2*, 363–386.

Negatu, A., McCauley, T. L., & Franklin, S. (In review). Automatization for software agents.

Ornstein, R. (1986). *Multimind*. Boston, MA: Houghton Mifflin.

Panksepp, J. (1998). *Affective neuroscience*. Oxford: Oxford University Press.

Picard, R. (1997). *Affective computing*. Cambridge, MA: The MIT Press.

Prem, E. (Ed.). (1997). Epistemological aspects of embodied artificial intelligence. *Cybernetics and Systems, 28*(5&6).

Ramamurthy, U., D'Mello, S., & Franklin, S. (to appear). Modified sparse distributed memory as transient episodic memory for cognitive software agents. Proceedings of the International Conference on Systems, Man

Rolls, E. T. (1999). *The brain and emotion*. Oxford: Oxford University Press.

Shastri, L. (2001). A computational model of episodic memory formation in the hippocampal system. *Neurocomputing, 38–40,* 889–897.

Shastri, L. (2002). Episodic memory and cortico-hippocampal interactions. *Trends in Cognitive Sciences, 6,* 162–168.

Sloman, A. (1987). Motives mechanisms emotions. *Cognition and Emotion, 1*, 217–234.

Sloman, A. (1998). The "semantics" of evolution: Trajectories and trade-offs in design space and niche space. In H. Coelho (Ed.), *Progress in artificial intelligence*. Berlin: Springer.

Sloman, A., & Chrisley, R. (2003). Virtual machines and consciousness. *Journal of Consciousness Studies, 10,* 133–172.

Song, H., & Franklin, S. (2000). A behavior instantiation agent architecture. *Connection Science, 12,* 21–44.

Taylor, J. G. (1999). *The race for consciousness*. Cambridge, MA: MIT Press.

Valenzuela-Rendon, M. (1991). The fuzzy classifier system: A classifier system for continuously varying variables. In *Proceedings of the Fourth International Conference on Genetic Algorithms*. San Mateo, CA: Morgan Kaufmann.

VanRullen, R., & Koch, C. (2003). Is perception discrete or continuous? *Trends in Cognitive Science, 7,* 207–213.

Zhang, Z., Dasgupta, D., & Franklin, S. (1998). Metacognition in software agents using classifier systems. In *Proceedings of the 15th National Conference on Artificial Intelligence*. Madison, WI: MIT Press.

Zhang, Z., Franklin, S., Olde, B., Wan, Y., & Graesser, A. (1998). Natural language sensing for autonomous agents. In *Proceedings of IEEE International Joint Symposia on Intellgence Systems '98*.