Learning Mechanisms for Intelligent Systems

U. Ramamurthy

A. Negatu

S. Franklin

Learning Mechanisms for Intelligent Systems

Uma Ramamurthy[1], Aregahegn Negatu, and Stan Franklin[2,3]

urmmrthy@memphis.edu, negatua@msci.memphis.edu, and stan.franklin@memphis.edu
Institute for Intelligent Systems and
Department of Mathematical Sciences
The University of Memphis

Abstract - In this paper, we describe mechanisms for several different types of learning being implemented in “conscious” software agents. We argue that in complex, dynamic domains, such learning mechanisms are essential for software agents to adapt and effectively ‘live’ in those domains. We further believe that a **development period** is required for knowledge acquisition. Particularly, in complex, dynamic domains where knowledge engineering is expensive, the development period provides a simple, but cost effective solution to knowledge acquisition. We speculate on the implications of these mechanisms for the evolving, complex electronic world, and for human learning and development.

Index Terms – Cognitive Agent, “Conscious” Software Agent, Case-based Reasoning, Development Period, Conceptual Learning, Behavioral Learning.

I. “Conscious” Software Agents

An *autonomous agent* is a system situated in, and part of, an environment, which senses that environment, acts on it in pursuit of its own agenda (Franklin and Graesser 1997}. Autonomous software agents that are equipped with cognitive features chosen from multiple senses, perception, concept formation, attention, problem-solving, decision making, short and long-term memories, learning, emotions, action-selection, etc. are called *cognitive agents* (Franklin 1997).

A “conscious” software agent is defined to be an cognitive software agent that implements Global Workspace Theory (Baars 1988, 1997). (No claim of sentience is being made.) Global Workspace Theory postulates that human cognition is implemented by a multitude of small, special purpose processes, almost always unconscious. Coalitions of such processes, when aroused by novel, problematic instances, find their way into a global workspace and thus into “consciousness”. This limited capacity workspace serves to broadcast the message of the coalition to all the unconscious processes, in order to recruit other processes to join in handling the current novel situation. Thus consciousness allows us to deal with novel and problematic situations. All this takes place in the auspices of goal c ontexts, perceptual contexts, conceptual contexts and cultural contexts. Each context is itself a coalition of processes.

We 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, which 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. 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. Conversely, the results of such experiments motivate corresponding modifications of the architecture and mechanisms of the cognitive 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).

In this paper, we focus on two such “conscious” software agents. “Conscious” Mattie (CMattie) is a “conscious” clerical software agent (McCauley & Franklin 1998, Negatu & Franklin 1999, Ramamurthy et al. 1998, Zhang et al. 1998, Bogner et al. 1999). She composes and emails out weekly seminar announcements, having communicated by email with seminar organizers and announcement recipients in natural language. There is no human involvement other than these email messages. CMattie’s cognitive modules include perception, action selection, associative memory, “consciousness”, emotion, metacognition, and learning. Her mechanisms include variants and/or extensions of Maes' behavior nets (1990) , Hofstadter and Mitchell's Copycat architecture (1994), Jackson's pandemonium theory (1987), Kanerva's sparse

[1]Supported in part by Cancer Center Support CORE grant, P30 CA 21765 and by American Lebanese Syrian Associated Charities (ALSAC)

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

[3]With essential contributions from the Conscious Software Research Group including Art Graesser, Satish Ambati, Ashraf Anwar, Myles Bogner, Arpad Kelemen, Irina Makkaveeva, Lee McCauley, Alexei Stoliartchouk, Ravikumar Kondadadi, Zhaohua Zhang

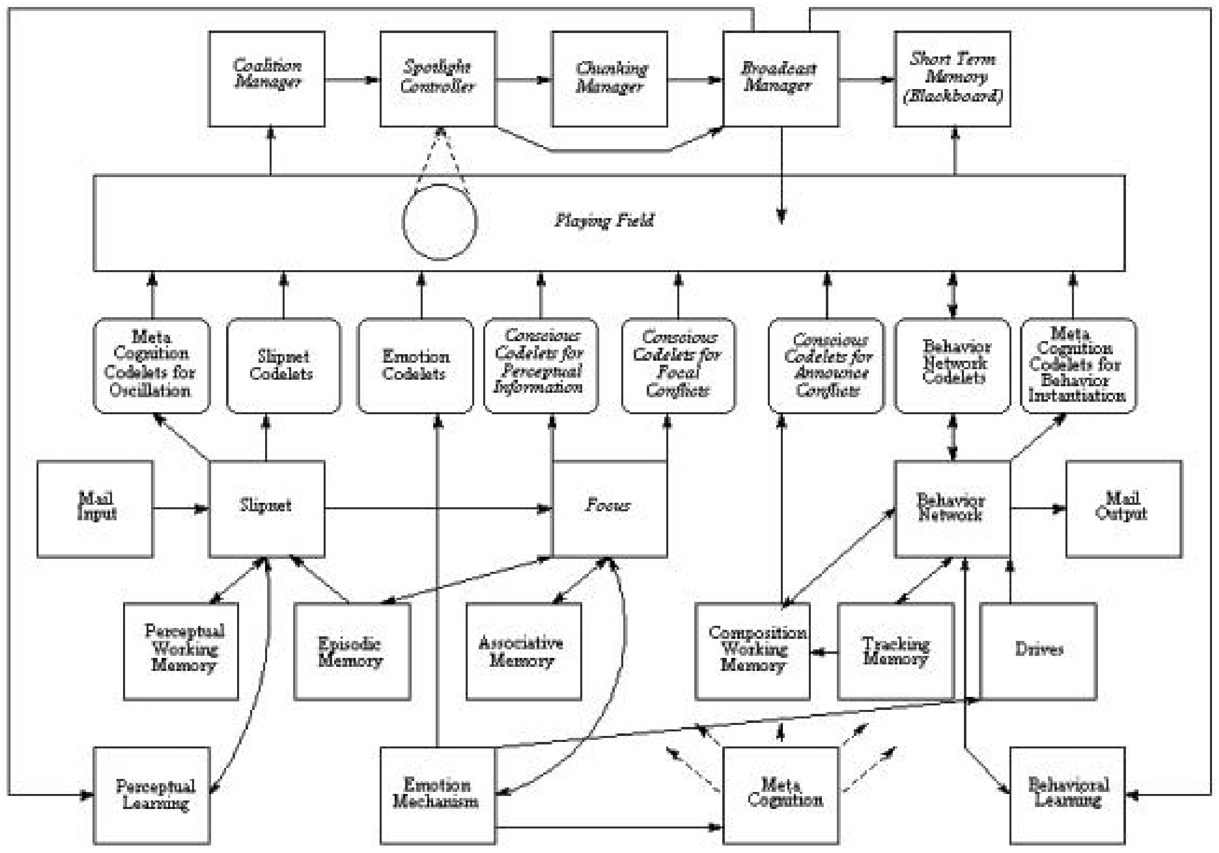
Coalition Manager
Spotlight Controller
Chunking Manager
Broadcast Manager
Short Term Memory (Blackboard)
Playing Field
Meta Cognition Codelets for Oscillation
Slipnet Codelets
Emotion Codelets
Conscious Codelets for Perceptual Information
Conscious Codelets for Focal Conflicts
Conscious Codelets for Announce 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: CMattie's Architecture

distributed memory (1988), and Holland's classifier systems (Holland 1986) .

IDA (Intelligent Distribution Agent) is a "conscious" software agent being developed for the US 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 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. 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 has to access several databases and understand the contents in these databases. She must adhere to some ninety Navy policies and thus ensure that the Navy's needs are satisfied. She must not only hold down moving costs, she also must cater to the needs and desires of the sailor as much as is possible. Finally, she has to write the orders and send them to the sailor.

II. "Conscious" Software Architecture and Mechanisms

These agents have quite complex architectures, with IDA being an order of magnitude more complex than CMattie. Figure 1 details CMattie's architecture.

In both the CMattie and IDA architectures, the processors postulated by global workspace theory are implemented by codelets, which are small pieces of code. These are specialized for some simple task and often play the role of demons waiting for appropriate conditions 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 who situations with novel input (Bogner et al. 1999). 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 information 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.

In CMattie and IDA, the coalition manager is responsible for forming and tracking coalitions of codelets. Such coalitions are initiated on the basis of the strengths of the mutual associations between the member codelets. At any given time, one of these coalitions finds it 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 in the system. The broadcast manager accomplishes this.

Perception Module: CMattie senses text. Her perception consists of understanding the text of incoming email messages in natural language. In sufficiently narrow domains, natural language understanding may be achieved via an analysis of surface features without using a traditional symbolic parser. CMattie's limited domain requires her to deal with about a dozen distinct message types, each with relatively predictable content. This allows CMattie to use surface level natural language processing. CMattie's perception module is implemented based on the Copycat architecture (Hofstadter and Mitchell 1994). The mechanism includes a slipnet that stores the domain knowledge of the agent and a pool of codelets (processors) specialized for specific recognition and identification tasks, along with templates for building and verifying understanding. Together, they comprise an integrated perceptual system for CMattie. With it, she recognizes, categorizes and understands incoming email messages. Message type nodes in the slipnet correspond to perceptual contexts in Global Workspace Theory.

IDA's perception module is modeled after those of CMattie, but is more complex. IDA's perception module must handle incoming email messages from sailors. She has a much more extensive slipnet to hold her domain knowledge and a larger pool of supporting codelets. IDA must also understand data received from the sailors' personnel records, from a listing of currently available jobs and from other related databases.

Action-Selection Module: Both CMattie and IDA depend on a behavior net (Maes 1990) type mechanism for high-level action selection in the service of built-in drives. Each has several distinct drives operating in parallel. These drives vary in urgency as time passes and the environment changes. Behaviors, corresponding to goal context in Global Workspace Theory, are typically mid-level actions, many depending on several 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, 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 there is a successor link from X to Y. There may be several such propositions resulting in several links between the same nodes. Next, whenever there is a successor going one way, there exists a predecessor link going the other. Finally, suppose there exists a proposition m on behavior Y's delete list that is also a precondition for behavior X. In such a case, there is a conflictor link from X to Y, which is 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

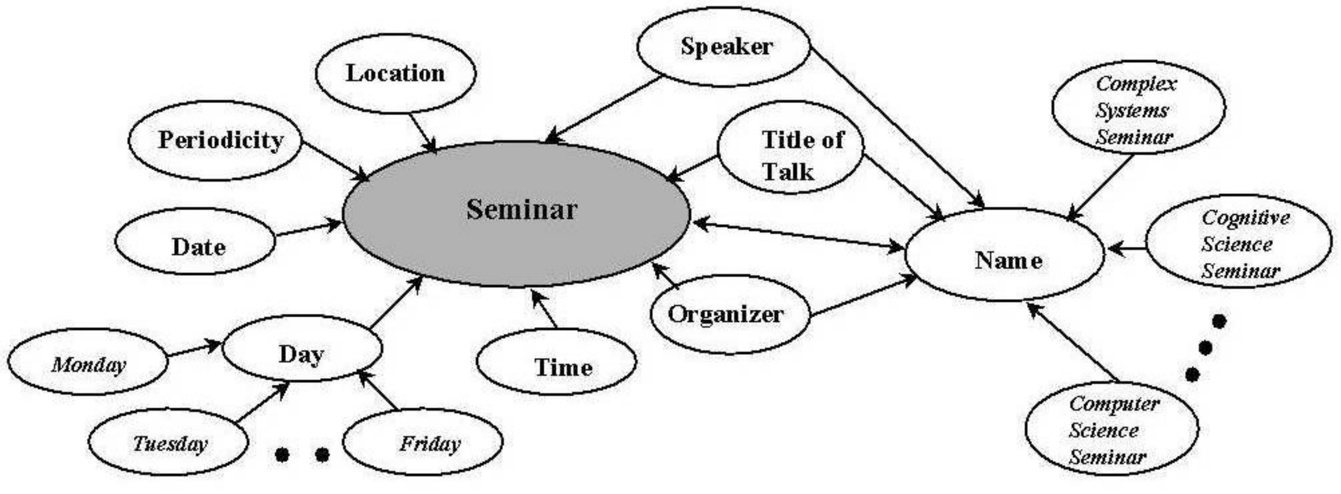

Figure 2: Segment of the Slipnet in CMattie's Perception Module

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. 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.

Action selection via behavior net suffices for CMattie due to her relatively constrained domain. IDA's domain is much more complex, and requires temporal deliberation in the sense of creating possible scenarios, partial plans of actions, and choosing between them (Sloman 1999). 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 arrives at the new billet with in a given time frame after utilizing travel time. 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. They are constructed in a computational workspace corresponding to working memory in humans. We use Barsalou's perceptual symbol systems as a guide (Barsalou 1999). The perceptual/conceptual knowledge base of this agent takes the form of a semantic net with activation called the slipnet, as described above. The name is taken from the Copycat architecture that employs a similar construct (Hofstadter et al. 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, including attention 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 (see Behavioral Learning below) whose goal is to find a billet for a particular sailor. Perhaps a behavior executes to read appropriate items from the sailor's personnel database record. Then, possibly, comes a behavior stream to locate the currently available billets. Next might be a behavior stream 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 mediation by the "consciousness" mechanism, as described in the previous paragraph is characteristic of IDA. The principle is that she 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 unconsciously 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 would allow a "conscious" response to an anomalous data item, for example.

III. Development and Learning

We believe that "conscious" software agents will require a ***development period*** to acquire the necessary domain knowledge of their domains. This was not particularly necessary in CMattie, but is very much needed for more complex agents like IDA. Each detailer is responsible for a particular community of sailors defined by job skills and pay grade, e.g. sonar technicians, E6-E9. IDA, our software detailer agent, is being designed to serve a single community, as do all human detailers. A tremendous amount of knowledge is required, much of it common to all detailers, but quite a lot is specialized to the particular community. For a single such agent serving one community, well-developed AI knowledge engineering techniques will suffice to gather and encode the needed knowledge. But, what about agents for all the other hundred or so communities? A knowledge engineering approach seems prohibitively expensive both in money and time.

We foresee a three-phase development process in IDA. During the first phase, the IDA observes the human detailer in action, acquiring the domain knowledge and learning the new concepts and behavi ors for this acquired knowledge. Such learning would occur by conversational interaction by IDA with the human detailer. During the second phase, IDA interacts with the sailors in the community with the human detailer observing and making suggestions. During this phase, all communication from IDA to the sailor would pass through the human detailer. In phase three, which is initiated by the human detailer, IDA would correspond directly with the sailor, but still receive suggestions from the human detailer. When the human detailer initiates the completion of this third phase, IDA will begin to act entirely independently. Though the development period would have ended by then, IDA will continue to learn from interactions with sailors and commands. IDA, at her own discretion or that of a sailor, can ask for the assistance of a human supervisor.

IV. Learning mechanisms employed

Learning occurs in various modules of these "conscious" agents. We briefly discuss several of these learning mechanisms, and then focus on conceptual and behavioral learning in these agents.

Learning into Associative Memory: There is a type of declarative learning in the Associative Memory modules of these agents. Both CMattie and IDA employ sparse distributed memory (SDM) as their major associative memories (Kanerva 1988). When these agents create a new percept, such a percept, consisting of the incoming information as modified by associations and the current emotion and behavior, is written to associative memory. For a quite unusual percept, this learning in Associative Memory may be only memorization. For more common percepts some sort of generalization takes place as a result of the mechanism of SDM. For example, CMattie learns the usual times of a particular seminar, and will send an inquiry to the organizer should a different time be specified. Due to her much more complex domain, we expect IDA to learn into associative memory in much more sophisticated ways. For example, she may well associate a certain class of sailors, identified by pay grade, job skills, length of obligatory service remaining, etc., with a preference for certain types of billets.

Learning concept codelets: A type of procedural learning occurs in the playing field of these agents. Codelets in CMattie and IDA participate in a pandemonium theory style organization (Jackson 1987). On the playing field we find the active codelets. Some of these are joined in coalitions. One such coalition should lie in the spotlight of "consciousness". Those codelets who share time in the spotlight of "consciousness" have associations between them formed or strengthened, or perhaps diminished if things are going badly. Coalitions of highly associated codelets may merge to form higher-level concept codelets. A concept codelet, when active, performs the same actions as do its member codelets combined. This is comparable to chunking in SOAR (Laird et al. 1987). When the same coalition of codelets, acting in parallel or in sequence, often produce a useful result, this coalition can be expected to merge into a concept codelet. This merging constitutes a second form of temporal proximity learning. Though we have implemented the mechanisms for such learning of concept codelets in both CMattie and IDA, we have as yet not found situations where such learning might be useful in these two agents. We do believe that such learning would be necessary in other complex "conscious" agents.

Learning of emotional influence: In both CMattie and IDA we include mechanisms for emotions (McCauley & Franklin 1998). The mechanism employed is similar to neural network. In IDA, a network is built up by connecting the attention/emotion codelets to key behaviors, goals, drives, perception codelets, etc. The links of this network are to have weights and carry activation. Weights will decay with disuse. Each use tends to decrease the decay rate. Weights will increase according to a sigmoidal function of any activation carried over the link, allowing for Hebbian style learning. Thus, IDA learns to use emotions to affect various cognitive functions as we humans do.

Metacognitive Learning: 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. Metacognition in CMattie is implemented as a fuzzy classifier system in order that it may learn. Learning actions always requires feedback on the results of prior actions. The Evaluator sub-module is implemented by a reinforcement learning algorithm (Barto et al. 1981) that assigns reward or punishment to classifiers based on the next inner percept. It also uses a reinforcement distribution algorithm (Bonarini 1997) to distribute credit among the classifiers. When things are not going too well over a period of time, learning occurs via a genetic algorithm (Holland 1975) acting to produce new classifiers.

V. Conceptual Learning

Perception is accomplished in these agents by recognizing and analyzing the surface features of the email/electronic communication between the agent and the humans that the agent interacts with. The perception modules in both CMattie and IDA consist of (1) an Input/Output module which acts as an email interface between the "conscious" agent and the sailors or human supervisor, (2) a slipnet which is a network of nodes and links containing the agent's permanent perceptual concepts, (3) a working memory that holds the contents of the received input from the agent's domain, the text, and (4) a case-based memory which acts as the episodic memory of the agent, storing episodes and traces of learning that occurs through case-

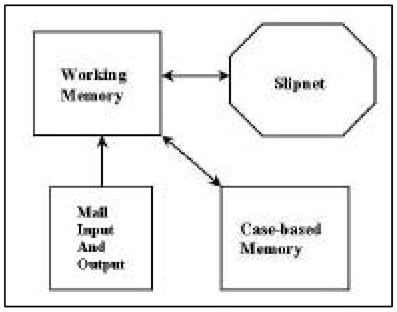

Figure 3: CMattie's Perception Module

based reasoning.

The perception modules of these agents have a limited number of concepts already defined in their slipnets. For example, CMattie "knows" what a seminar is, through the built-in seminar concept and its features. Similarly, IDA "knows" about PRD (projected relocation date), location, training, wait for a new job requisition list, request for a specific billet, etc. These are defined as built-in concepts in IDA's perception module.

When new inputs are sensed in the domain of these agents, they must learn new concepts and possibly new codelets in order to function in such dynamic domains. The conceptual learning in these agents is based on the following two premises: (1) agents learn based on what they already know; and (2) agents see every new situation in terms of at least one previously solved problem.

"Consciousness" acts a facilitator for learning in these agents. Conceptual learning occurs through internal interaction between "consciousness" and the perception module. The "conscious" broadcast recruits codelets that can contribute to the current problem or task for which they are relevant. The activation of such codelets increases, motivating them to become active. One such task is the conceptual learning mechanism in these agents.

Conceptual learning is effected through conversations with other (human) agents in the domain of these "conscious" agents, and implemented using case-based memory and case-based reasoning (Kolodner 1993). To be able to converse with others in their domains, "conscious" agents have some level of natural language capability built into them.

For example, in CMattie, the agent begins a conversation with the seminar organizer when she receives a message that contains words or phrases with which she is not familiar. She categorizes such messages as "don't understand" messages. She may also start a conversation when she receives a "negative" message from a seminar organizer that has some negative connotation. When the perception process is complete, the message type with its relevant features including the unfamiliar words/phrases or the negative connotation is written to the Focus of the "Consciousness" module, the module's interface with both associative and episodic memory. During the next "conscious" broadcast, codelets are recruited which instantiate a behavior stream that results in selecting a composition template to write a "question" message to the seminar organizer asking him/her to explain and help the agent understand those unfamiliar words/phrases or the negative connotation.

When the seminar organizer replies to this "question" message, the agent perceives that as an "answer" message from its surface features. Perception of an "answer" message triggers a "conscious" read from case-based memory. Such a memory-read retrieves the "question" message that elicited this "answer" message and all the messages related to this particular case. This provides the agent with the context that is necessary to continue the conversation and facilitate learning of new concepts.

Consider the situation that IDA will encounter when the Navy acquires a new type of sonar equipment. IDA has to learn about this new equipment.. She must learn about the training required for this equipment, the facilities where such training will be given, and where to look for dates and times.

When a sonar technician writes to IDA expressing his interest in undergoing training for the new equipment and in getting posted to a position where such equipment is in use, IDA will contact a human supervisor to find out about this new sonar equipment. Through "conversation" with the human supervisor and by her case-based reasoning capabilities, she will learn that the new equipment is a type of sonar equipment, its name and other specifics about the new equipment. (Since each equipment has specific NEC, she will eventually learn that this new equipment would require job qualification and training.) Since IDA already has other types of sonar equipment and their related information built in her perception module, she will add necessary new slipnet nodes and related new codelets to the perception module. To complete the acquisition of this new domain knowledge in her perception module, new relevant links between existing slipnet nodes and the new nodes will be created. Once this conceptual learning process is complete, IDA will be able to perceive and communicate effectively with sonar technicians about this new equipment, the qualifications and the training required for this new equipment.

IDA stores a trace of such learning in her case-based memory to enhance her case-based reasoning capabilities. The case-based reasoning in these agents depends on their past experiences and their ability to relate to new problems that they encounter in terms of their past experiences.

In CMattie, similar conceptual learning occurs when she encounters new concepts that are similar to her built-in concepts. This learning mechanism is based on viewing every novel situation in terms of a previously known and solved problem. When CMattie receives a message about a non-seminar event, say a Colloquium, she treats it as a seminar event and sends an acknowledgement to the sender stating that she is initializing a new seminar by the name "Colloquium Seminar". This acknowledgement might elicit one of several possible responses from the sender:

(1) The sender might send her a negative response, which will result in a conversation and thus an episode;

(2) CMattie includes the Colloquium Seminar in her weekly seminar announcement as the sender ignores her incorrect acknowledgement. This weekly announcement with the incorrect seminar might elicit a negative response from the sender;

(3) The sender might ignore the incorrect weekly announcement, but is likely to respond when CMattie reminds him the following week about the pending Colloquium seminar slot in the next week's seminar announcement;

(4) The sender might ignore all of CMattie's reminders. That itself acts as a feedback to CMattie, giving rise to a new concept which is similar to the seminar concept but with a different periodicity feature.

Regardless of which route the episode generation takes, CMattie eventually learns a new concept called Colloquium that is closely related to her built-in seminar concept.

In these "conscious" software agents, there are two main capabilities for bringing about conceptual learning. The agent has codelets in her perception module that look for words and phrases that she has not previously encountered. The perception module tracks such new words and phrases that occur with regularity by maintaining statistics and recognizing novelty. This aids in the agent's natural language understanding. The agent is capable of understanding messages with negative connotations.

Every time the agent learns a new concept, a trace of such learning is stored in the agent's case-based memory. These stored episodes enhance her case-based reasoning capabilities. After learning the Colloquium concept, CMattie might encounter a Dissertation Defense message. Since her case-based reasoning capability depends on past experiences and her ability to understand new situations in terms of her past experiences, she recalls her experience with the first Colloquium message and will be able to relate the two concepts together through her conversation with the sender of the message.

When the agent learns a new concept, a new slipnet node has to be created, with its related set of features implemented as other nodes. Thus, based on the context and relationships, these newly generated nodes have to be connected to the existing nodes in the slipnet. The slipnet is implemented in XML and with these agents being implemented in Java. The creation of new nodes becomes effective and simplified through the modification of and addition to the XML tree.

Once the agent has learnt a new concept in her perception module, she will need new behaviors to act effectively when she encounters instances of these new concepts in her domain. The behavioral learning mechanism (discussed next) in these agents enable them to learn new behaviors to act effectively in their dynamic domains.

VI. Behavioral Learning

Any agent (artificial or biological) senses, perceives and acts in order to satisfy its built in drives. The ever-present challenge for any agent is to produce the appropriate action relevant to internal states modulated by a perceived environmental situation. That is, the action selection mechanism of an agent decides what to do next.

In CMattie and IDA, new concepts get introduced via their conceptual learning mechanisms. New concepts require new behaviors, thus requiring an action selection module with a capability to learn. As shown in Figure 4, our behavioral learning system, to realize its adaptive action selection capability, uses four major components:

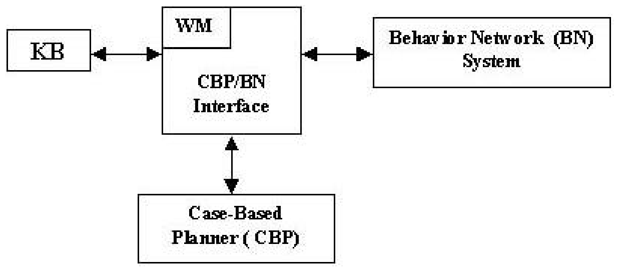

Figure 4: Behavioral Learning Module

1) The *behavior network (BN) system* can be viewed as a collection of behavior streams (action plans). Each such stream is a connected partially ordered set of behaviors (plan operators) that serve to satisfy a goal or sub-goal of the agent. A behavior stream is a partially ordered plan which guides execution of behaviors (plan operators) so as to effect the required transition from the initial state (mainly dependent on the internal representation of the perception) to the goal state. The BN system has additional functions including interface with consciousness and priming.

2) The *Case-Based Planner (CBP)* is a case based reasoning (CBR) system (Kolodner 1993). In general, a CBR system is a paradigm that solves new problems by adapting prior solutions to old problems and, to do so, it supports retrieval, adaptation, and retention processes. In our system, the CBP (Monza-Avila 1998, Veloso 1994) must have a flexible plan learning/adaptation mechanism. The CBP's processes operate on a unit of information called a *case*. In our adaptive action selection mechanism, a case is represented as a triplet consisting of <problem description, solution, outcome>. A *problem description* includes the initial state of the problem situation (the contents of the focus, relevant coalitions of codelets, and feature values of relevant concepts, relevant registers in working memory, etc), one or more (sub)goals that need to be satisfied in such a problem situation, and associated behavior streams (action plans) that achieve those goals. A *solution* is an action plan (behavior stream) whose execution beginning at the initial state of the problem achieves its stated (sub)goal(s); each of which in turn satisfying one or more of the innate drives which are that represent the primary motivation of the agent. An *outcome* is the expected result (for example, feedback from a human) when the solution plan is applied in the initial state.

3) The *Knowledge-Base (KB)* is used to store information needed in the behavioral learning process. That is, it contains all domain related knowledge, built-in and/or learnt, which is specific to the agent's action selection mechanism.

4) The *CBP/BN-Interface* module uses the KB module to couple the BN and the CBP modules, and to facilitate the knowledge acquisition process. It is used to (a) store newly acquired domain knowledge into the KB, (b) compile the problem description (from the BN side) in the format the CBP can use, (c) format a newly obtained plan (from the CBP) so that it can be integrated into the BN system and (d) facilitate effective conversation with human (via the BN) by providing information available in the CBP and/or the KB. The CBP/BN-interface uses its own working memory (WM).

To start with, our "conscious" agent is provided with a BN including a set of behavior streams (action plans) capable of producing actions appropriate to already known current concepts and situations in the domain. In IDA, there are built in streams to produce actions that a detailer performs relative to already known constraints such as available jobs and their required skills, a sailors' skills and experience, etc.

It is important that these agents adapt to new situations in their environments. In the case of IDA, change could come from the addition or deletion of a type of job, an introduction of a new technology and the corresponding skills needed by sailors, or by a change of Navy policy (constraints) under which job assignments are made.

Behavioral learning here is based on two principles: (a) the agent will use past experience to learn new behavior streams by adapting old plans that worked in similar situations, (b) the agent must carry on conversations with humans to acquire new domain knowledge. This also allows for feedback on the accuracy of new plans and for necessary revisions.

Suppose that IDA has learnt of new sonar equipment having been acquired. Once this equipment is conceptualized, IDA should learn how to produce the appropriate actions in relation to this newly acquired concept. That is, IDA should learn to formulate the new streams needed to interact with sailors and make the appropriate job assignments by acquiring the knowledge of what to do relative to the new equipment. The changes that are needed to be made include creation of new attention codelets, new scripts, new behavior priming codelets and new behavior streams.

In CMattie's domain, the perception module can learn a new concept "colloquium" based on the already known concept "seminar". As explained above, one of the differences that the colloquium and seminar concepts have is based on the feature value of "periodicity": seminar is regular and colloquium is irregular. One of the tasks CMattie does is to send reminders to a human organizer of a particular seminar to send her the information for the coming seminar so that she can distribute her regular posting in time. If a human organizer is dealing with colloquium, which is held irregularly, he or she shouldn't receive reminders to send colloquium information. This means, CMattie should know not to send reminder messages to organizers when she deals with colloquium. As discussed in a previous section, the perception mechanism learns the colloquium concept based on what has been known about the seminar concept. CMattie should have the capability to adapt its actions to do the right thing as its domain changes. That is, with the addition of the colloquium concept in its domain, it should learn not to send reminder messages to colloquium organizers while she continues to send reminders to seminar organizers.

When a newly learned concept is perceived, the "consciousness" mechanism broadcasts all the relevant information to recruit codelets, which will collectively pick the appropriate behavior stream(s) that will produce an appropriate response. Since a new concept is involved, the selected stream may fail to produce an appropriate action. This failure initiates the behavioral learning cycle. The learning happens by processing a conversation that the agent has with its human supervisor. At each interchange, the learning mechanism adapts streams from old solutions stored in the CBP system. A single interchange may not suffice to produce an appropriate new stream (action plan). But, episodic memory (implemented using case-based memory) stores the sequence of interchanges and the trace of the reasoning used in building a new behavior stream. This, along with the already acquired domain and control knowledge stored in the KB and CBP modules, will help in the effective use of past experience to speed up the learning process. A successfully learnt stream in the CBP module gets integrated into the BN system where it can be instantiated and executed.

In addition to learning new streams, the behavioral learning process must include the creation of new coalitions of behavioral codelets that will choose and instantiate the new stream whenever it becomes relevant. It must also create the codelets that will implement the actions of each individual new behavior. All this is accomplished by copying and modification of existing codelets.

The behavioral learning of "conscious" agents happens on-line where its human supervisor (domain expert) monitors and gives feedback on what the agent should do in relation to the change in its domain.

VII. Implications for today's electronic society and human learning

In complex, dynamic domains, learning is essential for a software agent to be intelligent and adaptable to changes in its domain. This is definitely true of today's electronic societies. Additionally, the agent interacts with humans and that brings in an additional layer of dynamism to the agent's domain. All these aspects make it essential for agents to have learning mechanisms. As we have discussed in this paper, an intelligent system needs several learning mechanisms to be implemented in it, if that system is to be life-like, adaptable and smart. In these agents, learning via several types of mechanisms allows them to become closely coupled to their domains, enabling easy adaptation.

Intelligence never comes cheaply. Our "conscious" software agents require a tremendous amount of knowledge engineering to live in their dynamic domains. In such agents, a development period offers a cost-effective solution to the expensive task of knowledge engineering. By watching a human-expert and interacting with that human-expert, the agent acquires domain knowledge and expertise in a simple and cost-effective way.

A basic tenet of global workspace theory (Baars 1988) says that consciousness is *sufficient* for learning. This is certainly true in our agents. The contents of "consciousness" are routinely written to associative memory. Is consciousness also *necessary* for learning? The learning of new associations between codelets and adjustments to such associations happens when their contents become "conscious." But it also occurs to a lesser extent when the codelets are active together, but unconscious. Our chunking mechanism also does not routinely come to consciousness, though a newly learned chunk may well have been "conscious." This seems to suggest that some procedural learning, some gradual improvement of skills may occur unconsciously with practice.

It also seems that our metacognitive learning occurs unconsciously. This must be at least partly a flaw in our model, since at least some metacognitive learning in humans happens consciously. We also suspect that some learning of weights in emotion networks of these agents will occur unconsciously. This seems quite possibly true of humans as well.

Our goals with these "conscious" software agents are both cognitive modeling and creation of useful, human-like, information agents that will 'live', learn and grow in today's complex, dynamic world of e-commerce, e-learning and e-business.

References

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.

Barto, A. G., R. S. Sutton, and P. S. Brouwer. 1981. Associative Search Network: a Reinforcement Learning Associative Memory. *Biological Cybernetics* 40:201–211.

Bogner, M., U. Ramamurthy, and S. Franklin. 1999. "Consciousness" and Conceptual Learning in a Socially Situated Agent. In *Human Cognition and Social Agent Technology*, Advances in Consciousness Research Series, 19. Ed. K. Dautenhahn. Amsterdam: John Benjamins.

Bonarini, A. 1997. *Anytime Learning and Adaptation of Structured Fuzzy Behaviors. Adaptive Behavior Volume 5*. Cambridge MA: The MIT Press.

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., and A. C. Graesser. 1997. Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. In *Intelligent Agents III*. Berlin: Springer Verlag.

Franklin, S., A. Kelemen, and L. McCauley; 1998; IDA: A Cognitive Agent Architecture. IEEE Conf on Systems, Man and Cybernetics.

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

Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press.

Holland, J. H. 1986. A Mathematical Framework for Studying Learning in Classifier Systems. In *Evolution, Games and Learning: Models for Adaption in Machine and Nature*, vol. al, Amsterdam, ed. D. Farmer. : North-Holland.

Holland, J. H., H. J., and Reitman J. S... 1978. *Cognitive Systems Based on Adaptive Algorithms. In: Pattern Directed Inference Systems (pp. 313-329), eds. D. A. Waterman & F. Hayey-Roth*. New York: Academic Press.

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

Kanerva, P. 1988. *Sparse Distributed Memory*. Cambridge MA: The MIT Press.

Kolodner, J. 1993. *Case-Based Reasoning*. : Morgan Kaufman.

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

Maes, P. 1990. How to do the right thing. *Connection Science* 1:3.

McCauley, T. L., and S. Franklin; 1998. An Architecture for Emotion. AAAI Fall Symposium Emotional and Intelligent: The Tangled Knot of Cognition"; AAAI; Orlando, FL.

Monza-Avila, A. Integrating twofold Case Retrieval and Complete Decision Replay in CAPlan/CbC, Doctoral dissertation, Dept. of Computer Science, University of Kaiseslautern, 1998.

Negatu, Aregahegn and S. Franklin, 1999. Behavioral Learning for Adaptive software Agent. Intelligent Systems, Proceeding of the ISCA 8th International Conference, pp. 91 – 95, Denver, Colorado.

Ramamurthy, U., S. Franklin, and A. Negatu. 1998. Learning Concepts in Software Agents. In *From animals to animats 5: Proceedings of The Fifth International Conference on Simulation of Adaptive Behavior*, ed. R. Pfeifer, B. Blumberg, J.-A. Meyer , and S. W. Wilson. Cambridge, Mass: MIT Press.

Sloman, A. 1999. What Sort of Architecture is Required for a Human-like Agent? In *Foundations of Rational Agency*, ed. M. Wooldridge, and A. Rao. Dordrecht, Netherlands: Kluwer Academic Publishers.

Veloso, M., Planning and Learning by Analogical Reasoning, Lecture Notes in Artificial Intelligence, Springer Verlag, 1994.

Zhang, Z., D. Dasgupta, and S. Franklin. 1998. Metacognition in Software Agents using Classifier Systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*. Madison, Wisconsin.