Building Life-Like "Conscious" Software Agents

Stan Franklin[1,2,3]
Institute for Intelligent Systems and
Department of Mathematical Sciences
The University of Memphis

Abstract

Here we'll briefly describe action selection and language generation mechanisms in two "life-like" software agents, CMattie and IDA, and discuss issues that bear on the topics of architectures for behavior control, interdependencies between emotions and goal-based behaviors, and coordination of scripted and improvised behaviors. These agents are life-like in the sense of interacting with humans via email in natural language. They are "conscious" only in the sense of implementing a psychological theory of consciousness (Baars 1988, 1997). At this writing we are exploring the transition from scripted language production to more improvised speech generation. We are also investigating deliberative behavior selection mechanisms whereby alternative scenarios are produced and evaluated, and one of them chosen and acted upon.

Agents: Autonomous, Cognitive and “Conscious”

The rise of the web has spawned a use of the word “agent” in other than its more common meaning as in “travel agent” or “insurance agent” or “real-estate agent.” In this new context an agent is a piece of software, a computer program, that in some sense acts on it’s own, typically in the service of its user, a person. Here I’ve chosen to define the technical term “autonomous agent,” at least partly to avoid the seemingly endless debates over exactly what is an agent. An *autonomous agent*, in this paper, 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. It also acts in such a way as to possibly influence what it senses at a later time (Franklin and Graesser 1997). These autonomous agents include biological agents such as humans and most, perhaps all, animals. They also include some mobile robots, like the robots that deliver pharmaceuticals in hospitals. Some computational agents such as many artificial life agents who “live” in artificial environments designed for them within computer systems (Ackley & Littman, 1992) are also autonomous agents. Finally, so also are the objects of our attention in this paper, software agents, at least some of them. These autonomous software agents include task-specific agents like spiders that search for links on the web, such entertainment agents as Julia (Mauldin,1994), and, much to the regret of many of us, computer viruses. The class also includes the “conscious” software agents we’ll describe here.

[1] [1]Supported in part by ONR grant N00014-98-1-0332

[2] With essential contributions from the Conscious Software Research Group including Art Graesser, Satish Ambati, Ashraf Anwar, Myles Bogner, Arpad Kelemen, Irina Makkaveeva, Lee McCauley, Aregahegn Negatu, Uma Ramamurthy, Zhaohua Zhang

[3] stan.franklin@memphis.edu, www.msci.memphis.edu/~franklin

But the notion of autonomous agent turns out to be too broad for our needs; it even includes a thermostat. So, let’s restrict it to make it more suitable. Suppose we equip our autonomous agent, with cognitive features, interpreting “cognitive” broadly so as to include emotions and such. Choose these features from among multiple senses, perception, short and long term memory, attention, planning, reasoning, problem solving, learning, emotions, moods, attitudes, multiple drives, etc., and call the resulting agent a *cognitive agent* (Franklin 1997). Cognitive agents would include humans, some of our primate relatives, and perhaps elephants, some cetaceans, and perhaps even Alex, an African grey parrot (Pepperberg 1994). Examples are rare among non-biological agents, perhaps eventually including Rod Brooks’ humanoid robot Cog (Brooks 1997)]], some agents modeled on Sloman’s architectural scheme (Sloman 1996), our own software agents VMattie, CMattie and IDA to be described below, and a handful of others.

Though quite ill defined, cognitive agents can play a useful, even a synergistic role in the study of human cognition, including consciousness. Here’s how it can work. A theory of cognition constrains the design of a cognitive agent that implements that theory. While a theory is typically abstract, functional and only broadly sketches an architecture, an implemented design must provide a fully articulated architecture, and the mechanisms upon which it rests. This architecture and these mechanisms serve to flesh out the theory, making it more concrete. Also, every design decision taken during an implementation constitutes a hypothesis about how human minds work. The hypothesis says that humans do it the way the agent was designed to do it, whatever “it” was. These hypotheses will suggest experiments with humans by means of which they can be tested. Conversely, the results of such experiments will suggest corresponding modifications of the architecture and mechanisms of the cognitive agent implementing the theory. The concepts and methodologies of cognitive science and of computer science will work synergistically to enhance our understanding of mechanisms of mind. I have written elsewhere in much more depth about this research strategy (Franklin 1997). A paper currently being prepared will detail hypotheses suggested by the CMattie model to be described below (Bogner, Franklin, Graesser and Baars, in preparation).

Here we’ll be concerned with *“conscious” software agents.* These agents are cognitive agents in that they consist of modules for perception, action selection (including constraint satisfaction and deliberation), several working memories, associative memory, episodic memory, emotion, several kinds of learning, and metacognition. They model much of human cognition. But, in addition, these agents include a module that models human consciousness according to global workspace theory (Baars 1988, 1997). Our aim in this work is twofold. We want to produce a useful conceptual and computational model of human cognition and consciousness. At the same time we aim to produce more flexible, more human-like AI systems.

In global workspace theory Baars postulates that human cognition is implemented by a multitude of relatively small, special purpose processes, almost always unconscious. Communication between them is rare and over a narrow bandwidth. Coalitions of such processes find their way into a global workspace (and into consciousness). This limited capacity workspace serves to broadcast the message of the coalition to all the unconscious processors, in order to recruit other processors to join in handling the current novel situation, or in solving the current problem. Thus consciousness, according to this theory, allows us to deal with novelty or problematic situations that can’t be dealt with efficiently, or at all, by habituated unconscious processes. This is the key insight of global workspace theory. There’s much, much more to it than is stated here including volitional action via William James’ ideamotor theory.

CMattie

CMattie is a “conscious” clerical software agent (Bogner, Ramamurthy and Franklin 2000; McCauley and Stan Franklin 1998; Ramamurthy, Bogner, and Franklin 1998, Zhang, Franklin and Dasgupta 1998). She composes and emails out weekly seminar announcements, having communicated by email 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 email messages. CMattie's cognitive modules include perception, learning, action selection, associative memory, "consciousness," emotion and metacognition.

Perception in CMattie consists mostly of understanding incoming email messages in natural language. In sufficiently narrow domains, natural language understanding may be achieved via an analysis of surface features without the use of a traditional symbolic parser. Allen describes this approach as complex, template-based matching, natural language processing (1995). CMattie’s limited domain requires her to deal with only a dozen or so distinct message types, each with relatively predictable content. This allows for surface level natural language processing. CMattie's language understanding module has been implemented as a Copycat-like architecture (Hofstadter et al 1994) though her understanding takes place differently. The mechanism includes a slipnet storing domain knowledge, and a pool of codelets (processors) specialized for specific recognition and identification tasks, along with templates for building and verifying understanding. Together they constitute an integrated sensing system for the autonomous agent CMattie. With it she's able to recognize, categorize and understand incoming email messages.

We include in CMattie mechanisms for emotions (McCauley & Franklin 1998). CMattie may “experience” such emotions as guilt at not getting an announcement out on time, frustration at not understanding a message, and anxiety at not knowing the speaker and title of an impending seminar. Action selection will be influenced by emotions via their effect on drives, modeling recent work on human action selection (Damasio 1994).

CMattie can “experience” four basic emotions, anger, fear, happiness and sadness. These emotions can vary in intensity as indicated by their activation levels. For example, anger can vary from mild annoyance to rage as its activation rises. A four vector containing the current activations of these four basic emotions represents CMattie’s current emotional state. Like humans, there’s always some emotional state however slight. Also like humans, her current emotional state is often some complex combination of basic emotions or results from some particular changes in them. The effect of emotions on codelets, drives, etc. varies with their intensity. Fear brought on by an imminent shutdown message might be expected to strengthen

CMattie’s self-preservation drive resulting in additional activation going from it into the behavior net.

CMattie’s emotional codelets serve to change her emotional state. When its preconditions are satisfied, an emotional codelet will enhance or diminish one of the four basic emotions. An emotion can build till saturation occurs. Repeated emotional stimuli result in habituation. Emotion codelets can also combine to implement more complex secondary emotions that act by affecting more than one basic emotion at once. Emotion codelets also serve to enhance or diminish the activation of other codelets, and to increase or decrease the strength of drives, thereby influencing CMattie’s choice of behaviors.

CMattie depends on a greatly enhanced version of a behavior net (Maes 1989) for high-level action selection in the service of their several parallel, built-in drives. These drives vary in urgency as time and the environment change. Behaviors are typically mid-level actions, many depending on several codelets for their execution. An expanded account of the operation of this action selection mechanism appears in the next section.

CMattie employs sparse distributed memory (SDM) as its major associative memory (Kanerva 1988). SDM is a content addressable memory that, in many ways, is an ideal computational mechanism for use as a long-term associative memory. Being 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 bits, zeros and ones. 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 a possible 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 groups of 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 average out of them. Reading is not always successful. Depending on the cue and the previously written information, 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 memory. A human often knows what he does or doesn't know. If asked for a telephone number I've once known, I may search for it. When asked for one I've never known, an immediate "I don't know" response ensues. SDM makes such decisions based on the speed of 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 address, and so on to convergence, that is, until subsequent contents look alike. If it doesn’t quickly converge, an “I don't know” is the response. The "on the tip of my tongue phenomenon" corresponds to being just at the threshold of convergence. 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’s the “distributed” part of sparse distributed memory. A well-rehearsed item can be retrieved with smaller cues. Another similarity is forgetting, which would tend to increase over time as a result of other similar writes to memory.

How does the agent use this associative memory? As one example, let’s suppose an email message for CMattie arrives, is transferred into the perceptual workspace (working memory), and is descended upon by perceptual codelets looking for words or phrases they recognize. When such are found, nodes in the slipnet (a semantic net type mechanism with activation passing that acts as a perceptual and conceptual knowledge structure) are activated, a message type is selected, and the appropriate template filled. The information thus created from the incoming message is then written to the perception registers in the focus, making it available to the rest of the system.

The contents of the focus are then used as an address to query associative memory. The results of this query, that is, whatever CMattie associates with this incoming information, are written into their own registers in the focus. This may include some emotion and some previous action. Attention codelets then attempt to take this information to “consciousness.” They bring along any discrepancies they may find, such as missing information, conflicting seminar times, etc. Information about the current emotion and the currently executing behavior are written to the focus by appropriate codelets. The current percept, consisting of the incoming information as modified by associations and the current emotion and behavior are then written to associative memory. Those percepts carrying strong emotions are written repeatedly yielding stronger associations. IDA handles perception in much the same way.

Metacognition should include knowledge of one’s own cognitive processes, and the ability to actively monitor and consciously regulate them. This would require self-monitoring, self-evaluation, and self-regulation. Following Minsky, we’ll think of CMattie’s “brain” as consisting of two parts, the A-brain and the B-brain (1985) The A-brain consists of all the other modules of the agent’s architecture. It performs all of her cognitive activities except metacognition. Its environment is the outside world, a dynamic, but limited, real world environment. The B-brain, sitting on top of the A-brain, monitors and regulates it. The B-brain’s environment is the A-brain, or more specifically, the A-brain’s activities.

The metacognitive module in this agent looks like an autonomous agent in its own right. It senses the A-brain’s activity and acts upon it over time in pursuit of its own agenda. It’s also structurally coupled to its quite restricted environment deriving its agenda from built in metacognitive drives. One such is to interrupt oscillatory behavior. Another such might be to keep the agent more on task, that is, to make it more likely that a behavior stream would carry out to completion. Yet another would push toward efficient allocation of resources. The mechanism is based on Holland's classifier systems (1986).

In the CMattie architecture (see Figure 1 below) the processors postulated by global workspace theory are implemented by codelets, small pieces of code. The use of these codelets and their associations were inspired by Jackson's pandemonium theory (1987). Codelets are specialized for some simple task and often play the role of demon waiting for 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 (earlier misnamed “conscious” codelets, or “consciousness” codelets as in Figure 1 below) who recognize novel or problematic situations (Bogner, 1998; Bogner, Ramamurthy, and Franklin, 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 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. The attention codelet increases its activation in order that the coalition might compete for “consciousness” if one is formed.

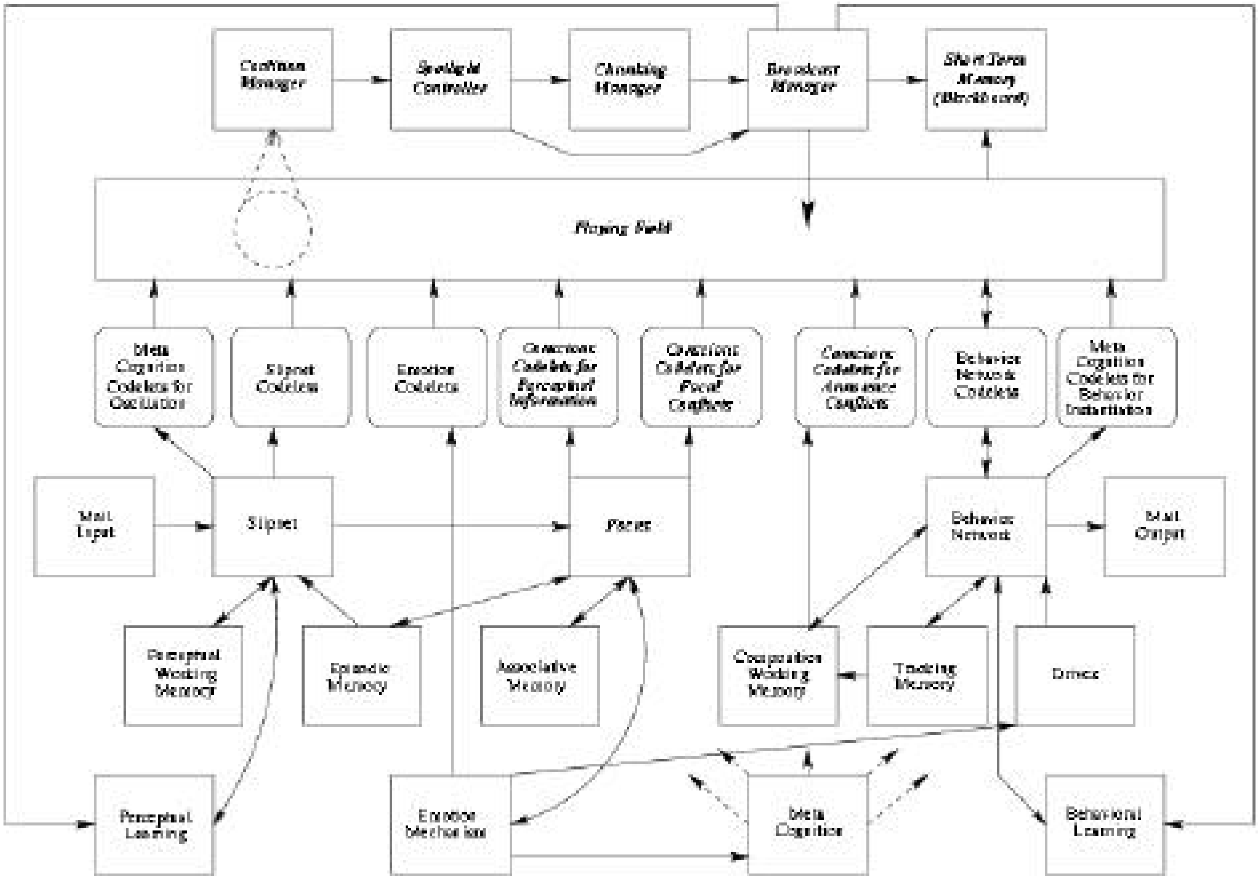
Coalition Manager
Spotlight Controller
Chunking Manager
Broadcast Manager
Short Term Memory (Blackboard)
Playing Field
Meta Cognition Codelets for Oscillation
Slipnet Codelets
Emotion Codelets
Conscious Codelets for Perceptual Information
Conscious Codelets for Focal Conflicts
Conscious Codelets for Awareness Conflicts
Behavior Network Codelets
Meta Cognition Codelets for Behavior Instantiation
Mail Input
Slipnet
Focus
Behavior Network
Mail Output
Perceptual Working Memory
Episodic Memory
Associative Memory
Composition Working Memory
Tracking Memory
Drives
Perceptual Learning
Emotion Mechanism
Meta Cognition
Behavioral Learning

Figure 1. The CMattie Architecture (including codelets)

CMattie's 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. Since association can both increase and diminish, the forming and tracking of coalitions is a dynamic process. Coalitions appear and disappear. Codelets may leave one coalition, and may join another. At any given time, one of these coalitions finds it way to “consciousness.” This is effected by the spotlight controller who chooses 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.

There’s more to CMattie’s architecture and mechanisms than can be adequately dealt with in a single journal article. A look at Figure 1 reveals several modules that haven’t been mentioned, for example, chunking manager, episodic memory, perceptual learning and behavioral learning. The chunking manager gathers highly associated coalitions of codelets in to a single “super” codelet in the manner of concept demons from pandemonium theory (Jackson 1987), or of chunking in SOAR (Laird, Newell and Rosenbloom 1987). CMattie’s episodic memory is cased based in order to be useful to the perceptual and behavior modules that will learn new concepts (Ramamurthy, Bogner and Franklin 1998), and new behaviors (Negatu and Franklin 1999) from interactions with seminar organizers. For example, CMattie might learn how a thesis defense differs from a seminar, and the behaviors appropriate to one.

IDA

IDA (Intelligent Distribution Agent) is to be a conscious software agent developed for the US Navy (Franklin, Kelemen, and McCauley 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 300 people, called detailers, full time to effect these new assignments. IDA's task is to facilitate this process, by playing the role of detailer. Designing IDA presents both communication problems, and action selection problems involving constraint satisfaction. She must communicate with sailors via email and in natural language, understanding the content and producing life-like responses. Sometimes she 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, by having the required number of sonar technicians on a destroyer with the required types of training. In doing so she must adhere to some ninety policies. She must hold down moving costs. And, she must cater to the needs and desires of the sailor as well as is possible. Finally, she must write the orders and start them on the way to the sailor.

IDA's architecture and mechanisms are largely modeled after those of CMattie described above, though more complex (see Figure 2 below). IDA’s perception mechanism will handle incoming email messages from sailors in much the same way as was described above for CMattie, though she’ll require a much more extensive slipnet and a larger corps of supporting codelets. IDA must also understand data retrieved from a sailor’s personnel record, from a listing of currently available jobs, and from other databases. Though this kind of perception is much easier than understanding natural language, it still requires additional perceptual mechanisms for IDA.

IDA’s emotion mechanism has also been considerably expanded. This includes the merging of emotion codelets with attention codelets as well as the introduction of a network of passing activation that allows emotions to affect essentially all action selection (McCauley, Franklin, and Bogner 1999). Much of its role is described below. IDA's action selection mechanism, again modeled after that of CMattie but much expanded, is also described below.

Associative memory presents a particular problem for IDA that didn't occur for CMattie. Kenerva's sparse distributed memory, as described above, requires large cues for successful retrieval. Some seventy percent of the focus must be filled in order guarantee convergence of a query. But in IDA's context associations must be made say to just a sailor's name and social security number, or some other equally small cue. We've developed a version of associative memory for IDA that is capable of successful retrieval from almost arbitrarily small cues. A paper describing this work in being prepared.

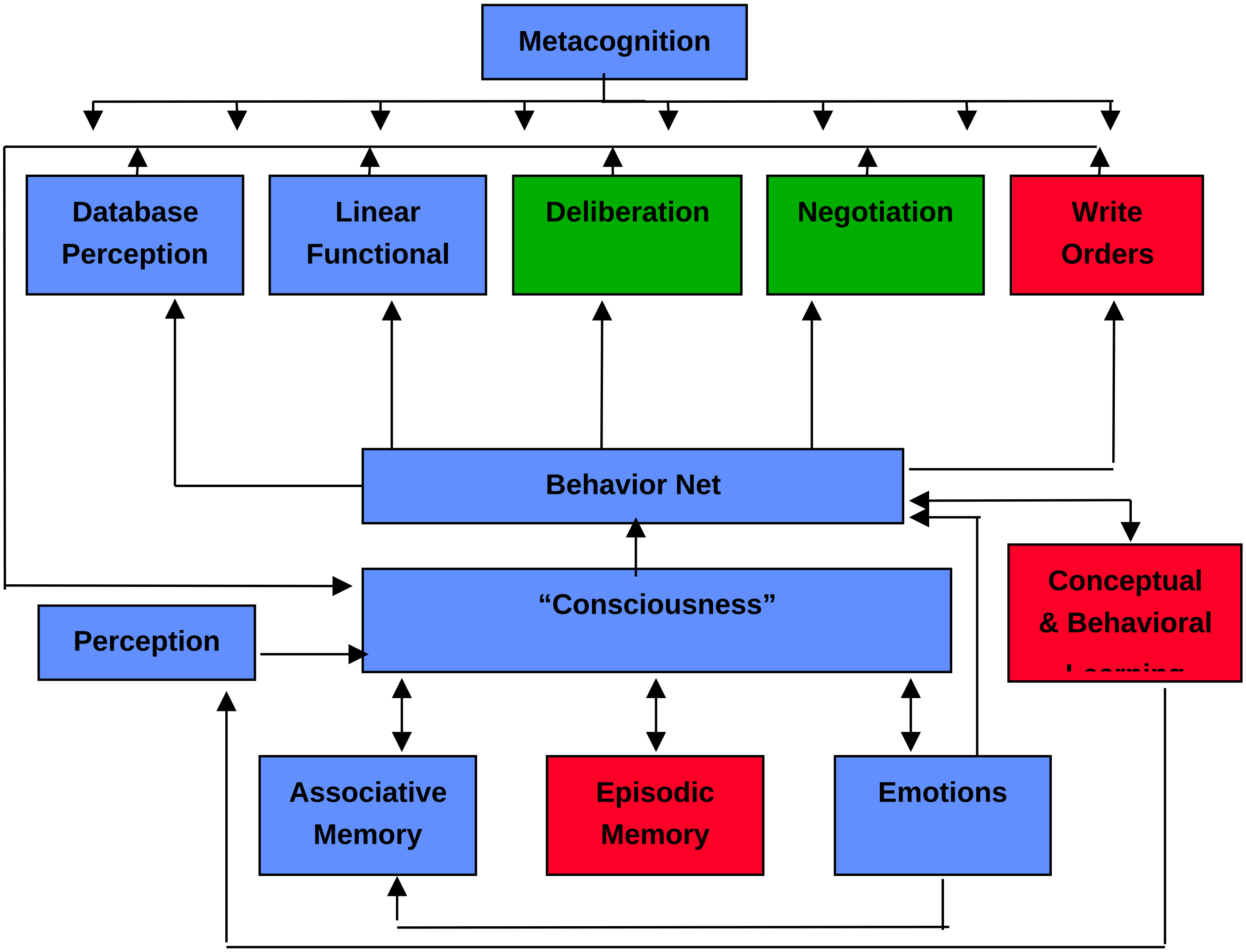

Figure 2. The IDA Architecture (not including codelets)

IDA will need deliberative reasoning in the service of action selection, where CMattie was able to do without. This process will be described in some detail in the next section. IDA will also require natural language generation in order to negotiate with sailors about their next job. This process is described in the section called Scripted and Improvised Behaviors. Her emotions will be involved in both of these activities as also described below. Metacognition for IDA is still in the planning stage, but we hope to learn from metacognition in CMattie.

Architectures for Behavior Control

Both CMattie and IDA depend on a behavior net (Maes 1990) for high-level action selection in the service of built-in drives, the primary motivators. Each has several distinct drives operating in parallel. These drives vary in urgency as time passes and the environment changes. Behaviors are typically mid-level actions, many depending on several codelets for their execution. From the point of view of global workspace theory, a behavior is a goal context. Each such context is a collection of processes, or in our case, a collection of codelets. A behavior can be almost identified with its codelets. A behavior net is composed of behaviors and their various links, along with goals and drives. 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, 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 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 activation stored in the behaviors themselves, from the environment, from drives, 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's 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 codelets responding to a “conscious” broadcast. Finally, activation spreads from behavior to behavior along links. Along successor links, one behavior strengthens those behaviors whose preconditions it can help fulfill by sending them activation. Along predecessor links, one behavior strengthens any other behavior whose add list 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. Call a behavior *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.

Action selection via behavior net suffices for CMattie due to her relatively constrained domain. IDA's domain is much more complex, and requires deliberation in the sense of creating possible scenarios, partial plans of actions, and choosing between them. For example, suppose IDA is considering a sailor and several possible jobs, all seemingly suitable. She must construct a scenario for each 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, after travel time, arrives at the new billet with in a given time frame. Such scenarios are valued on how well they fit the temporal constraints and on moving and training costs.

Scenarios are composed of scenes. IDA's scenes are organized around events. Each scene may require objects, actors, concepts, relations, and schema represented by frames. They are constructed in a computational workspace corresponding to working memory in humans. We use Barsalou's perceptual symbol systems as a guide (1999). The perceptual/conceptual knowledge base of this agent 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 and Mitchell, 1994). Nodes of the slipnet constitute the agent's perceptual symbols. Pieces of the slipnet containing nodes and links, together with codelets whose task it is to copy the piece to working memory constitute Barsalou's perceptual symbol simulators. These perceptual symbols are used to construct scenes in working memory. The scenes are strung together to form scenarios. The work is done by deliberation codelets. Evaluation of scenarios is also done by codelets.

Deliberation, as in humans, is mediated by the "consciousness" mechanism. Imagine IDA in the context of a behavior stream whose goal is to find a billet for a particular sailor. (Figure 2 above should help in following this description.) Perhaps a behavior executes to read appropriate items from the sailor's personnel database record. Then, possibly, comes a behavior to locate the currently available billets. Next might be a behavior that runs each billet and that sailor through IDA's constraint satisfaction module, producing a small number of candidate billets. Finally a deliberation behavior may be executed that sends deliberation codelets to working memory together with codelets carrying billet information. A particular billet's codelets wins its way into "consciousness." Scenario building codelets respond to the broadcast and begin creating scenes. This scenario building process, again as in humans, has both it's "unconscious" and its "conscious" activities. Eventually scenarios are created and evaluated for each candidate billet and one of them is chosen. Thus we have behavior control via deliberation.

The details of the interaction between the behavior net, the "consciousness" mechanism, and the workspace (working memory) are rather complex. Let's trace through the beginning of the creation of a hypothetical scenario. Suppose in the context of a particular sailor, information concerning a given candidate job is currently in the workspace. This might include such items as location, time interval to report, job skills required, etc. It would also include a priority assigned by the Navy, and by this point in the action, a numeric measure of fitness, that is, of how well this combination of sailor/job satisfies the various constraints involved, mostly by Navy policy.

Here’s how the creation of a scenario might begin. An attention codelet, noting that all candidate jobs in the workspace have been measured for fitness, chooses our given job for a scenario on the basis of its fitness. This codelet then greatly increases its activation and gathers information bearing codelets carrying information about this job, and the collection becomes active (joins the playing field). Due to their strong associations, this collection of codelets becomes a coalition. Due to their high activation, this coalition eventually occupies the spotlight of “consciousness” and the information carried is broadcast to all codelets. Here the role of “consciousness” as a recruiter of relevant resources is illustrated. Among the codelets receiving the broadcast is a collection of relevant codelets each of whom will bind its variables to some of the information being broadcast. Together they cause a goal structure, consisting of the goal to have created a scenario and the necessary behavior streams, to be instantiated as part of the behavior net. When the first behavior in this structure is executed, its associated codelets start a new scenario in the workspace with the first event being an arrive-no-later-than-date derived within the required interval. Note that the interaction just described between “consciousness” and the behavior net constitutes a major departure from Maes’ original concept, and makes Tyrrell’s criticisms moot (1994).

This same process, beginning with an attention codelet noticing that a scenario has begun with an arrive-no-later-than-date, gathers information codelets having to do with the sailor’s detachment from his or her current command. These go through the “consciousness” process, including the behavior net, with the result being a new detachment-date added to the emerging scenario. Scenes are all added in this way. Adjustments in dates are made by the same process, as is the decision to keep this scenario or give up on it.

Emotions and Goal-based Behaviors

In both CMattie and IDA we include mechanisms for emotions (McCauley and Franklin 1998). CMattie, for example may “experience” such emotions as guilt at not getting an announcement out on time, frustration at not understanding a message, and anxiety at not knowing the speaker and title of an impending seminar. Action selection will be influenced by emotions via their effect on drives, modeling recent work on human action selection (Damasio 1994). For example, anxiety over not having information about a scheduled seminar may lead to increased activation output from the drive to send reminders.

CMattie can “experience” four basic emotions, anger, fear, happiness and sadness. These emotions can vary in intensity as indicated by their activation levels. For example, anger can vary from mild annoyance to rage as its activation rises. A four vector containing the current activations of these four basic emotions represents CMattie’s current emotional state. Like humans, there’s always some emotional state however slight. Also like humans, her current emotional state is often some complex combination of basic emotions. The effect of emotions on codelets, drives, etc. varies with their intensity. Fear brought on by an imminent shutdown message might be expected to strengthen CMattie’s self-preservation drive resulting in additional activation going from it into the behavior net.

CMattie’s emotional codelets serve to change her emotional state. When its preconditions are satisfied, an emotional codelet will enhance or diminish one of the four basic emotions. An emotion can build till saturation occurs. Repeated emotional stimuli result in habituation. That is, a repeatedly stimulated emotion codelet will be less intense at each repetition (see McCauley and Franklin 1998 for details). As a result, repeated emotional stimuli have less effect on the action selection of these agents with each repetition. Emotion codelets can also combine to implement more complex secondary emotions that act by affecting more than one basic emotion at once. Emotion codelets also serve to enhance or diminish the activation of other codelets (McCauley, Franklin and Bogner 1999). For example, an attention codelet with additional emotional activation will be more likely to form a coalition that competes well for the spotlight of “consciousness.” As mentioned above, emotion codelets also act to increase or decrease the strength of drives, thereby influencing CMattie’s choice of behaviors in another way. Otherwise, emotion codelets affect behaviors only through their associated codelets. They play no direct role in the behavior net.

We expect the inclusion of emotions in these agents to result in more flexible, more appropriate, more human-like decision making as it does in humans (Damasio 1994, Rolls 1999). Whether or not this expectation will be fulfilled must await experimentation with IDA. The domain of CMattie is probably too simple for a difference between decision making with and without emotions to be apparent. Nonetheless, we do intend to perform such experiments on CMattie.

Scripted and Improvised Behaviors

CMattie’s behaviors consist almost entirely of sending email messages to seminar organizers, attendees and the system administrator. In every case these messages are composed by codelets filling out templates, that is scripts with blanks allowing them to be specialized to suit the current situation. This is even true when CMattie is in the process of learning new concepts and/or behavior via interactions with organizers (Ramamurthy, Bogner and Franklin 1998). If, for example, she writes, “If a colloquium is different from a seminar, how is it different?” she has filled in a template adding “seminar” and “colloquium.” Of course, she has to be able to understand the reply.

IDA, on the other hand, must be able to respond to quite varied email messages from sailors concerning their next assignment. Many ideas from these messages will be about standard requests and can be answered with a script. It may be that all such can be so answered. We’re currently cataloging such ideas, and will soon begin producing appropriate scripts. But, what of the rare idea that isn’t found in our catalog? If it’s something about which IDA has no knowledge she, like a human, will not be capable of any intelligent response except, possibly, to try to learn. If IDA knows something of the subject of the idea, this knowledge will likely be found in her slipnet. An improvised response would then be created by a language generation module working from the same principles as the deliberation module described above. Improvised linguistic structures created in the workspace might be combined with scripts to produce an appropriate response. All this would be controlled by a stream of behaviors in IDA's behavior net, and would be mediated by her "consciousness" mechanism.

In particular, IDA must compose a message offering the sailor a choice of one, two or sometimes three jobs. In this situation the jobs have already been chosen and the needed information concerning them are present in the workspace. The same back and forth to "consciousness" process is used as described at the end of the section entitled Archictectures for Behavior Control above. An attention codelet, noting that the decisions on which jobs to offer have been made, brings to "consciousness" the need for a message offering the jobs. Information codelets, responding to the content of the "conscious" broadcast, instantiate into the behavior net a goal structure to write an offering message. The codelets associated with the first behavior to be executed from that structure will write a salutation for that message appropriate to the sailor's occupation and pay grade. Another attention codelet will bring to "consciousness" the number of jobs to be offered and the need for an introductory paragraph. The responding information codelets will send activation to the appropriate behavior resulting in its codelets writing that paragraph from a built-in script into the workspace. The same process will next bring into the workspace a paragraph describing the first job to be offered. Note that different jobs will require very different such paragraphs. The appropriate information codelets for the particular kind of job in question will be recruited by the "conscious" broadcast, eventually resulting in the appropriate paragraph being written to the workspace. Some modification may be made by codelets. The same process will continue until the message is composed. This is a much more complex process than that used by CMattie. It's designed this way to accommodate the more individual nature of each message. It's a scripted language generation with modifications.

Concluding Remarks

The "conscious" software agent architecture offers a promising vehicle for producing autonomous software agents that are life-like in their interactions with humans via email. They will be life-like in that they understand the human correspondent's natural language and are able to respond, also in natural language. The architecture and mechanisms underlying these abilities are, themselves, life-like in that they are modeled after human cognition and consciousness. Such "conscious" software agents show promise of being able to duplicate the tasks of many different human information agents.

We've also proposed these agents as embodying conceptual and computational models of human consciousness and cognition. As such they promise to yield testable hypotheses for the cognitive scientists and for the cognitive neuroscientists. As mentioned above each design decision made during the development of the functioning of the agents translates into the hypothesis that humans function in the same way. As mentioned above, a paper proposing a number of such hypotheses is being prepared. Here's one example from that paper. Cognitive scientists seem to be agreed that enough conscious attention is highly correlated with learning (Baars 1997, p. 161). He goes on to say that "[t]here is a long and still unresolved controversy whether learning can occur without consciousness..." Our model posits a "yes, it can happen" answer to the question, and suggests a mechanism for it happening. Suppose an attention codelet, intent on bringing certain information to “consciousness,” becomes active, gathers the necessary information codelets, and competes for the spotlight of “consciousness.” The information carried by the coalition and intended for broadcast (to become “conscious”) is written to the focus when all this happens. If this coalition loses the competition for the spotlight, some other facts are broadcast and, shortly thereafter, the contents of the focus are written to associative memory. Those contents can include the information that never made it to “consciousness.” This hypothesis, while no doubt testable, is well known and “notoriously difficult.” We hope that many of our hypotheses will prove to be less well known and more tractable.

A quick look at the diagrams shown above together with skimming this paper shows that these agents are both quite complex, though IDA will be an order of magnitude more complex than CMattie. Intelligence never comes cheaply. Wilson’s Animat is a prime example (1991). There we have a very simple artificial animal in an absurdly simple environment (rocks, trees and food) having an awful time learning to find food. Our “conscious” software agents require a tremendous amount of knowledge engineering to gather the needed expertise. All this is on top of a complex architecture and underlying mechanisms. CMattie runs to about a quarter of a million lines of Java code. IDA may well require ten times as much. The many independent threads (each codelet requires its own thread) demand high-powered processors. “Conscious” software agents will never be cheap and easy, though the development tools on our to do list should help.

The mechanisms employed in our “conscious” software agents partake generously of the “new AI” with their multi-agents (codelets), their passing of activation everywhere, their behavior nets, slipnets, and classifiers. The design is based directly on my action selection paradigm of mind (Franklin 1995, 1997), which is closely related to the enactive paradigm of Varela, Thompson and Rosch. Both are squarely within the situated cognition camp. Our goals with these agents include both cognitive modeling and the creation of useful, more human-like, information agents. The cognitive modeling side led to our particular choices of mechanisms. I suspect CMattie could have as well been design along classical AI lines. I doubt that those symbolic techniques would have sufficed for IDA, though I have no evidence to back this up. Experimentation with CMattie is just beginning. Evaluation of her performance promises to be straightforward. Comparison with other agents and their mechanisms will, no doubt, prove less so. IDA is entering the implementation phase. Evaluating her performance is expected to be singularly difficult since she’s to take the place of a human detailer and the Navy has no way of effectively evaluating their performance.

References

Ackley, David, and Michael Liittman (1992). Interactions between Learning and Evolution. In Artificial Life II. (Langton, et al, ed.) Redwood City, CA: Addison-Wesley, 487-509.

Allen, J. J. 1995. *Natural Language Understanding*. Benjamin; Cummings.

Baars, B. J. (1988). A Cognitive Theory of Consciousness. Cambridge: Cambridge University Press.

Baars, B. J. (1997). In the Theater of Consciousness. Oxford: Oxford University Press.

Barsalou, L.W. (1999), Perceptual Symbol Systems, Behavioral and Brain Sciences. 22, 577-609

Bogner, M. (1998) Creating a “conscious” agent. Master's thesis, The University of Memphis.

Bogner, Myles, Uma Ramamurthy, and Stan Franklin (2000). "Consciousness" and Conceptual Learning in a Socially Situated Agent. in Kerstin Dautenhahn ed. Human Cognition and Social Agent Technology

Bogner, Myles, Stan Franklin, Art Graesser and Bernard Baars (in preparation), Hypotheses From “Conscious” Software.

Brooks, Rodney A. (1997), The Cog Project. J. of the Robotics Society of Japan, 15, 968-70.

Damasio, A. R. (1994) Descartes’ Error, New York: Gosset/Putnam Press.

Franklin, Stan (1995) Artificial Minds, Cambridge, MA:MIT Press.

Franklin, Stan. (1997) Autonomous Agents as Embodied AI, Cybernetics and Systems. Special Issue on Epistemological Aspects of Embodied AI 28:6 499-520.

Franklin, Stan, Arpad Kelemen, and Lee McCauley (1998), IDA: A Cognitive Agent Architecture, Proceedings of the IEEE Conference on Systems, Man and Cybernetics, 2646-2651.

Franklin, S. & Graesser, A. (1997) Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. Intelligent Agents III, Berlin: Springer Verlag, 21-35.

Franklin, Stan and Graesser, Art. (1999). Models of consciousness. Consciousness and Cognition. 8 285-305.

Hofstadter, D. R. and Mitchell, M. (1994), "The Copycat Project: A model of mental fluidity and analogy-making." In Holyoak, K.J. & Barnden, J.A. (Eds.) Advances in connectionist and neural computation theory, Vol. 2: Analogical connections. Norwood, N.J.: Ablex.

Holland, J. H. (1986), "A Mathematical Framework for Studying Learning in Classifier Systems." *Physica* 22 D:307–317. (Also in Evolution, Games andLearning. Farmer, J. D., Lapedes, A., Packard, N. H., and Wendroff, B. (eds.). NorthHolland (Amsterdam)).

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

Kanerva, Pentti (1988), Sparse Distributed Memory, Cambridge MA: The MIT Press.

Laird, J.E., A. Newell and P.S. Rosenbloom (1987), SOAR:An Architecture for General Intelligence, Artificial Intelligence, 33:1-64.

Mauldin, M. L. (1994) Chatterbots, Tinymuds, And The Turing Test: Entering The Loebner Prize Competition. In: Proceedings of the Twelfth National Conference on Artificial Intelligence, AAAI Press, 16-21.

Maes, Pattie (1989), 'How to do the right thing', Connection Science, 1:3, 291-323.

McCauley, Thomas L. and Stan Franklin (1998) An Architecture for Emotion, AAAI Fall Symposium "Emotional and Intelligent: The Tangled Knot of Cognition" Menlo Park, CA: AAAI Press, 122-127.

McCauley, Lee, Franklin, Stan, and Bogner, Myles.(1999). An Emotion-Based "Conscious" Software Agent Architecture. Proceedings of the International Workshop on Affect in Interactions Towards a New Generation of Interfaces, Siena, Italy, October 21-22, New York: Springer-Verlag.

Minsky, Marvin (1985), Society of Mind,. New York: Simon and Schuster.

Negatu, Aregahegn and Stan Franklin (1999), Behavioral learning for adaptive software agents, Intelligent Systems: ISCA 5th International Conference

Pepperberg, I. M. (1994), Numerical Competence in an African Grey Parrot. J. Comp. Psych 108, 36-44.

Ramamurthy, Uma , Myles Bogner, and Stan Franklin (1998), "Conscious" Learning In An Adaptive Software Agent, From Animals to Animats 5, 372-377.

Rolls, Edmund T. (1999), The Brain and Emotion, Oxford: Oxford University Press.

Sloman, A. (1996) What Sort of Architecture is Required for a Human-like Agent? Cognitive Modeling Workshop , AAAI96, Portland Oregon.

Song, Hongjun and Stan Franklin (2000), A Behavior Instantiation Agent Architecture, to appear.

Tyrrell, T. (1994). An Evaluation of Maes' "Bottom-Up Mechanism for Behavior Selection". *Adaptive Behavior,* Vol. 2, No. 4, pages 307-348, 1994.

Varela, F.J., Thompson, E. and Rosch, E. (1991) The Embodied Mind. Cambridge, MA:MIT Press.

Wilson, S. W. (1991), "The Animat Path to AI." From Animals to Animats, (editors J.-A. Meyer and S. W. Wilson), Cambridge MA,: The MIT Press.

Zhang, Zhaohua, Stan Franklin and Dipankar Dasgupta (1998), Metacognition in Software Agents using Classifer Systems, Proc AAAI 98, 82-88