IMPLEMENTING EMOTIONS IN AUTONOMOUS AGENTS

A Thesis
Presented for the
Master of Science
Degree
The University of Memphis

Thomas Lee McCauley
August 1999
DEDICATION

To Helen
ACKNOWLEDGEMENTS

I would like to thank my major professor, Dr. Stan Franklin, for his encouragement and unguarded criticism. His ideas and council were invaluable during the course of this research. I would also like to thank the other members of my committee, Dr. King-Ip Lin and Dr. Dipanker Dasgupta, for dedicating a portion of their valuable time and knowledge to my advancement. Much of this work was done within the bailiwick of the Conscious Software Research group. This group was of great help in refining many of the ideas presented here, and their assistance is very appreciated. My parents deserve special recognition. I may have made the journey, but they provided the first steps. Finally, to my wife, Helen, words cannot express my gratitude. You are my inspiration and the example of the person I most want to be like.
ABSTRACT

Emotions are not generally considered to be an advantage when one wishes to build an artificial intelligence system that acts in a rational manner. However, the opposite is, in fact, the truth. Emotions are an essential part of intelligence and rational behavior. Recent evidence points to the fact that humans who have suffered damage to key areas of the brain responsible for communication to the limbic system, generally regarded as the seat of emotion in the brain, cannot function in a reasonable way either socially or logically. This evidence also gives us a glimpse into what emotions really do for cognition. From here it is possible to begin to create ways of giving emotional functionality to artificial systems.

One such model of an emotional mechanism is detailed that allows an agent to represent a broad range of emotions. In addition, the mechanism gives the agent the ability to learn complex emotions from a set of primitives and to associate actions that will pursue or avoid particular emotional stimuli. Two implementations, called CMattie and IDA, are being designed with this emotion component.

The role of learning in systems based on emotions must also be explored. The decades old paradigm of learning via an external evaluation function or human observer is fully replaced by internal emotional judgement. A portion of this judgement must be designed into the agent from its inception, and in such a way that the agent continues to act rationally after learning more complex emotions.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 ARE EMOTIONS REALLY NECESSARY?</td>
<td>2</td>
</tr>
<tr>
<td>1.2 THE FUNCTIONS OF EMOTIONS</td>
<td>4</td>
</tr>
<tr>
<td>2.0 WHAT ARE EMOTIONS?</td>
<td>7</td>
</tr>
<tr>
<td>2.1 A FORMAL DEFINITION</td>
<td>8</td>
</tr>
<tr>
<td>2.1.1 Autonomous Agents</td>
<td>8</td>
</tr>
<tr>
<td>2.1.2 Perceptions</td>
<td>9</td>
</tr>
<tr>
<td>2.2 COMPONENTS OF EMOTION</td>
<td>11</td>
</tr>
<tr>
<td>3.0 IMPLEMENTATION OF EMOTION</td>
<td>14</td>
</tr>
<tr>
<td>3.1 PANDEMONIUM THEORY</td>
<td>15</td>
</tr>
<tr>
<td>3.2 CONSCIOUS MATTIE (CMATTIE)</td>
<td>18</td>
</tr>
<tr>
<td>3.3 AN INTELLIGENT DISTRIBUTION AGENT (IDA)</td>
<td>22</td>
</tr>
<tr>
<td>3.4 EMOTIONS IN CMATTIE AND IDA</td>
<td>28</td>
</tr>
<tr>
<td>3.4.1 Theory</td>
<td>28</td>
</tr>
<tr>
<td>3.4.2 Code Description</td>
<td>31</td>
</tr>
<tr>
<td>3.4.3 Implementation Demonstration</td>
<td>33</td>
</tr>
<tr>
<td>4.0 A NEW PERSPECTIVE ON LEARNING IN ARTIFICIAL SYSTEMS</td>
<td>38</td>
</tr>
<tr>
<td>5.0 CONCLUSION</td>
<td>43</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>45</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>53</td>
</tr>
</tbody>
</table>
TABLE OF FIGURES

Figure 1.1: Paul MacLean’s diagram of the “triune brain.” .................................................3

Figure 3.1: CMattie’s architecture (Bogner, Ramamurthy, and Franklin 1998) .......... 20

Figure 3.2: IDA’s architecture (Franklin, Kelemen, and McCauley 1998) ................. 24

Figure 3.3: Perception Bank in CMattie (Bogner, Ramamurthy, and Franklin 1998) .. 27

Figure 3.4: Typical Wumpus World screen. .................................................................35

Figure 3.5: A situation where the likelihood of danger is high. The hunter will, most likely, climb out of the cave without proceeding. .................................................37
CHAPTER 1

INTRODUCTION

We know the truth, not only by the reason, but also by the heart.
Blaise Pascal
*Thoughts. Chap. X. 1.*

Human beings are inundated on a moment to moment basis with the experience of existence. From the non-conscious feel of our feet hitting the pavement as we run to the very conscious sight of the snarling dog that is chasing us – we *experience* life. Just now, as you read the previous sentence you, hopefully, experienced an unexpected shift in context that resulted in, if not humor then, at least, puzzlement. In addition to the levity that the situation may have evoked, you might have experienced some empathy with being chased by an angry dog. You might have even felt the slightest bit of fear resulting from a prior incident similar to this situation. The feelings just mentioned, however, do not require a special circumstance such as the threat of bodily harm to be present. In fact, humans live in an ocean of emotional sensation that is consciously experienced only to the very slightest of degrees. Amazingly, we seem to be aware of just that portion of sensation that is of most importance to our lives at any given moment; only the most *salient* pieces of information are felt.

Salience, along with valence and intensity, are major components of emotions. In addition, human emotions have a saturation level and display short
term habituation. The first part of this paper will explain these components and set forth a working definition of emotions as well as arguments for its validity.

1.1 Are Emotions Really Necessary?

It is still very much an open question as to where emotion comes from; what part of the computational system we know of as the brain elicits this seemingly non-computational response? While this is an important question to answer, a more important question for computer science and artificial intelligence is whether emotion is important to the function of animals, humans, and autonomous agents in general. Current information suggests that emotions are not only important for rational behavior, but essential (Cytowic 1993; Demasio 1994). The course of evolution has traversed a myriad of possible paths in an effort to produce highly efficient brains that exhibit highly rational behavior. The reason often given from an anthropological standpoint for humans’ superior cognitive abilities is that humans developed an especially large brain (Relethford 1997). Specifically, the increased size of the cortex is attributed with the success of the human mind. The majority of direct measurement studies that have been conducted on the human brain up to this point have been focused on the cortex, mainly because of its position closest to the skull. Paul MacLean’s diagram of the “triune brain” (Figure 1.1), while oversimplified, is a good explanatory device. The first and oldest layer of the brain is shown to be the Reptilian layer. The
Paleomammalian layer of the brain, also known as the limbic system, is considered to be fairly old since it is common to almost all mammals. The Neomammalian layer, which includes the cortex, is shown as completely surrounding and, thereby, dominating the two older layers. This layer is considered to be the youngest portion of the brain evolutionarily. What this diagram misrepresents is that the limbic system did not stop evolving while the cortex developed; on the contrary, the human limbic system is the most highly developed in the animal kingdom. In addition, it is not quite as separate from the cortex as the diagram would have you believe, the limbic system and the cortex are interconnected to such a degree that one cannot act without influencing the
other (Cytowic 1993). While the cortex is considered to be the seat of logic and reasoning in the brain and the limbic system is considered to be the seat of emotions, it is not the case that a larger cortex results in a more rational entity. For example, the Australian spiny anteater has a tremendously large frontal cortex with relation to the overall size of the brain. “If we had frontal lobes as massive as the spiny anteater’s per unit volume of brain we would have to carry our heads in front of us in an extra-large wheelbarrow” (Cytowic 1993). The point here is to show that the development of a large cortical area for analytic judgement without the reciprocal development of the limbic area for emotional judgement does not necessarily produce a more intelligent creature. The distinction between humans and other animals is not the fact that we have a relatively large cortex, nor is it the development of the limbic system alone that accounts for humanity’s peculiarity. Instead, the distinctive feature is that both the cortex and the limbic system co-evolved (Cytowic 1993).

1.2 The Functions of Emotions

Physical evidence may suggest that the limbic system and consequently emotional judgement is necessary for rational behavior, but what functions do emotions play that are so important? Some evidence was presented by Antonio Damasio in his book, Descartes’ Error (Damasio 1994) regarding the necessity of emotions for rational behavior. One of Damasio’s patients, “Eliot,” has a frontal lobe disorder resulting from tissue damage caused by a brain tumor. As is the case with most of Damasio’s patients, Eliot has damage in a key area of the cortex
that communicates with the limbic system. While Eliot scored average or above average on several different tests including IQ tests, when it came down to real life decisions, he would often resort to a seemingly infinite search of possible actions. For example, a simple decision as to when a meeting should be scheduled would result in Eliot considering every possible option and factor no matter how small or unlikely. Damasio’s conclusion is that Eliot is lacking “somatic markers.” These markers would allow the brain to constrain the search to those items that are associated with good feelings or that are not associated with bad feelings. In general, Damasio’s patients consistently make disastrous decisions and do not seem to have the ability to learn from their mistakes. They might make a risky investment that falls through only to repeat the decision multiple times with the same result (Damasio 1994).

The fact that the communication between the cortex and the limbic system is impaired in these individuals allows us to make some judgements as to the purpose of the interplay between these two systems. It appears that the limbic system provides a filter for incoming sensory data as well as providing a relevance measurement for internal data. What this means is that the limbic system assists in narrowing both incoming and internal data by attaching appropriate salience to the signals. All incoming and outgoing signals must go through the limbic system; in particular, the hippocampus appears to be the converging point (Cytowic 1993; LeDoux 1989). At a more basic level, emotions give humans the ability to assess our situation quickly and without a great deal of computational effort. They provide a short cut for determining if the current state
of the world, including our own bodies and minds, is beneficial or detrimental. Learning can then take place based on internal measures as opposed to relying on some external system for performance evaluation as is necessary in many artificial systems.

If emotions are important, then how can they be modeled in computationally based systems? The second section of this paper will focus on one way to model emotions within autonomous agents. The model is being implemented as part of the Conscious Mattie (CMattie) and Intelligent Distribution Agent (IDA) projects.
CHAPTER 2

WHAT ARE EMOTIONS?

The definition of emotions is still a hotly debated issue, and will not be resolved here. The description presented below will not answer all of the questions concerning emotions. For a list of twelve open questions in emotion research see Lazarus (1991). This section will, however, explain the operational definition used for the formulation of the mechanisms detailed in future sections.

In attempting to discover the essence of emotions in humans, one is quickly struck by the enormity and breadth of the problem. There has been a great deal of research done in the field of psychology, neurology, and philosophy that all get at the foundations of emotions in one way or another. The psychological literature tends to discuss the way that emotions are triggered and how they affect our overall psyche (Anderson 1995; Ortony 1988) while the neurological studies try to explain how the brain manifests emotions at the level of neuron firings and neurotransmitter diffusion (Cytowic 1993; Damasio 1994; LeDoux 1989; Pribram 1980; Scherer 1993). The philosophical ideas that relate to emotions are usually found, as they should be, in essays on reason or qualia and tend to focus on why emotions are important or harmful (Dennett 1978; Dretske 1996). Plato, for instance, argued that emotions were to be avoided and that men should strive for a path of pure reason (Grube 1981). Aristotle, on the other hand, believed that emotions are their own form of reason; that they should be
understood as more than just animalistic passions and as an integrated part of thinking (Cooper 1960). No one area of study holds the key to the makeup of emotions. Taken together, however, the information may point us down the right path towards a solution.

2.1 A Formal Definition

Emotions are the feelings that are coupled with external perceptions and internal judgements within an autonomous agent. There are two parts of this formulation that need to be explained. First, there must be a clear definition of what constitutes an autonomous agent (Franklin and Graesser 1996).

2.1.1 Autonomous Agents

Put simply, an autonomous agent is an agent that is coupled to its environment in such a way that it can sense its environment and act, over time, in pursuit of its own agenda in order to change what it senses in the future. The coupling of an agent to its environment means that the agent's senses and actions are particular to that environment. An artificial agent, for instance, that is designed to play chess would play a lousy game of checkers. A biological example would be a fish out of water. In addition to this coupling of an agent to its environment, the agent's actions must effect the environment so as to change what the agent senses from the environment. If this is not the case, then the agent's actions are random and meaningless. The agent's agenda or goals are of particular importance in maintaining the autonomy of the agent. As long as the agent is generating and pursuing its own goals through its actions and no other
agent has direct control over which goals are chosen, then the agent can be considered autonomous. It should be noted that the software industry has moved in the direction of agent based programming as the next step in coding paradigm evolution.

2.1.2 Perceptions

For the next part of our formulation of emotions, let us decide on an appropriate definition of perception. As it is with many of the words that we use to describe functions of the brain, there is no clear boundary between what we call sensations and what we call perceptions. The common definitions are that, in general, sensations are considered to be the “raw” data coming from the environment and perceptions are considered to be information created after some processing of the sensations. An example of a sensation would be the smell of peach cobbler fresh from the oven. A perception would be the association of the smell with the source of the smell, namely the peach cobbler. The fuzzy boundary between these two concepts becomes apparent when you consider the question of whether we can be conscious of a sensation. If consciousness requires at least some relevance processing and a sensation is the raw data from the environment before any processing has taken place, then how could consciousness receive a raw sensation? Therefore, by definition a conscious awareness must be a perception. But what about that smell of peach cobbler, surely that is raw sensory data? As was discussed previously, all signals entering the brain must be processed through the limbic system. The signals coming from your olfactory nerves are only that – signals; something must translate those
signals into what you feel as a smell. Every feeling a person has can be shown in this way to be a result of processing and, therefore, must be a perception by the quite rigid definition presented. For this reason the concept of perception encompasses the full range of things originating in the outside world that can be felt. It is possible that one of the results of the convergence of signals in the hippocampus is the phenomenon of consciousness. It is, therefore, also possible that what we think of as sensations are signals that have been minimally processed and arrive at the hippocampus before the signal is sent on to the cortex for explicit categorization. The intention here is not to get rid of the term "sensation," it is merely to provide an understanding that there is no clear distinction between sensation and perception.

This convergence of sensation and perception is not trivial. In particular, it leads to the realization that emotions are not just the strong reactions that people have when they win the lottery or find out that a loved one has died. Emotions are also in the subtleties of life, in the smells of your house that you never seem to notice or the color of the stop lights that seem to actively block your progress to work. Most of what goes on in our daily lives, even the little things that have no real consequence beyond the moment that they happen, are strongly emotional in nature.

What we are left with is that emotions are inescapable. No matter how much we might want to be like Data, the android character in the Star Trek: Next Generation movies, who can turn his “emotion chip” on and off, humans cannot function without emotional content. Besides, what would life be like without
emotions? For instance, how would you describe your drive to work if there were no emotional difference between the wreck you witnessed and when you passed the beige station wagon?

2.2 Components of Emotion

Emotions are much more than a linear or one-dimensional evaluation of an agent’s current state. The emotional state of an agent is a complex combination of many different factors as well as, possibly, many different emotions. The factors or components that determine an individual emotion, briefly mentioned previously, include salience, valence, and intensity. Also, emotions display some important properties, such as saturation and short-term habituation, which need to be included in any satisfactory model.

One of the most important functions of emotions is that they provide for a relevance filter. This process is called determining the salience of an event, object, situation, etc. Salience is a complex task to accomplish computationally. For example, an airplane pilot must decide which of any number of different gauges or lights are relevant for a particular situation. The light that says that the landing gear is up is not a salient feature at 30,000 feet, but at 1000 feet during a landing, this same feature is very salient. With a dynamically changing environment where the number of possible salient features are large and the situations that would dictate that set of features are almost infinite, the calculation of salience can become an endless search through an incredibly large search space. Emotions provide a way of drastically narrowing the search space for any given situation.
The intensity of an emotion is the strength at which it fires. This measure, along with valence, is a major factor effecting the final salience of a percept of inner judgement. Intensity is, generally, a measure of how well a feature matches or does not match an expected pattern. Sadness, for example, might occur if an agent experiences a pattern involving the loss of an important thing. This is a very abstract example but can be played out in any number of ways. A person might lose the phone number of a friend or find out that their pet has died. These two items elicit similar emotional responses but at different intensity levels. Surprises, on the other hand, usually occur when an expected pattern is not present. The airplane pilot, for instance, might very well be surprised to note that the landing gear is not down during the landing.

The other value that contributes heavily to the salience of an emotion is the valence. In a simplistic way, valence can be thought of as denoting whether an emotion is positive or negative. Obviously, human emotions are not this simple. For humans, valence is the category of feeling associated with an emotion. These, possibly, many categories can, generally, be separated into good and bad emotions; therefore, we use the positive/negative metaphor as an approximation of valence.

There is a limit to how strong an emotion can be. When a particular emotion fires numerous times in succession it will become stronger until it reaches its saturation point. This is the point at which any further stimulation of this same type will not increase the emotion experienced. Also with repeated stimulation comes short-term habituation. This is when an emotion has been
experienced multiple times in close succession causing the increase in emotional
strength of future firings of the same emotion to decrease. A good example of
habituation would be when a person is surprised by a load unexpected noise. It
only takes two or three load and unexpected noises heard in a short period of time
before the person is no longer surprised at the level that the same stimulus would
have evoked if encountered in isolation. Each person tends to have different
levels at which particular emotions become saturated or habituated. If someone's
levels are too far from the normal levels accepted by society, then the person can
often be categorized as having a psychological malady.
CHAPTER 3

IMPLEMENTATION OF EMOTION

The next step is to explore ways of integrating emotions into computationally based models of intelligence. Unfortunately, we must learn to crawl before we walk. The method described here cannot be argued to have a full range of emotions. It does, however, represent our first steps in understanding emotions in an artificial system and a way to enhance some existing systems with emotional content.

Up to this point, Artificial Intelligence researchers have concentrated on modeling many other cognitive abilities such as formal logic, natural language, perception, learning, and many others. While these have been considered keys to intelligence in the past, evidence continues to mount indicating that emotions are essential to the function of all of these tasks. It must, therefore, be considered that there is a limit to the success of any model that does not include emotions.

With this goal in mind, what additional functions would an emotional model need? What criteria are necessary to create such a system? The model must be able to (1) represent a wide range of emotions, not just the few basic ones that we can clearly differentiate, but also the myriad of more complex and learned emotions, (2) make emotional judgements about incoming data as well as internal perceptions, and (3) remember and learn from past emotional experiences. The method described below satisfies all of these criteria. The system can display a
wide range of basic and complex emotional responses to both internal and external stimuli and it learns from past emotional experiences. However, the model does not create a dependency on the emotion mechanism for overall reasoning in the system. Even so, many of the benefits of emotions can be gleaned from this partial solution. An added advantage of this strategy is that this form of emotions can be easily added to existing systems to allow them to display appropriate emotions or reason about other agent’s emotions. It should be noted that this method does not attempt to model the more subtle forms of emotions described previously.

3.1 Pandemonium Theory

Both the CMattie and IDA projects utilize an architecture designed around John Jackson's Pandemonium Association Engine (1987). This architecture is based on a psychological theory called Pandemonium Theory (Selfridge 1959) which was used to describe human perception. Later, Jackson (1987) presented it to the computer science community in an extended and more concrete form (Franklin 1995) that makes it useful for control of autonomous agents.

In Jackson’s version of Pandemonium Theory, the analogy of an arena is used. The arena consists of stands, a playing field, and a sub-arena. It is also populated by a multitude of “codelets,” each a simple agent. Some of the codelets will be on the playing field doing whatever it is they are designed to do;

---

1 Jackson uses the term “demon” where we use “codelet,” a term borrowed from the Copycat architecture (Hofstadter and Mitchell 1994). We do this since not all of Jackson’s demons may be so in the customary sense of the word as used in computer science.
these codelets are considered “active.” The rest of the codelets are in the stands watching the playing field and waiting for something to happen that excites them. Of course, what is exciting may be different for each codelet. The more exciting the action on the field is to any particular codelet, the louder that codelet yells. If a codelet yells loudly enough, it gets to go down to the playing field and become active. At this point, it can perform its function. Its act may excite other codelets, who may become active and excite yet other codelets, etc.

Which codelets excite which other codelets is not a random matter; each codelet has associations with other codelets that act much like weighted links in a neural network. The activation level of a codelet (a measure of how loudly it is yelling) spreads down the codelet’s association links and, therefore, contributes to the activation level of the receiving codelet. In addition, these associations are not static. Whenever a codelet enters the playing field, the sub-arena creates associations (if they do not already exist) between the incoming codelet and any codelets already on the field. A strong output association and a weaker input association are created between the codelets currently on the playing field and the arriving codelet. The actual strength of the associations depends on a gain value that the sub-arena calculates. In the stadium metaphor, the sub-arena is where all of the processes are carried out that are needed for the basic function of the architecture. In addition to the calculation of the gain, an example of a process that the sub-arena would be responsible for is the act of actually passing activation down a link. The gain is intended to be an estimate of how well the whole system is doing at any given time. This will become important as we apply
emotions to the calculating of this value. In addition to creating these new associations, existing association strengths between codelets on the playing field increase (or decrease) at each time step based on the gain value. Also, multiple codelets that have strong associations with each other can be grouped together, to create a single new codelet called a concept codelet. From the moment of their creation onward, these concept codelets act almost like any other codelet in the system. They differ in that the decay rate of their associations is less, and the amount of time that they spend on the playing field at any one calling is increased.

The sub-arena performs the actual input and output functions of the system as well as most of the automatic maintenance functions. It calculates the gain, a single variable intended to convey how well the agent is performing. Jackson did not specify a mechanism for such an assessment. Surely the assessment must be both domain dependent and goal dependent. Since the gain determines how to strengthen or weaken associations between codelets, how this judgment is arrived at, and how the goal hierarchy is laid out is of considerable importance. The agent accomplishes goal directed behavior only by an accurate assessment of its moment to moment status. For humans there is a complex system of sensory labeling and emotional responses, tuned through evolution, which allows us to determine our performance, based on currently active goal contexts.

The current goal context of this system changes dynamically. It can be thought of as emerging from the codelets active at a given time. (How this happens will be described below.) Some high-level concept codelets can remain
on the playing field for quite a long time and, therefore, influence the actions of
the whole agent for that time. An example of such a high level codelet might be
one that tends to send activation to those codelets involved in getting some lunch.
Multiple goal contexts can be competing or cooperating to accomplish their tasks.

3.2 Conscious Mattie (CMattie)

The first model of emotions that will be detailed is being implemented in a
system called Conscious Mattie (CMattie). CMattie (Bogner 1998; Franklin
1997) is the next incarnation of Virtual Mattie (VMattie), an intelligent clerical
agent (Franklin et al 1996; Zhang et al 1998; Song and Franklin forthcoming).
VMattie’s task is to prepare and distribute announcements for weekly seminars
that occur throughout a semester in the Mathematical Sciences Department at the
University of Memphis. She communicates with seminar organizers and
announcement recipients via email in natural language, and maintains a list of
email addresses for both. VMattie is completely autonomous, actively requesting
information that has not been forthcoming, and deciding when to send the
announcements, reminders, and acknowledgements without human intervention.
No format has been prescribed for any type of incoming email message to her.
CMattie will occupy this same domain but will have a number of additions. For
one, the underlying architecture for CMattie will be a version of the
Pandemonium architecture of John Jackson (1987) which will be discussed latter.
Modules for metacognition, learning, associative memory, and consciousness will
be included to add further functionality to Pandemonium. One possible drawback
to this domain with respects to the emotion component is that it may not be rich enough to require the emergence of complex emotions.

CMattie is designed to model the global workspace theory of consciousness (Baars 1988, 1997). Baars’ processes correspond to codelets (or demons) from pandemonium theory, and are also called codelets in CMattie. All actions are taken at the codelet level. Baars postulates goal contexts that correspond to higher level behaviors in CMattie. Some action selection is made at the behavior level, and then implemented by the appropriate lower-level codelets. Emotion codelets influence not only other codelets but behaviors as well. For an overview of CMattie’s architecture see Figure 3.1.

CMattie also has an associative memory capability based on a sparse distributed memory mechanism (Kanerva 1988). A new perception associates with past experiences including actions and emotions. These remembered emotions activate emotion codelets that, in turn, influence current action selection. Thus, CMattie will produce actions, at least partially based on emotional content, and appropriate for the active goal contexts. This is quite in keeping with current research on human decision-making using emotions (Cytowic 1993; Damasio 1994).

What sorts of emotional reactions can be expected of CMattie? She may experience fear at an imminent shutdown message from the operating system. She may be annoyed at having reminded an organizer twice to send speaker-topic information with no reply. She may be pleased at having learned the new concept “colloquium.”
Consciousness

Metacognition

Sparse Distributed Memory

Focus

Emotion Mechanism

Emotion Suggested Actions

Perception Registers

Perceptual Working Memory

Slipnet

Mail Input and Output

Case-Based Memory

Behavior Network

Working Memory

Tracking Memory

Drives

Key:

→ Solid arrow signifies regular data transfer.

↔ Dotted arrow signifies potential activation of target can occur with data transfer.

● Filled circle indicates modules where spotlight can shine.

Figure 3.1: CMattie’s architecture (Bogner, Ramamurthy, and Franklin 1998)
What effects might these have? Fear of an impending system shutdown may result in CMattie’s ignoring another message being processed in her perceptual workspace in favor of quickly saving files. Annoyance may result in a more sharply worded reminder. Pleasure may reinforce her learning activities. Increasing activation or association between behaviors or codelets, or some combination of the two will bring about all of these influences.

Will CMattie be aware of her emotions? Yes and no. The yes answer results from past emotions appearing in one part of the focus and of the present emotion in another part. Consciousness routinely shines its spotlight on the focus. Hence, CMattie “experiences” these emotions. The question could also be asked if CMattie would be aware of what emotion she was “experiencing.” Put another way, does the spotlight of consciousness ever shine on the emotion mechanism? Here the answer is no, not because it would be difficult to make it happen, but because we’ve found no justification for doing so. However, consciousness may shine on a particular emotion codelet that happens to have been included in a coalition of codelets. This can occur if the emotion codelet has been repeatedly triggered at approximately the same time as the other codelets in the coalition and, therefore, via Pandemonium, acquiring high associations with these codelets. This does not give CMattie an awareness of her overall emotional state explicitly but does give her an awareness of what is likely to be the dominant emotional trigger at that moment.
3.3 An Intelligent Distribution Agent (IDA)

IDA is an Intelligent Distribution Agent for the U.S. Navy. 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 200 people, called detailers, full time to effect these new assignments. IDA’s task is to facilitate this process, by playing the role of detailer as best she can. Designing IDA presents both communication problems and constraint satisfaction problems. She must communicate with sailors via email and in natural language, understanding the content. She must access a number of databases, again understanding the content. She must see that the Navy’s needs are satisfied, for example, the required number of sonar technicians on a destroyer with the required types of training. She must hold down moving costs. And, she must cater to the needs and desires of the sailor as well as is possible.

Here we’ll briefly describe a design for IDA including a high level architecture and the mechanisms by which it’s to be implemented. With the help of diagrams we’ll describe a preconscious version of IDA, and then discuss the additional mechanisms needed to render her conscious.

IDA will sense her world using three different sensory modalities. She will receive email messages, read database screens and, eventually, sense operating system commands and messages. Each sensory mode will require at least one knowledge base and a workspace. The mechanism here will be based loosely on the Copycat Architecture (Hofstadter and Mitchell 1994; Zhang et al 1998). Each knowledge base will be a slipnet, a fluid semantic net. The workspace (working
memory) will allow perception (comprehension), a constructive process. See the right side of Figure 3.2 for five such pairs. Each, other than the email, will understand material from a particular database, for example personnel records, a list of job openings, a list of sailors to be assigned. Sensing the operating system isn’t present in Preconscious IDA.

Note that each of IDA’s senses is an active sense, like our vision rather than our hearing. They require actions on IDA’s part before sensing can take place, for example reading email or accessing a database. One component of IDA’s action selection is an enhanced version of a behavior net (Maes 1990a; Song and Franklin forthcoming). See Figure 3.2. The behavior net is a directed graph with behaviors as vertices and three different kinds of links along which activation spreads. Activation originates from internal, explicitly represented drives, from IDA’s understanding of the external word through the Focus, and from internal states. The behavior whose activation is above some threshold value and is the highest among those with all prerequisites satisfied becomes the next goal context as specified in global workspace theory. The several small actions typically needed to complete a behavior are taken by codelets, which will be discussed in more detail later. IDA’s behaviors are partitioned into streams, the connected components of the digraph, each in the service of one or more drives. Streams of behaviors are like plans, except that they may not be linear, and might well be interrupted during their execution or possibly not completed. Examples of IDA’s streams include Access EAIS, Access Personnel Record, Send Acknowledgement, Offer Assignments, Produce Orders.
Preconscious IDA Architecture

Figure 3.2: IDA’s architecture (Franklin, Kelemen, and McCauley 1998)

IDA is very much a multi-agent system, the agents being the codelets that underlie all the higher level constructs and that ultimately perform all of IDA’s actions. We’ve mentioned the codelets that underlie behaviors. Others underlie slipnet nodes and perform actions necessary for constructing IDA’s understanding of an email message or of a database screen (Zhang et al 1998). Still other codelets will play a vital role in consciousness, as we’ll see below. The codelets
are represented in Figure 3.2 by a long box at the bottom, since they underlie essentially everything else.

Having gathered all relevant information, IDA must somehow select which assignments she’ll offer a given sailor. See the lower left of Figure 3.2. Being a constraint satisfaction problem, considerable knowledge will be required to make these selections. This knowledge could be in the form of a traditional, rule-based expert system, but more likely will be implemented as a classifier system (Holland 1986) in order to facilitate learning at a later stage. The constraint satisfaction mechanism will be housed in the selection module. The choice of mechanism for it is currently being researched.

IDA’s emotion module (McCauley and Franklin 1998), like a human’s, provides a multi-dimensional method for ascertaining how well she’s doing. We will experiment with building in mechanisms for emotions, such as anxiety at not understanding a message, guilt at not responding to a sailor in a timely fashion, and annoyance at an unreasonable request from a sailor. Emotions in humans and in IDA influence all decisions as to action (Damasio 1994). IDA’s action selection will be influenced by emotions via their effect on drives. Including emotional capabilities in non-biological autonomous agents is not a new idea (Bates, Loyall, and Reilly 1991; Picard 1997; Sloman and Poli 1996).

As a glance at Figure 3.2 shows, IDA has a number of different memories. The Offer Memory is a traditional database that keeps track of the assignments IDA has offered various sailors. The template memory is another that holds the various templates that IDA uses to compose commands to access databases or
issue orders, and to compose messages to sailors. IDA’s intermediate term
memory acts as an episodic memory, providing context for email messages and
for the contents of database screens. It’ll be implemented as a case-based memory
to facilitate case-based learning at a later stage. IDA’s associative memory does
what you’d expect. It associates memories, emotions and actions with incoming
percepts. It’s implemented by an extension of sparse distributed memory
(Kanerva 1988).

The operation of these last two, more complex, memory systems deserves
more explanation. As IDA’s most recent percept reaches the perception register
(See Figure 3.3) having been constructed (comprehended) by one of the
perception modules, several events occur in simulated parallel. The associative
memory is read using the percept as the cue. Since sparse distributed memory is
content addressable, associations with the percept, including an emotional
overtone and an action previously taken in a similar situation are typically
returned into an expanded copy of the perception registers (see Figure 3.3). At
the same time intermediate term memory is read with the same cue. The most
similar case is returned, again with emotion and action, into yet another copy of
the expanded perception registers. In the full version consciousness will come into
play at this point. Now, an action and an emotion are selected into the two
remaining copies of the expanded perception registers along with the current
percept. Each is then written to its appropriate memory. IDA has then processed a
single percept.
<table>
<thead>
<tr>
<th>Emotion</th>
<th>Perception Register</th>
<th>Action</th>
<th>From Intermediate Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emotion</td>
<td>Perception Register</td>
<td>Action</td>
<td>From Associative</td>
</tr>
<tr>
<td></td>
<td>Perception Register</td>
<td></td>
<td>Most Recent</td>
</tr>
<tr>
<td>Emotion</td>
<td>Perception Register</td>
<td>Action</td>
<td>To Associative</td>
</tr>
<tr>
<td>Emotion</td>
<td>Perception Register</td>
<td>Action</td>
<td>To Intermediate Term</td>
</tr>
</tbody>
</table>

**Figure 3.3**: Perception Bank in CMattie (Bogner, Ramamurthy, and Franklin 1998)

Global workspace theory postulates the contents of consciousness to be coalitions of codelets shined on by a spotlight. Imagine a codelet workspace populated by many active codelets each working, unconsciously and in parallel, on its own agenda. The spotlight seeks out coalitions of codelets that arise from novel or problematic situations. When the spotlight picks out some coalition of codelets, the information contained therein is broadcast to all the codelets, active or not. The idea is to recruit resources that are relevant codelets to help in dealing with the situation. It seems that in humans almost any resource may be relevant depending on the situation. The global workspace method attacks the problem of finding the relevant resources by brute force. Broadcast to them all. IDA will use this method. To do so, she’ll need a coalition manager, a spotlight controller, and a broadcast manager (Bogner, Ramamurthy, and Franklin to appear).
3.4 Emotions in CMattie and IDA

One of the key components of Jackson’s system is the gain. It is the gain that determines how link strengths are updated and, consequently, how well the agent pursues its intended goals. Therefore, it is of great importance to understand how the value of the gain is calculated at any given time, and how that value is used. One might view gain as a one-dimensional “temperature” as in the Copycat Architecture (Hofstadter and Mitchell 1994). The introduction of emotions into an agent architecture allows for a more sophisticated assessment of the desirability of the current situation or perception.

3.4.1 Theory

A major issue in the design of connectionist models has been how systems can learn over time without constant supervision by either a human or some other external system. Ackley and Littman solve this problem for their artificial life agents by having those agents inherit an evaluation network that provides reinforcement so that its action selection network can learn (1992). In humans, emotions seem to play the role of the evaluation network. As well as affecting our choice of actions, they evaluate the results of these actions so that we may learn. Including emotions in an agent architecture could serve this same purpose.

This dilemma is solved by the addition of emotion codelets whose resulting action is the updating of the gain value. The gain in these architectures is not a single value like Copycat's temperature; instead, it is a vector of four real numbers that can be thought of as analogous to the four basic emotions, anger, sadness, happiness, and fear. It is possible that two more elements could be added
representing disgust and surprise (Ekman 1992; Izard 1993), although, for current purposes the four emotions mentioned should suffice. CMattie's domain is narrow enough so that surprise and disgust would not be of great benefit. This may not be the case for IDA who will probably need disgust and surprise added to her repertoire. The agent's emotional state at any one time is, therefore, considered to be the combination of the four (or six) emotions. A particular emotion may have an extremely high value as compared to the other emotions, and, consequently, dominate the agent's overall emotional state. For example, if a train has blocked the route to your favorite restaurant and you are hungry and in a hurry, your emotional state may be dominated by anger even though many other more subtle emotions may be active at the time. The same type of thing can occur in CMattie and IDA although in a more limited domain. In such a case the agent can be said to be angry. It is important to note, however, that the agent will always have some emotional state whether it is an easily definable one such as anger or a less definable aggregation of emotions. No combination of emotions are preprogrammed; therefore, any recognizable complex emotions that occur will be emergent.

The value of an individual element (emotion) in the gain can be modified when an emotion codelet fires. Emotion codelets are a subset of primitive codelets and, therefore, have preconditions based on the particular state or perception the codelet is designed to recognize. When an emotion codelet’s preconditions are met it fires, modifying the value of a global variable representing the portion of the emotion vector associated with the codelet’s
preconditions. A two step process determines the actual value of an emotion at any one time. First, the initial intensity of the emotion codelet is adjusted to include valence, saturation, and repetition via the formula

\[
a = \nu \frac{1}{1 + e^{\frac{-\nu x + x_0}{1.5}}}
\]

where

- \( a \) = adjusted intensity at creation time
- \( x \) = the initial intensity of the emotion
- \( \nu \) = the valence \( \{1,-1\} \)
- \( x_0 \) = habituation, it shifts the function to the left or right

The \( x_0 \) parameter will have its value increased when the same stimulus is received repeatedly within a short period of time. The effect of \( x_0 \) is the modeling of the short-term habituation of repeated emotional stimuli.

The second step in the process is that each emotion codelet that has fired creates an instantiation of itself with the current value for adjusted intensity and a time stamp. This new instantiated emotion codelet is placed on the playing field where it can perform its actions. This new codelet will add its adjusted intensity value (not to be confused with activation) to the global variable representing its particular emotion based on the formula (modified from Picard 1997)

\[
y = ae^{-b(t-t_0)}
\]

where

- \( a \) = adjusted intensity at creation time
- \( b \) = decay rate of the emotion
- \( t \) = current time
- \( t_0 \) = time at creation of the codelet
Remember that the emotion vector is not a single value, so a single codelet will only effect one part of the vector, anger, for instance. The overall anger value for the agent would, therefore, be a summation of all of the \( y \) values for codelets that fire and that effect the anger portion of the emotion vector. When \( y \) approaches zero the codelet will stop effecting the emotion vector. As stated previously, the emotional state of the agent at any given time is stored in long term memory. During the recall process these remembered emotions re-effect the emotional state of the agent by instantiating a new codelet in much the same way as an original emotion codelet would. In such a circumstance, the codelet will affect the emotional state of the agent using the previous formula adjusted for the new time of activation and with a degraded initial intensity.

There can be multiple emotion codelets, each with its own pattern that can cause it to fire. The system is not limited to the firing of only one emotion codelet at any one time. The resulting emotional state of the agent, represented by the gain vector, is, therefore, a combination of the recent firings of various emotion codelets. Also, multiple emotion codelets can be included in concept codelets, thereby learning complex emotions that are associated with a higher level concept.

### 3.4.2 Code Description

It was important in the design of the actual code for an emotion mechanism to make it as portable and easy to integrate as possible. To this end, this implementation was done in Java 1.1 conforming to the Java Bean
specification. Generally, this means that the objects derived from these classes can be used within a visual tool to design applications with relative ease. The objects will also be fully serializable, which means that the emotional state of the agent as well as the state of any individual codelets can be easily saved for complete restoration at some later time. Java Bean compatibility also means that the objects communicate via events rather than needing to know the exact methods or properties needed to determine whether a particular situation has occurred in some other foreign object. Finally, a Java Bean object can be used as an ActiveX component through the ActiveX Bridge for Java. This will allow the objects described below to be used in any application that is ActiveX compatible such as Visual Basic, C++, Delphi, and many more. The full javadoc generated documentation for the code is provided in Appendix A.

There were several new classes of objects created for this project. The first, and probably most central class created, was the EmotionVector class. This object serves mainly as a repository or central location where all changes to the emotional state of the agent are registered. This class contains "get" methods to allow other objects to determine the emotional state of the agent and "set" methods to allow programmers to configure the initial state and properties of emotions at design time. The EmotionVector receives EmotionEventObjects generated by the EmotionCodelets. The EmotionEventObject class is an extension of the EventObject class with additional methods for retrieving the name of the emotion to be effected and the change that should be applied to that emotion.
The class that will do most of the work is the EmotionCodelet class. This is actually an abstract class that is meant for extension rather than direct instantiation. The abstract class, however, supplies all the code necessary for an object to function as an EmotionCodelet except for the DetectPattern() method. The DetectPattern() method is responsible for the actual determination of whether a specific pattern exists or not. If the pattern does exist, then this method sets the appropriate properties needed to fire the codelet and effect the EmotionVector. An example EmotionCodelet has been provided, called EventEmotionCodelet, that responds to any type of EventObject. This allows the EventEmotionCodelet to be attached to any object that produces an EventObject. A simple example might be the attaching of a button, which produces an EventObject when pressed, to an EventEmotionCodelet. Then, whenever the button is pressed, the EventEmotionCodelet receives the event, calculates its emotional response, and produces an EmotionEventObject. The EmotionEventObject is then caught by the EmotionVector where the value of the EmotionEventObject is added to the overall emotional state of the agent. The EventEmotionCodelet is meant as a general purpose EmotionCodelet that can be used in many different ways and applications. In addition it could be extended to listen for specific types of events so as to respond in some special way.

3.4.3 Implementation Demonstration

As a way of demonstrating the emotional mechanism in action, a toy problem was implemented that used emotional perceptions as the action selection mechanism almost exclusively. The system created for this demonstration was an
adaptation of the classic Wumpus World game. The game consists of caves represented as a grid, usually no bigger than four-by-four, in which are placed at random a wumpus, a pile of gold, and a number of pits (Figure 3.4). There is also a hunter in the game that, in the classic version, is always placed at the bottom left-hand corner of the grid at the beginning of the game. The object is for the hunter to get the pile of gold and get back out by navigating through the grid without falling into any pits or getting eaten by the wumpus. Getting back out of the cave means returning to the square where the hunter started the game and climbing out. There is very little light in the cave so the hunter has very limited perceptions of his surroundings. The hunter’s perceptions are as follows, (a) he can feel a breeze if he is in a grid square that is adjacent (North, South, East, or West of his current position, no diagonal sensing) to a pit; (b) he can smell a stench when he is in a grid square that is adjacent to the wumpus; (c) he sees a faint glimmer when he is in a grid square that is adjacent to the gold; (d) he can see the gold when he is in the same grid square as the gold. The hunter’s actions are similarly limited, they include (a) moving in one of the four cardinal directions, (b) grabbing the gold, (c) climbing out of the cave, and (d) firing his single arrow in one of the four directions. The use of this last action deserves some additional explanation. If the hunter can determine the grid location of the wumpus, and the hunter is directly North, South, East, or West of that location, then the hunter can fire his arrow in an attempt to kill the wumpus and, thereby, eliminating one of his obstacles. The key here is that the hunter only gets one shot per game and must, therefore, be quite judicious in its use.
Figure 3.4: Typical Wumpus World screen.

The challenge was to create a hunter agent that used emotional perceptions almost exclusively for action selection. The first step in this process was to create PerceptEmotionCodelets that fired when one of the basic perceptions occurred. For instance, when a breeze was perceived by the hunter, a PerceptEmotionCodelet fired that caused some level of fear to be added to the current emotional state of the hunter. Similarly, the other EmotionCodelets were created to respond to the other possible perceptions as well as internal states of the agent. Then a series of rules were designed that used the agent’s current emotional state to generate expected emotional states for all of the possible actions that the hunter could perform. The hunter was given a simple memory
that allowed him to remember where he had been and what his emotional state was in each grid square that he had visited. Finally, a simple function was created that chose which action to perform at any given time based on the expected resulting emotional state.

Using only this emotional information, the hunter performed quite well. Even if a solution was possible, the hunter was not guaranteed to find the gold and get back out. However, the hunter did consistently find the gold, avoid the pits and wumpuses, and get out of the cave. In cases where there was no solution or the possibility of a solution was unclear (see Figure 3.5), the hunter would usually correctly assess the situation and leave the cave without retrieving the gold. It is important to note that there was no plan generation and no explicit goal represented. Even so, the hunter's performance was comparable to first order logic in solving this problem.

For most versions of the Wumpus World, first order logic can adequately solve any given configuration (Russell and Norvig 1996). What is interesting about our emotional mechanism as compared to first order logic is that, without adding rules or any other mechanism, our hunter can handle any size grid, any starting location, any number of pits or wumpi, and configurations where the wumpuses or wumpi are allowed to move. For first order logic to handle all of these conditions requires the addition of non-monotonic reasoning and a great deal of rules.

As it is with humans, emotions in the hunter provide a way to make decisions that are robust given incomplete information in a dynamic environment.
Figure 3.5: A situation where the likelihood of danger is high. The hunter will, most likely, climb out of the cave without proceeding.

The mechanism is non-computationally intensive and easy to utilize in any number of different algorithms.

A natural and relatively simple extension of the demonstration agent would be to allow the hunter to learn what emotional states to expect given the current emotional state. Over several trials, for example, the hunter would learn that going into an unknown (not previously visited) grid square when his fear is high (either from feeling a breeze or smelling a stench) could likely result in his death - which doesn't feel so good.
CHAPTER 4

A NEW PERSPECTIVE ON LEARNING IN ARTIFICIAL SYSTEMS

It is important to note how this emotional mechanism changes the way that learning takes place in an artificial system. In most systems there is a desired output for every input vector. With any architecture that is designed with emotion, however, there is only desired input. An agent based on this type of architecture must be situated within an environment and, by its actions, be able to change its environment in a way that it can sense the change (Franklin and Graesser 1997). What this means for learning is that such an agent should be choosing its actions in such a way as to manipulate its environment so that the agent maximized pleasure and minimizes displeasure. This is different from the classic reinforcement scheme used in many neural networks (Watkins 1989) where a simple positive or negative valence is returned to the system by the environment after an output is produced. Some methods have been designed to allow connectionist systems to accomplish unsupervised learning via clustering algorithms. These algorithms allow a system to categorize input without the need for external evaluation, but still lack any way of associating which categories represent states that the agent should avoid or maintain. The system described here, which might be labeled Unsupervised Internal Reinforcement, uses the set
of internal emotion codelets to recognize pleasurable and non-pleasurable states of the environment.

Why is this method an advantage over standard reinforcement? For one, the judgement as to whether an output/action is correct is not dependent on some external judge. From the agent’s point of view reinforcement, by its definition, can never be unsupervised because the agent is always dependent on this external evaluation. Secondly, in a reinforcement scheme a given output $b$ for a given input $a$ will always elicit the same reinforcement value. This method only allows the agent to react to its input and does not provide for the fact that an autonomous agent's internal states play a crucial role in determining the most rational action for any particular situation. A given input could elicit a different response from the agent based on internal states that cannot be known to any external evaluation system. For instance, the agent might learn that a particular object is not as threatening as it previously thought. It might even learn that the object is beneficial if treated correctly. A good example of this point for humans would be the use of fire; if not treated appropriately a person might well get burned. Finally, standard reinforcement does not provide a good mechanism to allow an agent to generalize and act appropriately in unpredictable or dynamic environments. Unsupervised Internal Reinforcement encourages the agent to manipulate its environment over time to maximize positive valence – pleasure – and minimize negative valence - displeasure - in such a way that it can learn to extrapolate and arrive at good enough solutions for unforeseen situations.
A question arises as a result of this formulation of learning. Why is it that humans do not seek pleasure to the exclusion of everything else? We do not have sex with every member of the opposite gender that we meet. We do not all use pleasure inducing drugs. We do not seem to be trying to maximize pleasure. The answer to this issue is that human are maximizing pleasure. As previously stated, emotion is not a one-dimensional thing; there are multiple emotions and multiple drives that those emotions pursue to varying degrees over time. This complex interplay makes the very definitions of pleasure and displeasure hard to nail down. Put another way, pleasure does not necessarily mean being in a state of constant bliss. Humans are situated in their environment and have bodies. Evolution has designed the human body in such a way as to turn pleasure into pain if it is experienced at too high a level or for too long a time. Therefore, the optimal solution for the maximization/minimization problem is not the maintaining of a high pleasure level, it is the maintaining of something close to a balance, an air of contentedness. The maximization of only one type of pleasure without taking into account the myriad of other drives would result in a highly dysfunctional agent. This type of dysfunction, to some degree, is quite common among humans; the number of people diagnosed with some sort of addiction is testament to this fact.

The real question for learning in artificial systems has, therefore, become one of how best to maximize pleasure and minimize displeasure at any one moment as opposed to minimizing the error. It seems fairly obvious that a minimization of error scheme is only useful for omniscient agents whose
environment can be completely known either by the agent or by some external evaluation system. Not only must the entire environment be known, but, in order for an error to be calculated, the correct response for every possible state must also be known. For situated agents in more complex and dynamic environments, however, emotions serve as a heuristic that allows the agent to react to its changing situation in a quick and rational manner. In essence, emotions provide a “good enough” response to an environmental state.

There has been a great deal of research that indicates that, for humans, emotion is one of, if not the, key element that brings about “rational” behavior (Adolphs 1996; Cytowic 1993; Damasio 1994). The definition of “rational” behavior is important for situated agents. Rational behavior is that behavior that avoids non-pleasurable states and/or pursues pleasurable states. As mentioned previously, emotions for humans have been adjusted and prewired over millions of years of evolution. Even so, many of the decisions that humans make in the course of our daily lives are based on our culture and on those complex learned emotions that are not prewired. How humans manage to learn these complex emotions and how these become coupled to actions is of paramount importance. A promising model is described by Juan Velásquez that is similar to Marvin Minsky’s K-lines implementation (Minsky 1986; Velásques 1998). Velásquez’s model involves associating an emotion to the particular sensory input that it evokes. This association then acts much like an inhibitory or excitatory behavior increasing or decreasing the likelihood that a particular action is chosen. The
system can, therefore, have multiple excitatory and/or inhibitory links to any particular action.

For our current implementation, the emotion codelets will effect the drives of the system, which will, in turn, effect the behavior net (Franklin 1997). However, future work will attempt to determine if complex behaviors can be emergent without the use of explicit drive and goal generation modules.
CHAPTER 5

CONCLUSION

It is clear that emotions are an integral part of what we call intelligence. Evidence shows that humans rely on the emotional mechanisms of the brain to such an extent that people with malfunctions of brain areas that facilitate emotional markers fail to display completely rational behavior. CMattie and IDA will include a model of emotion that displays a broad range of both intrinsic and learned emotions in a manner that is easily applied to any number of domains. For instance, a robot placed on a remote planet could “feel” a sense of accomplishment at reaching a destination or relief when its batteries are recharged after being low. Of course for such a simple case, these same behaviors can be programmed into a rule based system with good results. However, when the sensory input becomes more complex, such as the differentiation of various elements by their color or shape, then an emotional mechanism can accomplish the given task with a minimal amount of computation and allow the agent to learn from its own experience. For the planetary robot, this might mean learning that a particular type of rock formation, that was not known about during the original programming of the robot represents a dangerous area and should be avoided.

Learning for a system with an emotional component or, better yet, integrated emotions must be thought of in a different light than has been common for artificial intelligence researchers to date. Emotions can take the place of the
evaluation function or the external observer. This means that some new questions must be asked when the agent is being designed. What are the motivations for this agent? Why would it want to accomplish the tasks we set for it? What things in its environment represent the good and bad elements from the agent’s perspective? Correct answers to these questions form the foundation for the emotional mechanism of the agent. From this foundation an agent can learn the complex relationships that give it pleasure, pain, or stability in any environment.

Emotions are more than the strong feelings associated with things and events. They are also the elements of moment to moment life that provide us with the feelings of existence. The method detailed here will provide a way for many of the multi-agent systems that are being developed today to easily add emotional cognition to their repertoire of abilities. What remains is the full integration of emotions into a complete model of cognition. Future work will focus on this goal: the design and implementation of an architecture that displays emotion as an inseparable component with logic and perception.
BIBLIOGRAPHY


Bogner, Myles, Uma Ramamurthy, and Stan Franklin (to appear), Modeling Global Workspace Perception In A Socially Situated Agent, in Kerstin Dautenhahn ed. Human Cognition and Social Agent Technology.


Lawrence Erlbaum Associates, East Sussex, U.K.


Rockefeller University Press, New York.


APPENDIX A
APPENDIX A
Javadoc generated code documentation

Class Emotion.EmotionVector

java.lang.Object
    +----Emotion.EmotionVector

public class EmotionVector
extends Object
implements Serializable

Constructor Index
- EmotionVector()

Method Index
- EmotionVector()
  basic constructor.
- EmotionVector(float, float, float, float)
  constructor that sets the initial values for the basic emotions.
- getAnger()
  returns a floating point number that is the anger level.
- getFear()
  returns a floating point number that is the fear level.
- getHappiness()
  returns a floating point number that is the happiness level.
getSadness()

returns a floating point number that is the sadness level.

setAnger(float)

setFear(float)

setHappiness(float)

setSadness(float)

UpdateAnger(float)

updateEmotion(EmotionEventObject)

UpdateFear(float)

UpdateHappiness(float)

UpdateSadness(float)

**Constructors**

public EmotionVector()

**Methods**

public void EmotionVector(float iHappiness,
                          float iSadness,
                          float iAnger,
                          float iFear)

constructor that sets the initial values for the basic emotions.

**Parameters:**
iHappiness - initial happiness
iSadness - initial sadness
iAnger - initial anger
iFear - initial fear

public void EmotionVector()
basic constructor.

```
public void setHappiness(float h)
```

UpdateHappiness

```
public synchronized void UpdateHappiness(float h)
```

setSadness

```
public void setSadness(float s)
```

UpdateSadness

```
public synchronized void UpdateSadness(float s)
```

setAnger

```
public void setAnger(float a)
```

UpdateAnger

```
public synchronized void UpdateAnger(float a)
```

setFear

```
public void setFear(float f)
```

UpdateFear

```
public synchronized void UpdateFear(float f)
```

getHappiness

```
public float getHappiness()
```

returns a floating point number that is the happiness level.

Returns:

s float

getSadness

```
public float getSadness()
```

returns a floating point number that is the sadness level.

Returns:

s float

getAnger

```
public float getAnger()
```

returns a floating point number that is the anger level.

Returns:

s float

getFear
public float getFear()
returns a floating point number that is the fear level.

Returns:
s float

public synchronized void updateEmotion(EmotionEventObject e)

Class
Emotion.EmotionCodelets.EmotionCodelet

java.lang.Object
   +----Emotion.EmotionCodelets.EmotionCodelet

public abstract class EmotionCodelet
extends Object
implements Runnable, Serializable

Variable Index
   • activationDecayRate
   • activationLevel
   • AssociationList
   • id
**InfoToBeBroadcast**

**name**

**Constructor Index**

- `EmotionCodelet()`  constructor.

**Method Index**

- `addEmotionListener(EmotionListener)`
- `getDecay()`
- `getEmotionAffected()`
- `getHabituation()`
- `getHabituationDecay()`
- `getHabituationThreshold()`
- `getId()`
- `getName()`
- `getValence()`
- `initialize()`
- `LeaveField()`  destructor.
- `removeEmotionListener(EmotionListener)`
- `run()`  decides when to try to detect the pattern and when to fire the codelet.
- `setDecay(float)`
- `setEmotionAffected(String)`
- `setHabituationDecay(float)`
- `setHabituationThreshold(float)`
- `setId(String)`
- `setName(String)`
- `setValence(int)`

**Variables**

- `name`
  
  public String name
- `id`
  
  public String id
- `AssociationList`
  
  public Vector AssociationList
- `activationLevel`
  
  public float activationLevel
- `activationDecayRate`
  
  public float activationDecayRate
- `InfoToBeBroadcast`
  
  public HasTable InfoToBeBroadcast

**Constructors**

- `EmotionCodelet`
  
  public EmotionCodelet()
      constructor. Adds codelet to the playing field.

**Methods**

- `getHabituation`
  
  public float getHabituation()
- `getHabituationDecay`
  
  public float getHabituationDecay()
- `setHabituationDecay`
  
  public void setHabituationDecay(float h)
- `getHabituationThreshold`
  
  public float getHabituationThreshold()
- `setHabituationThreshold`
public void setHabituationThreshold(float t)

setValence

public void setValence(int v)

setDecay

public void setDecay(float d)

setEmotionAffected

public void setEmotionAffected(String e)

getValence

public int getValence()

getDecay

public float getDecay()

getEmotionAffected

public String getEmotionAffected()

addEmotionListener

public void addEmotionListener(EmotionListener l)

removeEmotionListener

public void removeEmotionListener(EmotionListener l)

setName

public void setName(String n)

getName

public String getName()

getId

public String getId()

setId

public void setId(String n)

initialize

public void initialize()

LeaveField

public void LeaveField()
	destructor. Removes codelet from the playing field.

run

public void run()
	decides when to try to detect the pattern and when to fire the codelet.
Class
Emotion.EmotionCodelets.EventEmotionCodelet

java.lang.Object
    +----Emotion.EmotionCodelets.EmotionCodelet
        +----Emotion.EmotionCodelets.EventEmotionCodelet

class EventEmotionCodelet
extends EmotionCodelet
implements Runnable, Serializable

Constructor Index
EventEmotionCodelet()

Constructors
EventEmotionCodelet()

public EventEmotionCodelet()
Class Emotion.EmotionEventObject

java.lang.Object
    \|-- java.util.EventObject
        \|-- Emotion.EmotionEventObject

public class EmotionEventObject
    extends EventObject

Constructor Index

EmotionEventObject(String, float, Object)

Method Index

- getName()
- getValue()

Constructors

EmotionEventObject

public EmotionEventObject(String Name,
                           float Value,
                           Object o)

Methods

getValue

public float getValue()

getName

public String getName()
**Interface Emotion.EmotionListener**

```java
public interface EmotionListener
    extends EventListener
```

---

**Method Index**

*changeEmotion*(EmotionEventObject)

---

**Methods**

*changeEmotion*

```java
public abstract void changeEmotion(EmotionEventObject e)
```