Believable Social and Emotional Agents

W. Scott Neal Reilly

May 1996
CMU-CS-96-138

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Thesis Committee:

Joseph Bates, Chair
Jaime Carbonell
Reid Simmons
Aaron Sloman, University of Birmingham, England

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was partially supported by Fujitsu Laboratories, Mitsubishi Electric Research Labs, and Justsystem Corporation. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of Fujitsu Laboratories, Mitsubishi Electric Research Labs, or Justsystem Corporation.

Keywords: Artificial Intelligence, Interactive Art, Interactive Drama, Interactive Entertainment, Interactive Fiction, Believable Agents, Emotional Agents, Social Agents, Interactive Characters, Oz, Tok, Em, Agent Architectures, Autonomous Agents, Social Behaviors, Agent Modeling, Personality

Believable Social and Emotional Agents

W. Scott Neal Reilly
wsr+@cs.cmu.edu

Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

One of the key steps in creating quality interactive drama is the ability to create quality interactive characters (or believable agents). Two important aspects of such characters will be that they appear emotional and that they can engage in social interactions. My basic approach to these problems has been to use a broad agent architecture and minimal amounts of modeling of other agent in the environment. This approach is based on an understanding of the artistic nature of the problem.

To enable agent-builders (artists) to create emotional agents, I provide a general framework for building emotional agents, default emotion-processing rules, and discussion about how to create quality, emotional characters. My framework gets a lot of its power from being part of a broad agent architecture. The concept is simple: the agent will be emotionally richer if there are more things to have emotions about and more ways to express them. This reliance on breadth has also meant that I have been able to create simple emotion models that rely on perception and motivation instead of deep modeling of other agents and complex cognitive processing.

To enable agent builders to create social behaviors for believable agents, I have designed a methodology that provides heuristics for incorporating personality into social behaviors and suggests how to model other agents in the environment. I propose an approach to modeling other agents that calls for limiting the amount of modeling of other agents to that which is sufficient to create the desired behavior. Using this technique, I have been able to build robust social behaviors that use surprisingly little representation. I have used this methodology to build a number of social behaviors, like negotiation and making friends.

I have built three simulations containing seven agents to drive and test this work. I have also conducted user studies to demonstrate that these agents appear to be emotional and can engage in non-trivial social interactions while also being good characters with distinct personalities.

Acknowledgements

I would like the thank my advisors, Joe and Jaime. When you get to CMU as a first year graduate student, you're told that there are basically two kinds of advisors: the younger advisor with a lot of time and enthusiasm—maybe too much time and enthusiasm—and the more experienced advisor that you don't see much but that is tremendously helpful when you do catch him or her in the country. I got the best of both worlds. Joe's vision and enthusism were contagious from the start and Jaime has been able to see to the heart of what I've been doing and make useful suggestions better than anyone I have yet to meet outside the Oz group.

I would like the thank my other committee members, Reid and Aaron, who have helped make my work more rigorous and scientific when I would have preferred to take an easier path.

I would like to thank Sara Kiesler of CMU's Department of Social and Decision Science who helped me put together the experiments that I describe.

I would like to thank all the members of the Oz Project that have made this work so much fun, including Bryan, Peter, Phoebe, Mark, Matt and, of course, Joe.

I would like to thank all of the people who have made my stay at CMU exciting, stimulating, and a whole lot of fun, including Sharon, Catherine, and a group of friends that I would love to list, but I'm too afraid of leaving someone off. Besides, this thesis is long enough as it is.

I would like to thank Phoebe, LeAnn, and Bob who read this monster and helped make it much more readable and comprehensible. You should probably thank them too.

Finally, I would like to thank my parents for supporting me and believing in me all these years. I would also like to thank LeAnn who gives meaning to my life and to my work.

Table of Contents

CHAPTER 1 Introduction

1.1 Imagine…

Imagine you could enter into the world of Indiana Jones—that you could be the world-famous archaeologist looking for lost treasure in exotic parts of the world, meeting interesting people and clashing with treacherous villains.

Imagine you could play the part of Hercule Poirot or Miss Marple trying to solve an intriguing and dangerous murder mystery.

Imagine you could be Sir Galahad on the quest for the Holy Grail. Or a hardened police sergeant trying to rescue a hostage. Or a space explorer meeting new intelligent civilizations.

Why can't you do these things? These are the kinds of things that many computer and video games claim to offer but all of them seem to fall short of their promises.

The problem, in part, seems to be that computer and video games have attempted to succeed purely on their interactive nature. Interactivity is certainly a powerful tool (see Sloan [Sloan91] and Kelso et al. [Kelso92]), but is it enough?

I believe the answer is no.

What is missing in current computer and video games? It seems to be the same two things that have made good novels stand out from bad novels, good movies from bad movies, and good drama from bad drama. These elements are *plot* and *character*.

The Oz Project at Carnegie Mellon has been developing technology that we hope will make it possible for artists to create simulated worlds that contain rich characters and that give a human interactor the feeling of being an important part of an interesting story. We hope the interactor can "suspend his or her disbelief" and become deeply engaged by the experience. This happens when people become deeply engaged by a movie or novel and we hope to be able to achieve a similar experience. In fact, we hope that the experience could be even more intense than that provided by a good movie because it is interactive. We call such experiences *interactive drama*.

There are (at least) two main problems in creating such a system. The first is how to create simulated worlds where the user has the feeling of freedom but where the user also has some artist-shaped experience. This problem is being studied by Wehyrauch [Wehyrauch96] and will not be central to the work described here.

The second main problem with creating interactive story worlds is how to build characters for such worlds that are as rich as characters in other media (e.g., movies, novels) while also being interactive. These interactive characters are also called *believable agents*.

Artificial Intelligence (AI) has developed a number of tools for building interactive agents that provide a good starting point, but they are not sufficient. For example, AI has not traditionally been concerned with creating agents that are emotional. Traditional AI has also focused on multi-agent interactions for the purposes of problem solving but has not investigated how to create agents that interact with each other while displaying distinctive personalities. I believe that being able to create artistically defined characters that display emotions and engage in personality-based social interactions will be as critical to this new artistic medium as it is to traditional, non-interactive, artistic media.

The goal of building believable agents is inherently an artistic one. Traditional AI goals of creating competence and building models of human cognition are only tangentially related because creating believability is not the same as creating intelligence or realism. Therefore, the tools that have been designed for those tasks are not appropriate. I will return to this point in section 1.4.1.

My approach to the problems of creating believable agents that are social and emotional is to create a new set of tools and methodologies that are suitable for the *artistic* nature of these problems. These tools and methodologies are a first step towards enabling artists to create interactive story systems with quality interactive characters that can display emotions and engage in believable social behaviors.

1.2 Structure and Contributions of the Thesis

This introductory chapter will provide an overview of interactive drama, interactive characters (or *believable agents*), and the relationship between the emotional and social aspects of agents. Because this is a new and rather unusual kind of problem (at least as far as traditional artificial intelligence is concerned), the background and motivation provided in the introduction are crucial to understand what follows.

After I have provided this important background material, the rest of the thesis is broken into three parts. Part I deals with the creation of believable emotional agents; Part II deals with the creating of believable social agents; Part III provides a summary of the main contributions of the thesis and some speculation about future directions for the research.

Part I of the thesis will focus on creating believable emotional agents. The major contributions of this part of the thesis include:

- A set of tools for creating believable emotional agents that includes:
  - a framework for building believable emotional agents,
  - a default set of emotional processes to provide reasonable default emotional behavior, and
  - discussions about how to create specific emotional characters within this framework.
- A methodology for creating emotions within a broad set of capabilities that allows artists to create emotionally rich characters. This methodology also enabled me to create models of how to generate emotions that rely on perception and motivation as well as cognition. These models can be simpler and faster than purely cognitive models
- Validation that the tools I have built can be used to create characters that users find to be both emotional and believable.

Part II of this thesis will focus on creating believable social agents. The major contributions of this part of the thesis include:

- A two-part methodology for creating believable social behaviors for specific characters.
    - Part 1 of the methodology suggests a number of important elements of personality that should be incorporated into social behaviors in order to make them personality-rich.
    - Part 2 of the methodology prescribes using a minimal amount of representation for modeling other agents in the environment.
- A set of believable social behaviors that provide:
    - case studies for explaining the methodology in depth and how to apply it in practice,
    - evidence for the breadth of behaviors the methodology can be used to create, and
    - examples of social behaviors for specific characters that use small amounts of representation of other agents.
- Validation that users find social characters build using this methodology can be good characters.

Finally, in Part III, I will summarize the contributions of the thesis and speculate about possible future directions for this work.

Now, I begin with an introduction to interactive drama, which will provide the context and motivation for the work described in the rest of the thesis.

1.3 Interactive Drama and the Oz Project

The dream of interactive drama isn't necessarily a unified one, though I don't mean that in a derogatory sense. I mean it in the same sense that fine arts or computer science or any other large field isn't coherent. "Interactive drama" is just an umbrella phrase for a set of rather different kinds of things.

Interactive drama might have a single human interactor (often called the "user" or the "player") or many interactors. The story might be created from a pre-defined set of possible user choices or it might emerge from a (possibly guided [Wehrauch96]) complex simulation. The user might "see" the world through text, animation, or a full-blown virtual-reality interface. The user might "act" in the drama by typing commands, moving a mouse, or through speech and gestures.

And these are only a few of the possibilities.

One type of interactive drama is interactive fiction, which involves a text-based interface where the user types commands and is provided with text descriptions of the world. The plots emerge from the simulation, which is (traditionally) a simple physical environment with mostly physically based puzzles.[1] Solving one puzzle allows the user to get to the next puzzle, which provides direction to the plot. Characters are very simple or non-existent. One popular example from this genre is Zork [Blank80].

Another type of interactive drama is the interactive play, which is (typically) a traditional play with a number of choice-points (usually one) that allow the audience to pick one of a few paths. All of the possible plots are determined ahead of time. The characters are the actors and do not directly interact with the audience. For example, near the end of "The Murder of Edwin Drood" [Holmes86] there is an intermission at which point the audience gets to decide who the murderer will be; the cast acts out a different (but pre-scripted) ending based on that choice.

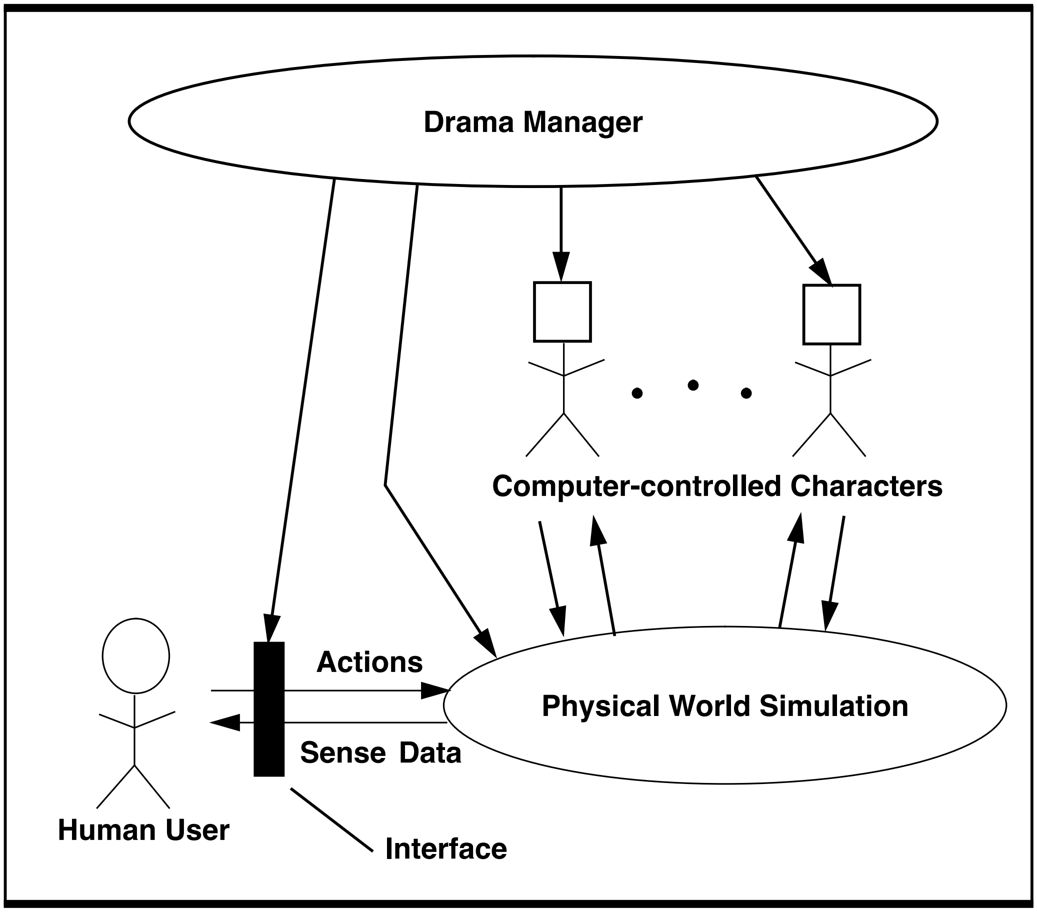
1.3.1 The Oz Project

The work I will describe is being done in the context of the Oz system, which is a set of tools for creating certain kinds of interactive drama. Although the Oz notion of interactive drama has its boundaries, I still believe that the ideas about interactive plot and characters that have developed within the Oz system will be applicable to a fair range of interactive drama systems [Bates92c].

Figure 1-1 shows the Oz architecture for an interactive drama system. There is a physical-world simulation that includes some number of characters. One of the characters is, or is controlled by, a human user. How the user "sees" the world and acts in the world are not strictly defined. There is a drama manager that is responsible for subtly controlling the user's experience towards some author-defined end. The drama manager can manipulate the physical world, the autonomous characters, and the user interface in order to create a dramatic experience for the user.

1. This is not completely accurate as some works that call themselves interactive fiction are hypertext based. For example, choose-your-own-adventure books are a kind of interactive fiction where the player acts by picking from a list of pages to turn to that represent different actions.

FIGURE 1-1 **The Oz Interactive Drama Architecture.**



Classic interactive fiction, like Zork, fits this model, though the drama manager slot is empty and the characters (other than the user) are few and simple. Interactive cinema would not fit this model because it is based on a pre-defined branching structure instead of on a physical simulation with autonomous characters. Choose-your-own adventure stories do not fit this model for the same reason.

Within this model we[1] have built two very different kinds of interactive drama systems. The first system is a text-based system that is similar to interactive-fiction games in feel; the user types commands and is given text descriptions of the world. We have built a number of worlds in this style, the first of which was an apartment with a house cat, Lyotard, as its sole occupant (see [Bates92a] & [Bates92b]). The user's goal was to explore the apartment and make friends with

1. "We," in this chapter, will refer to the members of the Oz group.

the cat, though there was no drama manager to push this story. The drama manager is ongoing research [Wehyrauch96] and was not used in any of the systems that I will describe in this thesis. Figure 1-2 provides a simple trace from a user interacting with this world.

The second type of system we have built is a graphical world with animated creatures. The user controls a creature in the world with a mouse. The only world we have built in this style so far is called the "Edge of Intention" or "The Woggles." This world contains three ellipsoidal creatures called woggles, each with a unique personality. The user controls the motions of a fourth woggle. Again, there is no drama manager in this world, so the "story" is generated purely from the interactions of the characters. Figure 1-3 shows the woggles posing for the camera.

1.4 Believable Agents

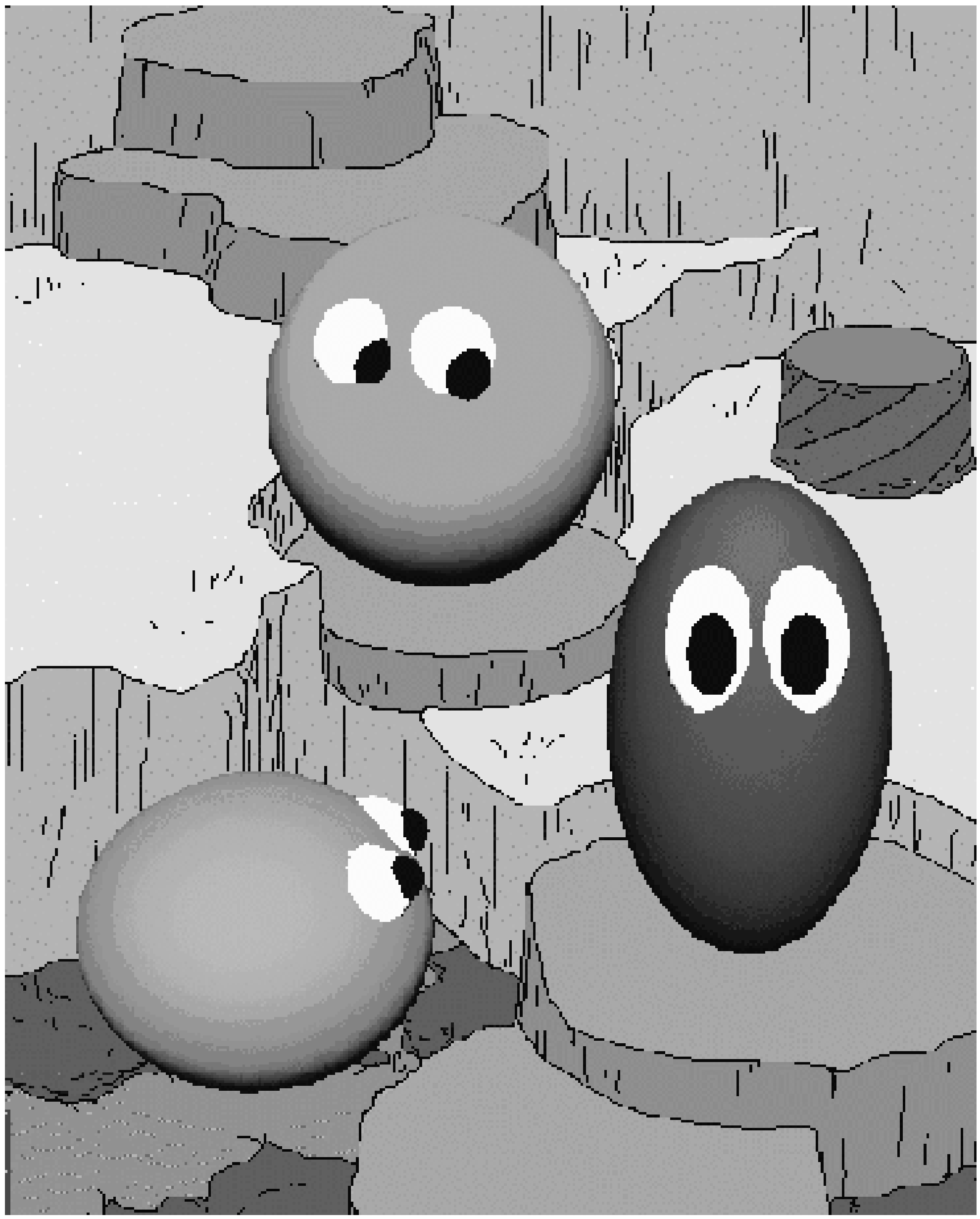
An important part of building good interactive drama is building good interactive characters. Not all good stories will require characters, but certainly many will. Much of the work on the Oz system has gone into building tools that will allow artists to create personality-rich, interactive characters.

This section presents the Oz approach to building believable agents in a top-down manner. I start with an overview of the nature of the problem. *Understanding the problem clearly is crucial to understanding my proposed solutions*. Then I explain the "broad but shallow" methodology for building believable agents. In section 1.4.4, I describe the Tok agent architecture; Tok is a broad architecture specifically developed for building believable agents. The section concludes with an brief overview of the Hap language, which is the programming language used to build the various components of the Tok architecture. Understanding the details of the language are important only insofar as they help the reader understand how I implemented the systems I will describe later in the thesis.

FIGURE 1-2 **An Interaction with Lyotard.** This is part of a larger trace and has been edited for brevity. The user types commands at the PLAYER> prompt. All other text is computer generated, so it may not be very interesting but is consistent with the underlying physical simulation. Consistency is a problem in many other interactive fiction systems that rely on canned text.

```
You are in the dining room.
To the south, you see the sunroom.
To the east, you see the kitchen.

The end table and the small chair are in the dining room.
The jar is on the end table.
The nine black sardines are in the jar.

PLAYER> take the jar
You take the jar.

PLAYER> go south
You are in the sunroom.
Lyotard goes to the sunroom.

PLAYER> give a sardine to lyotard
You offer the black sardine to Lyotard.
Lyotard runs to the dining room.

PLAYER> follow lyotard
You run to the dining room.
Lyotard looks around nervously.

PLAYER> pet lyotard
You pet Lyotard.
Lyotard bites you.
```

FIGURE 1-3 **The Woggles.** The stripes on the bodies are due to the world's lighting.



1.4.1 The Nature of the Goal

The problem of creating believable agents lies somewhere between the arts and artificial intelligence (AI).

Artists know how to create believable characters, although they do not have provably correct methods for doing so. If they did, there wouldn't ever be bad characters. Artists do, however, have a sense for what works and what doesn't. In order to create *believable* agents, we need to understand what artists know.

AI researchers know how to create autonomous agents. There are a number of AI architectures that can be used to create goal-driven, reactive, robust, autonomous agents. In order to create believable *agents*, we need to know what AI researchers know.

The goal of the Oz Project is to join these two disciplines to produce autonomous, interactive agents that have the qualities that have made the non-interactive characters of traditional media believable. This is what we mean by *believable agents*.

The Oz approach to creating believable agents (and my approach to creating believable emotions and social behaviors) is to start with the artistic nature of the goal and work backwards to the tools that are appropriate to such a goal. In other words, instead of bringing to the task our non-artistic notions of what the goal should be, we have read and studied the arts and listened to artists. Instead of starting with AI tools designed for other tasks, we have built tools specifically to support this artistic task.

The term *believable* is a specific term from the arts to describe characters that "work." Believable characters are characters that seem to be alive and that an audience has emotions for or about. Believable does not mean honest, convincing, or realistic. It gets at something else, something artistic. For the remainder of this section, I will discuss three lessons from the arts about the fundamental nature of believability that I feel are critical to understanding the (sometimes unusual) directions of the Oz work and, more specifically, my work.

1. **Believable agents may not be intelligent.**

   Much AI research is devoted to creating intelligent agents that can perform difficult tasks quickly and efficiently. In our domain, intelligence often takes a back seat to other concerns. It will even occasionally be desirable to have stupid characters (e.g., Forrest Gump or Woody on "Cheers").

Lesson: We don't want agent architectures that enforce rationality and intelligence. AI systems designed for these goals would be inappropriate for building believable agents.

2. **Believable agents may not be realistic.**

   Some branches of AI try to mimic nature, either in order to understand humans better or to produce efficient architectures. Cognitive modeling and work on "lifelike" agents (e.g., ALIVE [Maes95]) are examples of this kind of approach. Unrealistic characters, however, can be very believable. For instance, animated characters are far from being realistic, but often make very believable characters. In many animated films, the best characters are often talking animals, furniture, and other non-realistic sorts of things. Bugs Bunny and many of the characters in Disney's "Beauty and the Beast" are good examples.

   In fact, it turns out that sometimes being more realistic can *decrease* believability. For example, watching extremely realistic animation of human faces, like that of Terzopoulus [Terzopoulus95], can be somewhat disturbing, whereas watching unrealistic animation, like Charlie Brown, can be very satisfying. The reason for this is that the state of the art in computer animation can make a mostly realistic human face, but not a completely realistic one; this close-but-not-quite face is very disturbing to watch because people are so well adapted to watching human faces. From the standpoint of believability, *it is better to go with the less realistic characters which meet the audience's expectations than to go with the more realistic characters which don't*.

   Another key reason to avoid realism is that powerful artistic techniques, like abstraction/simplification and exaggeration, rely on altering reality for more effective characters. The idea behind these techniques is that the artist (or actor) is attempting to communicate the essential personality of the character to the audience. Because of this, artistic characters rarely (never?) have personalities as complex as humans, and the important traits they do have are often exaggerated for emphasis. Felix and Oscar in Neil Simon's "The Odd Couple" [Simon66] are good examples of simplified, exaggerated characters.

   Lesson: We don't want architectures that enforce realism. This means that we don't want systems that only generate externally realistic behavior. It also means that we are willing to use unrealistic internal processing—the goal is not to create cognitively plausible agents; it is to create good characters.

3. **Believable agents will have strong personalities.**

   Most AI researchers would be happy with personality-deficient agents that could competently perform useful tasks. However, one of the strengths of traditional characters is their interesting personalities. A goal of our work is to be able to create a variety of different agents, each with a distinct and interesting personality. These personalities should affect everything about the agent, including how the agent moves, thinks, and talks. Also, idiosyncratic quirks are extremely important parts of the agent's personality. Traditional AI isn't particularly interested in either artistic personalities or variation across agents.

Lesson: Personality should permeate the architecture and should not be constrained more than is absolutely necessary.

In our quest for believability, the Oz Project has adopted an approach that we think best allows us to achieve the kinds of characters we want. We call this the "broad but shallow" approach to building agents.

1.4.2 Broad Agents

Members of the Oz project (see, for example, [Bates91] and [Reilly94]) have previously argued that the way to create believable agents is to use broad agent architectures. Such architectures have three important properties: they have a broad set of capabilities, each capability is typically (but not necessarily) somewhat shallow, and all of the capabilities are tightly integrated. I will discuss each of these properties in turn and describe why we feel each is important for our primary goal of creating believable agents.

Using a Broad Set of Capabilities

The focus of much AI research has been on creating systems that do a small number of things particularly well, such as language understanding, problem solving, or learning (for an exception, see [Sloman94]). Even potentially broad architectures like SOAR [Laird87] were used for many years very narrowly—usually doing only a few things in any one program. (Recently, however, SOAR projects like TacAir SOAR have started to use more breadth [Tambe95].)

Believable agents need to be capable of behaving like interesting characters in a simulated environment. Such agents need to appear to have goals and emotions and they need to interact naturally and reasonably with their environment. They need to have enough capabilities to handle the variety of situations they are likely to encounter in an environment containing a human user. If the characters don't have a broad set of capabilities, they will likely break the user's suspension of disbelief. Such agents will need a broad set of capabilities, like perception, language understanding, language generation, emotions, goal-driven behavior, reactivity to the environment, memory, inference, social skills, and possibly others.

Some of these demands are simply a product of the agent needing to act within a complex, dynamic simulated world, but many are suggested to us by artists in other media. For instance, Thomas and Johnston [Thomas81], who are old-time Disney animators, tell us about the importance of emotion, perception, the appearance of thought, and interpersonal interactions for bringing animated characters to life.

Shallowness and the Suspension of Disbelief

We generally think, however, that we will only need to model the set of capabilities shallowly. Users of Oz worlds, like movie-goers and novel readers, will typically want to suspend their disbelief. This is similar to what happens with users of Weizenbaum's ELIZA system [Weizenbaum66] who buy into it despite its extreme shallowness. We expect users of interactive drama systems will want to suspend their disbelief just like moviegoers and novel readers do all the time. If this is true, we hope that as long as our characters don't act significantly out of character, users will tend to find them believable—often even coming up with reasons and excuses for unusual behavior.

Kelso et. al provide some experimental evidence to support this claim [Kelso92]. Users in their experiments were placed in situations similar to those that users would be in if they were interacting with an Oz-like system, except that the other characters and the director were humans instead of being computer-controlled. In this experiment, the users reported very intense experiences even though external observers found there to be numerous problems with both the story and the characters.

We're not claiming that depth of competence is necessarily bad or that it will never be necessary; we're only claiming that a broad set of shallow capabilities will be sufficient for creating many believable agents and is probably a more fruitful approach than the more typical narrow-and-deep methodology. A broad-and-deep approach would probably be more generally useful, but it is currently too difficult. We also don't believe that depth is necessary for many artistic agents.

Integration of Capabilities

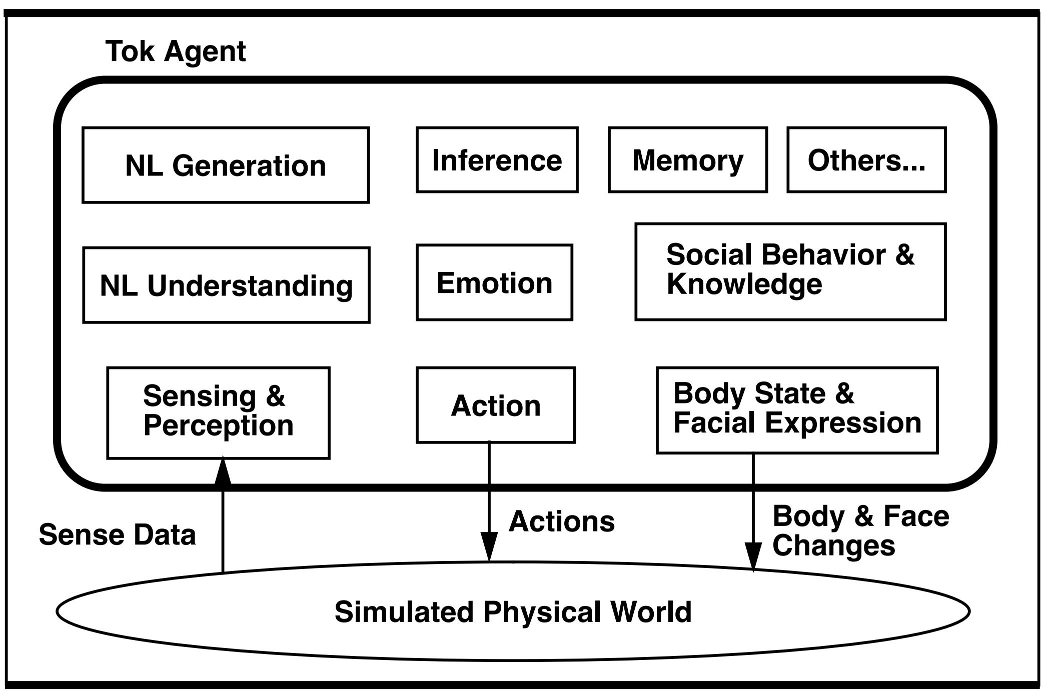
One of the key elements of this broad-and-shallow approach is the integration of the agent's various capabilities. Our experience has been that a large set of shallow capabilities becomes much more powerful when those capabilities are tightly integrated. By integrating capabilities, it is possible to create synergistic effects that make the result much more powerful than individual capabilities alone.

For example, let's look at the interactions between two capabilities: natural language and emotion. It isn't hard to imagine building characters that lack one or the other. In some cases that would even be desirable. Let's imagine, though, that that's not what we want, so we put in an emotion system and a natural language system and go to the effort of integrating them. Now we can have emotions that are based on language, such as being angry at a verbal insult (language understanding) or being frustrated at not being able to think of an appropriate word (language generation). Our agent is now also capable of talking about its emotions. Emotions might also affect how the agent speaks, such as stuttering when the agent is nervous. All of these important effects arise only once we have integrated emotions and language skills in our agent.

1.4.3 The Tok Agent Architecture

The members of the Oz Project designed and built the Tok agent architecture (see [Bates92a] and [Bates92b]) in an effort to provide the kind of integrated breadth necessary to build artistic characters for interactive drama. Figure 1-4 provides a high-level view of the Tok architecture. I have made no attempt to describe the interactions between the various components of the architecture here, although I will discuss many of these interactions throughout the thesis.

FIGURE 1-4 The Tok agent architecture



One way we achieve the tight integration of all of these components is by writing the code for all of the various subsystems in a common behavior-based programming language, Hap. In the next section, I will describe Hap in more detail.

1.4.4 The Hap Language

In order to build broad, believable agents, we needed a suitable language. In our case, we use the Hap language of Loyall and Bates [Loyall96]. The Hap language has been designed specifically to support believable agents, although it is similar to other work such as reactive architectures (e.g., [Firby89] and [Georgeff87]), behavior-based architectures (e.g., [Brooks86]), and situated action (e.g., [Agre90] and [Suchman88]).Throughout the thesis, I will point out various ways that I have extended the basic Hap language to better suit the task of building social and emotional agents.

It is not necessary to understand Hap very deeply in order to understand what follows. For the purposes of discussion in this thesis, the most important things to understand about Hap are the following:

1. Hap maintains a dynamic forest of active behaviors, with each behavior attempting to perform some set or sequence of actions. Those actions can be either external or internal (within the "mind"). The actions can also include subgoals, which lead to other behaviors.

   For example, an agent might have goals to eat when hungry, sleep when tired, and play otherwise. Each goal can lead to a number of different behaviors. One eat behavior might be to find a restaurant and eat there; both of these steps are subgoals that lead to other behaviors. The find-restaurant behavior might have a mental step that stores the location of the restaurant for future use.

2. All behaviors are pre-coded in a production memory. Hap does no planning in the traditional sense.

   For example, in the eat behavior, the behavior to find a restaurant is pre-coded by the artist. There is no on-the-fly planning to achieve this goal, though the choice of behaviors to accomplish goals depends on external perception and internal state.

3. Goal success is not necessarily a testable property of the world. That is, it may not be possible to write an expression that represents the state in which a goal has been achieved. Instead, some goals are purely behavioral in that they succeed when some set or sequence of actions has been performed.

   For example, one behavior the agent might use for the playing goal would be to throw a ball up in the air and catch it. The goal state and initial state are similar. The point of the goal is the execution of some action, not the achievement of some particular state of the world.

4. Hap supports the creation of reactive behaviors. Here are some of the ways that this is accomplished:
   - Demons can create new goals. For example, the goal to sleep is created when the agent gets tired.
   - Goals can be interrupted and resumed. For example, the goal to find a restaurant is temporarily suspended to dodge out of the way of an on-coming car and then resumed.
   - Preconditions on behaviors make sure that they are chosen only when appropriate. For example, the behavior to play with a ball depends on having a ball. If the agent doesn't have a ball, another behavior will have to be chosen or the goal will fail.
   - Goals can succeed serendipitously by means of success-tests that mark the goal as successful even if the associated behavior has not been accomplished. For example, if the agent's goal is to get to a restaurant and the agent starts off towards a known restaurant, the agent might come across a new restaurant along the way that will fulfill the goal even though the chosen behavior is not completed.
   - Behaviors can be rejected when the context of the agent changes—this is done by means of context-conditions that mark behaviors as having failed when the context is no longer appropriate. For example, the playing with a ball behavior depends on the agent having a ball. If another agent should take the ball away in the middle of the behavior, the behavior ends.
5. Hap allows multiple threads of processing. This way agents can have multiple goals being processed together. The highest priority goal will be processed until an action has been chosen for that goal or processing is otherwise halted. Then the next highest priority goal begins processing. This ends when all goals have been processed or when the time allotted to choose actions has expired.[1]
6. Hap is a general programming language, so many types of behaviors can be written using it, from physical behaviors to natural language behaviors to emotion-based behaviors[2].

1. There are two versions of Hap. One allows multiple threads; the other does not. Much of my work relates to both versions, though not all. Where it is relevant, I will point out what work is unique to one version of Hap or the other.

1.5 The Relationship between Emotions and Social Behavior

Now that I have presented an overview of the Oz system and believable agents, I will motivate my choice of focusing on the social and emotional aspects of believable agents.

This thesis may, at first, seem a bit disjoint—as though it were really just two small theses bound together: one about emotions for agents and one about social behaviors. As I described in the previous section, however, there is an advantage to breadth and these two aspects of agents seem particularly well suited for each other.

There are (at least) two ways that emotions and social skills combine to make Tok agents more believable. First, the combination allows for emotions that are greater in number and variety. Second, the relationships that agents have with each other are more believable because of this integration of capabilities. I will expand on these ideas in turn.

First, *social factors are very important in determining emotions*. Many causes of emotion in people and in artistic characters arise from social factors. For instance, anger, love, hate, jealousy, and grief are often associated with other agents in the world. Without social knowledge and relationships with other agents, there would be a large gap in the types of emotions our agents could express.

Relationships also affect an agent's emotions in other ways, such as modifying what emotions are felt and how strong they are when the cause of the emotion is another agent. For example, if Sue hears Bob insulting Fred, she might have very different emotional reactions based on her relationships with Bob and Fred. If she is friends with Fred, she might feel intensely angry at Bob. If she likes Bob and not Fred, she might find Bob somewhat amusing instead.

Second, *the relationships between agents are more believable because of the integration of emotions and interpersonal relationships*. This is because the dynamics of social relationships often depend on emotions. If Bill is regularly mean to John, it is likely that John will get angry at Bill and eventually learn to dislike Bill. If John couldn't feel anger, he would probably continue to associate with Bill, which seems less believable. Tok agents could enter into social rela-

2. Emotion-based behaviors are things like generating fear when an important goal of the agent is threatened. Other kinds of behaviors (such as physical behaviors) written in Hap can also take emotional information into account in a variety of ways, such as walking across a room angrily. This will be expanded on in Chapter 5.

tionships with other agents without having emotions, but by integrating these two capabilities these agents have much richer relationships that can change over time based on emotional factors.

1.6 Simulation Systems

Before I get on with the rest of the thesis, I will briefly introduce three simulation systems that I designed and built as part of this research. I used these three systems to help test and focus my research. As I will describe, each of the simulations I have built has been designed to push specific areas of this work.

All of the systems I will describe here are text-based, mostly because I didn't want to spend too much time on unrelated difficulties dealing with animation, which is not one of my areas of expertise.[1] The Oz system also had some tools for doing text-based speech interactions, whereas there were no methods for doing either real or simulated speech in the animated systems. By working in the text-based system I was able to explore more complex social behaviors, like negotiation, than if I had used the animation-based system.

The three simulations I built are called "Robbery World," "Office Politics," and "The Playground." Each consists of a simple physical world with a few distinct locations and a few characters. The typical interaction runs roughly 10 to 20 minutes. I think of them as interactive versions of short stories or animated shorts. They are not especially large but are still complex enough to be interesting.

The text descriptions of the environment and of events in the world are all computer generated. This often makes them a bit stilted and, occasionally, ungrammatical. In most of the traces I will present in the thesis, I will edit them to make them more readable. This is the case with the traces presented here. Unedited traces can be found in Appendix A.

It is important to point out that the natural language capabilities of these agents are much more limited than they might appear. The language understanding systems are based on keyword matching and the language generation is done by templates. For the worlds I have created, these simple mechanisms have proved mostly adequate. More importantly, though, they have allowed me to pursue work in emotion and social behavior without having to first solve the difficult problems involved in natural language generation and understanding. Loyall's

1. As described in section 1.3, the Oz system has been used to develop both text-based and animation-based interactive systems. The work described in this thesis has been incorporated into both types of systems.

work [Loyall96] should prove to be a much better solution for believable language generation than the simple mechanisms I have used.

"Robbery World" was designed to push the ability of my tools to model characters' emotions. The user plays the part of a police officer attempting to thwart a convenience-store holdup. The other characters are a gunman and a cashier. Figure 1-5 shows part of a sample trace from an interaction with this simulation. For more traces, see Appendix A. As noted before, the text output of this system is all computer generated, so it tends to be somewhat stilted. Hopefully, the emotions and social interactions of the characters will be evident despite the interface.

FIGURE 1-5 **An interaction with "Robbery World"**

```
You are in the parking lot.
To the north, you see the convenience store.
The cashier and the gunman are in the convenience store.
The gunman is holding his gun.
The gunman is wearing the ski mask.
The cashier is holding the bag.

The gunman is now red.
The gunman is now scowling.
The gunman is now tense.
The gunman says to you ''Back off and nobody gets hurt!''.
The cashier says to the gunman ''Please don't kill me. I'll
give you whatever you want.''.

PLAYER> Gunman: Come out of there now, you low-life scum!
The gunman says to you ''Hey, shut your trap! I ain't listen-
ing to none of your crap!''.

PLAYER> Gunman: Come out now or I'm coming in!
The gunman says to the cashier ''Come on! Hurry up!''.

PLAYER> go north
The gunman aims his gun at you.

PLAYER> shoot the gunman
The gunman is now wounded.
The gunman is now pale.
The gunman is now bug-eyed.
The gunman is now trembling.
The gunman says to you ''OK. I give up. I'm turning myself
in.''.
```

“Office Politics” was designed to test some social aspects of the characters, such as how relationships between characters might affect their behaviors. It also provides characters that engage in simple social behaviors, like helping and deceiving each other. The user is a project manager attempting to send off an email project proposal by 5 pm but the email server is broken. The other characters include Mary, the user’s boss; Gus, the technician; and Sarah, another manager. This scenario was inspired by Cesta and Miceli’s work [Cesta93]. They use a similar scenario to demonstrate their approach to agents modeling other agents. Since my approach to this problem is quite different, this system provides a point of comparison to other work in this area. Figure 1-6 shows part of a trace from this simulation.

FIGURE 1-6 An Interaction with “Office Politics”

```
You are in Gus’s office.
The desk, the chair and Gus are in Gus’s office.
Gus works.

[4:35] PLAYER> Gus: Can you help me fix the email server?
Gus says ‘‘Sorry, bud. I’d like to help, but I’m swamped to-
day.’’.

[4:36] PLAYER> go north
You are in the eastern hallway.
To the west, you see the western hallway.

[4:37] PLAYER> go west
You are in the western hallway.
To the north, you see Mary’s office.

[4:38] PLAYER> go north
You are in Mary’s office.
The desk, the chair and Mary are in Mary’s office.
Mary works.

[4:39] PLAYER> Mary: I have to get a project proposal out and
the server is broken.
Mary says ‘‘I don’t know how to fix the server. Why don’t you
ask Gus?’’.

[4:40] PLAYER> Mary: I did, but Gus is too busy to help.
Mary says ‘‘Alright, I’ll talk to Gus, but I really don’t
have time for this.’’.
```

"The Playground" was also designed to test the social side of the agents. In this simulation, the user is a school child trading baseball cards during recess. The other characters are Sluggo, the bully, and Melvin, the nerd. As will be argued in Chapter 10, this simulation shows that I can create believable characters that engage in reasonably complex social behaviors like negotiation and initiating relationships with others. A partial trace from this simulation can be found in Figure 1-7.

FIGURE 1-7 An Interaction with "The Playground"

```
PLAYER> get in the sandbox
Melvin is now smiling.
Melvin says ''Greetings, earthling.''.

PLAYER> Melvin: Wanna trade?
Melvin says ''The Klingon Emperor wants to know if you
would be willing to part with Babe Ruth for Reggie Jack-
son?''.

PLAYER> Melvin: No thanks.
Melvin says ''Captain Picard would make the trade.''.

PLAYER> Melvin: Sorry, no deal.
Melvin says ''Are there any cards we (the Klingon high com-
mand and I) can offer you in return for Babe Ruth?''.

PLAYER> Melvin: Well, do you have anything that would help
me get Mays from Sluggo?
Melvin says ''So the Vulcan ambassador wishes to get Willie
Mays from Darth Sluggo. He really likes Mickey Mantle you
know and I'd trade you.''.

PLAYER> Melvin: Okay, how about Mantle for Ruth?
Melvin says ''The Klingon high command accepts Vulcan's
generous offer.''.

[Time passes. The trade is finalized and the user approach-
es Sluggo.]

PLAYER> Sluggo: Do you want to trade with me?
Sluggo says ''Make an offer, butthead.''.
```

1.7 Summary

Here are some of the important issues that have come up in this chapter.

- I introduced the idea of interactive drama, where a human user gets to play the part of a character in a interactive story-based simulation. I also described the Oz system and its model of interactive drama.
- I introduced the problem of enabling artists to build interactive characters for interactive drama systems. These interactive characters are called *believable agents*. Because this is inherently an artistic problem, I described some of the lessons we can learn from the arts about how to make good characters: they don't need to be intelligent; they don't need to be realistic; they should have distinctive personalities. All of these affect the kinds of approaches that are appropriate for solving the problem.
- I described the Oz approach to building agents with broad sets of shallow but tightly integrated capabilities and argued that this is a reasonable approach to building believable agents. I also described the Tok agent architecture and the Hap language that is used to write most of the components of the Tok architecture.
- I argued that emotions and social behaviors are distinct but closely related parts of the agent architecture. By studying both I am able to make both systems richer and more interesting.
- I described three simulated systems and seven believable agents that I have built using the techniques that will be described in the thesis. These three systems, *Robbery World*, *Office Politics*, and *The Playground*, were used to motivate and test much of the research described in the thesis.

Part I:

Believable Emotional Agents

CHAPTER 2 Believable Emotional Agents

2.1 Introduction to the Problem

Characters in non-interactive media, like novels and movies, are often emotional. In fact, artists tell us that emotions are critical to the believability of their characters. Frank Thomas and Ollie Johnston, two of Disney's original animators, wrote a book called *The Illusion of Life* about creating believable animated characters; here are some of the things they have to say [Thomas81]:

> From the very beginning, it was obvious that these feelings of the characters would be the heart and soul of Disney pictures. (p.473)

> From the earliest days, it has been the portrayal of emotions that has given the Disney characters the illusion of life. (p.505)

The overriding goal for this part of the thesis is to enable artists to create interactive versions of believable emotional characters like the ones that Thomas and Johnston talk about. This is what I mean by *believable emotional agents*.

2.2 Foundation

My primary goal is to enable artists to create believable emotional agents. In order to accomplish this goal, I drew on previous work from a range of sources, including art, psychology, and AI. The arts helped me understand the problem better and psychology and AI provided some insights into how I might solve the problem.

2.2.1 Art & Entertainment

It is artists who best know how to create believable characters and how to imbue them with emotions, so it is fitting to turn to the arts for guidance in building interactive believable characters.

The first contribution of artists is that they identify emotion as an important problem for building believable agents. For instance, the excerpts from Thomas and Johnston at the beginning of the chapter indicate how important they feel emotions are for creating quality characters.

Artists also provide ideas about how to create effective emotions for characters. These ideas are not formal, so they cannot be directly implemented, but they have helped me make a number of important design decisions that I will discuss in more detail later.

One important idea about how to create effective emotional agents is that the *emotions should be specific to the character*. In other words, each character needs to be unique and its emotions need to fit its particular personality. Again, here are some excerpts from *The Illusion of Life* [Thomas81]:

> These characters showed hatred and scorn in their own way, but in a convincing manner. They were equally entertaining, but they were in no way interchangeable, which points up the importance of the storyman's knowing his characters…. (p. 483)
>
> … [I]t is the animator who must think deeply into the personality of the cartoon actors. Each must be handled differently, because each will express his emotions in his own way. (p. 487)

These quotations refer to the expression of emotion, but it is also important for the characters to have individual emotional reactions to situations as well. For instance, in Disney's *Snow White*, each of the seven dwarves might feel very differently about a single event because of his distinctive personality. And, as the quotations above state, even when characters have similar responses, they need to express those reactions individually.

Another important idea from the arts is that the *characters' emotions need to be expressed broadly*. That is, emotions must affect everything about the character: the way it moves, the way it talks, the expression on its face. An underlying assumption here is that the purpose of a good character is to clearly communicate its thoughts, feelings, and personality to the audience (or, in the case of interactive characters, the user). By expressing emotion in only the character's face, for example, artists find it harder to communicate than if the whole character is used to express the emotion. Thomas and Johnston have this to say on the subject [Thomas81]:

> If a scene calls for showing tense emotions such as anguish, scorn, bitterness, or envy with only facial expression, the animator will be quite limited. But if the story is built so that the character reveals these feelings in what he does and how he does it... the scenes can be gripping and entertaining. (p. 482)
>
> *The expression must be captured throughout the whole body as well as in the face*. (p. 443) [Emphasis in the original.]
>
> One producer at Disney's insisted that if a character said he felt a certain way, that was all that was needed.... But it does not work like that. It is not enough simply to proclaim that a character is mad or worried or impatient. There must be business to support the statement and a situation in which he can demonstrate these emotions if the audience is to be convinced that it is so. (p. 387)

Finally, the arts remind us that *the goal is believable emotions, not realistic emotions*. Artists will often want to create characters that are exaggerated or "larger than life," which is at odds with achieving realism. Also, animated characters can be believable, even though they are clearly unrealistic. Some characters may seem quite realistic; others will be wildly unrealistic, but they can all be *believable* in the artistic sense of the word—I want to enable artists to create whichever they want. My experience has been that this is the hardest of the artistic principles to grasp—even Walt Disney had trouble expressing it. Again, from [Thomas81]:

> There was some confusion among the animators when Walt first asked for more realism then criticized the result because it was not exaggerated enough... When Walt asked for realism, he wanted a caricature of realism. One artist analyzed it correctly when he said, "I don't think he meant 'realism.' I think he meant something that was more convincing, that made a bigger contact with people, and he just said 'realism' because 'real' things do... (p. 66)

Thomas and Johnston, however, are not unclear on the issue [Thomas81]:

> *It should be believable, but not realistic*.... Tell your story through the broad cartoon characters rather than the "straight" ones. There is no way to animate strong-enough attitudes, feelings, or expressions on realistic characters to get the communication you should have. The more real, the less latitude for clear communication. (p. 375, emphasis added)

Although I have focused primarily on the animation work reported in Thomas and Johnston, these ideas are not particular to them or to animation. Some of these ideas can be traced as far back as Aristotle's *Poetics* [Aristotle87b] and artists in other media (such as screenplays [Horton94], novels [Gardner91], and even comic books [McCloud91]) make similar claims.

To summarize, the four important lessons to draw from the arts are:

- Emotions are important for creating believable characters.

- Emotions need to be specific to the character in question.
- Emotions need to be expressed broadly.
- Emotions must be believable but may not always be realistic.

2.2.2 Psychology & Cognitive AI

The arts provide insights into the problem of creating believable emotional agents, but they do not provide computational ideas for how to build autonomous agents with emotions. Psychology and cognitive AI provide some ideas about how to approach this problem.

Once again, as the goal of this work is based in the arts, not in psychology, I am not particularly interested in using cognitively plausible models of emotion. In fact, as indicated in the previous section, artists may well want characters with emotions that are cognitively unrealistic, but nonetheless appropriate for the characters.

This means that the emotion theories I draw on do not have to be "correct" to suit my needs. They only need to be able to help artists build believable emotional characters. I chose as a basis for my work the emotion theories of Ortony, Clore and Collins (OCC) [Ortony88] and Gilboa and Ortony [Elliott92][1]. The first describes when people are emotional and the second describes how people express emotions.

One reason for choosing these models is that they were designed to be implemented computationally. Other researchers (e.g., [Elliott92] and [Warner91]) have also implemented versions of these models.

Another reason for adopting these models is that they are reasonably simple to understand. Because I eventually want artists to use these models and it is likely that these artists will not have much formal psychology training, I wanted to keep the models as simple as possible. Decisions about what is "simple" are clearly subjective and it is possible that my tools are harder to use than if I had chosen some other models. I will describe my decisions and how I made them in this thesis but I leave it as future work to determine if other models are more useful and easily understood by artists.

1. Gilboa and Ortony never published this theory, though it is described in [Elliott92].

I will, however, briefly describe two of the models that I did not choose as a basis for my work: the basic emotion model of Oatley [Oatley92] and the emergent emotion model of Sloman [Sloman86].

Oatley hypothesizes five basic emotions that are the foundation of all emotional experience: joy, distress, fear, anger, and disgust. All other emotions are hypothesized to be related to these emotions in various (and, it appears to me, unspecified) ways. Although it is possible that this model has some psychological basis in reality, because the model doesn't make explicit how specific, non-basic emotions relate to the basic five, I felt it would make it difficult for the artist who wanted to create characters with non-basic emotions like hope or jealousy.

Sloman (like Simon [Simon67]) hypothesizes that emotions are emergent properties of complex, resource-limited, motivation-processing systems. In this model, emotions are states of the overall system and there is no separate emotion component. In other words, mechanisms of the mind designed to deal with the difficulties of a complex environment can be in "perturbance" states where the agent's high-level cognitive processes are partially "out of control." Sloman calls these "perturbance" states "emotional." (Note that he uses the word "emotional," but he does not use the word "emotion" because of the confusion surrounding the definition of that word.) Beaudoin [Beaudoin94] has also explored this model in depth and built a simulated environment to test and demonstrate some of these ideas. Again, this may be sound psychology, but, as I will discuss, I have chosen to create an explicit emotion system to give artists more direct control over the emotions of their characters.

It may be that as artists want more and more complex characters and as psychological models become more and more accurate and powerful, a new set of tools will need to be built that rely on more cognitively plausible models. The important thing to remember is that the goal is an artistic one and using cognitively plausible models is only appropriate if it helps achieve this goal.

2.2.3 Story-Based AI

Other AI researchers who have influenced my work come from the area of story-based AI. I have found useful ideas and inspiration in the work of Meehan [Meehan76], Carbonell [Carbonell79], Dyer [Dyer83], and Lebowitz [Lebowitz84,Lebowitz85]. Each has provided some insight into the problem of building believable emotional characters.

Dyer's BORIS system understands stories about emotional episodes, such as divorces. BORIS does not generate stories, nor is it interactive, so on the surface it may not appear especially similar. However, in order to store the emotional content of stories, Dyer created an AFFECT structure that I modified and used as a means of storing the emotional experiences of characters.

Lebowitz and Meehan developed story-generation systems that include characters. Lebowitz's UNIVERSE creates soap operas and Meehan's TALE-SPIN creates fables. Both systems adopt ways of representing relationships and feelings about other characters that are similar to the attitude system that I have adopted. The *attitudes* that characters have towards each other in UNIVERSE are described along four dimensions (like-dislike, attractedness, dominant-submissive, and intimate-distant). Meehan's characters use the following scales: affection, competition, deception, trust, domination, familiarity, indebtedness.

These systems relate to my work in two important ways, as I shall describe: they support my decision to study the emotional and social aspects of characters together and they provide good examples of the importance of providing the artist with as many choices as possible.

The first idea is fundamental to the dual nature of this thesis; I feel that *the emotional and social aspects of characters are closely enough related that to do either one well, it is necessary to do both*. This is the argument I presented in section 1.5. The fact that both Lebowitz and Meehan were able to use a single system to handle emotions and social relationships of agents gives additional support to the interconnection of these two facets of agents.

The second idea, that *I want to supply choices to the artist,* is based on an observation about the choices that Lebowitz and Meehan made. Both devised a system to model the relationships and emotions that characters can have about each other. The systems are quite different from each other and Lebowitz's model is different from the psychological model that he drew on [Wish76]. Furthermore, the attitude system that the OCC model [Ortony88] proposes has only like and dislike attitudes. So, here are four different models, all of which are useful, though limited. And if I were to choose the superset of all of these models, the chances are good that I would still miss attitudes that artists would want to use. I didn't want to build in assumptions (like these other systems did) about things like what attitudes characters can have about each other for fear of stifling the creativity of artists. The contrast of the different attitude systems of Lebowitz, Meehan, Wish, and OCC with my more general solution to this problem, provides an example my overall approach of providing artists with freedom and flexibility whenever possible.

Continuing with the look at story-based AI, Carbonell focused more on personality than emotions in his work. His analysis of different personality types suggests how various traits can be expressed through plans, goals, and reactions to failure.

(The goal of this analysis was to understand stories better.) This analysis, however, proves useful for understanding many of the ways emotions can be expressed. For example, one of the reaction-to-failure traits is "depressed." A depressed agent, according to this theory, will often respond to failures by being more likely to abandon the plan and goal. In Chapter 5, I will discuss a number of ways to express emotions, one of which is making the current plan and goal more likely to be abandoned.

2.3 Contribution: Tools for Building Emotional Agents

In this section I will introduce a set of tools, collectively called Em, that support artists in the creation of believable emotional agents. In section 2.4, I will discuss how working within a broad agent architecture has helped me create these tools.

Using the foundational work described in the previous section, I created a number of tools that support the creation of believable emotional agents: a framework (or architecture) for building emotional agents, a specific system built within this architecture that provides reasonable default emotional processing, and discussions about how to use the first two tools to create specific believable emotional agents.

Before I describe these tools, it is important to recall that the goal is not to create cognitively plausible emotional agents. The architecture allows many unrealistic (but possibly interesting) agents to be built and the default emotional processing, though informed by the psychology literature, has been tailored to meet a specific artistic end.

2.3.1 The Em Emotion Architecture

The first tool that I provide artists is an *emotion architecture*[1] that sits within a larger agent architecture. The emotion architecture determines the boundaries of what is and is not possible for the agent builder to create in terms of emotional agents. For example, the architecture determines what inputs are available to the agent builder for determining which emotions the agent will have. If the agent builder didn't have access to the agent's goals, it would be impossible to create emotions based on those goals.

1. The definition of *emotion architecture* and many of the other terms I will use can be found in Figures 2-2 and 2-3. (Additional terms will be introduced in future chapters.) I have tried to avoid using the generic term *emotion* because of the confusion it can cause. Read and Sloman [Read93] have previously discussed some of the terminological perils associated with working in the area of emotion research. To reiterate the note in the two figures, much of the terminology I use is specific to my work. The underlying assumptions behind this terminology will be made clear in the next few chapters.

Figure 2-1 provides a look at the bare-bones version of the Em emotion architecture. I describe the details of the Em architecture in Chapters 3 through 5, but I begin here with a brief overview of the whole architecture.

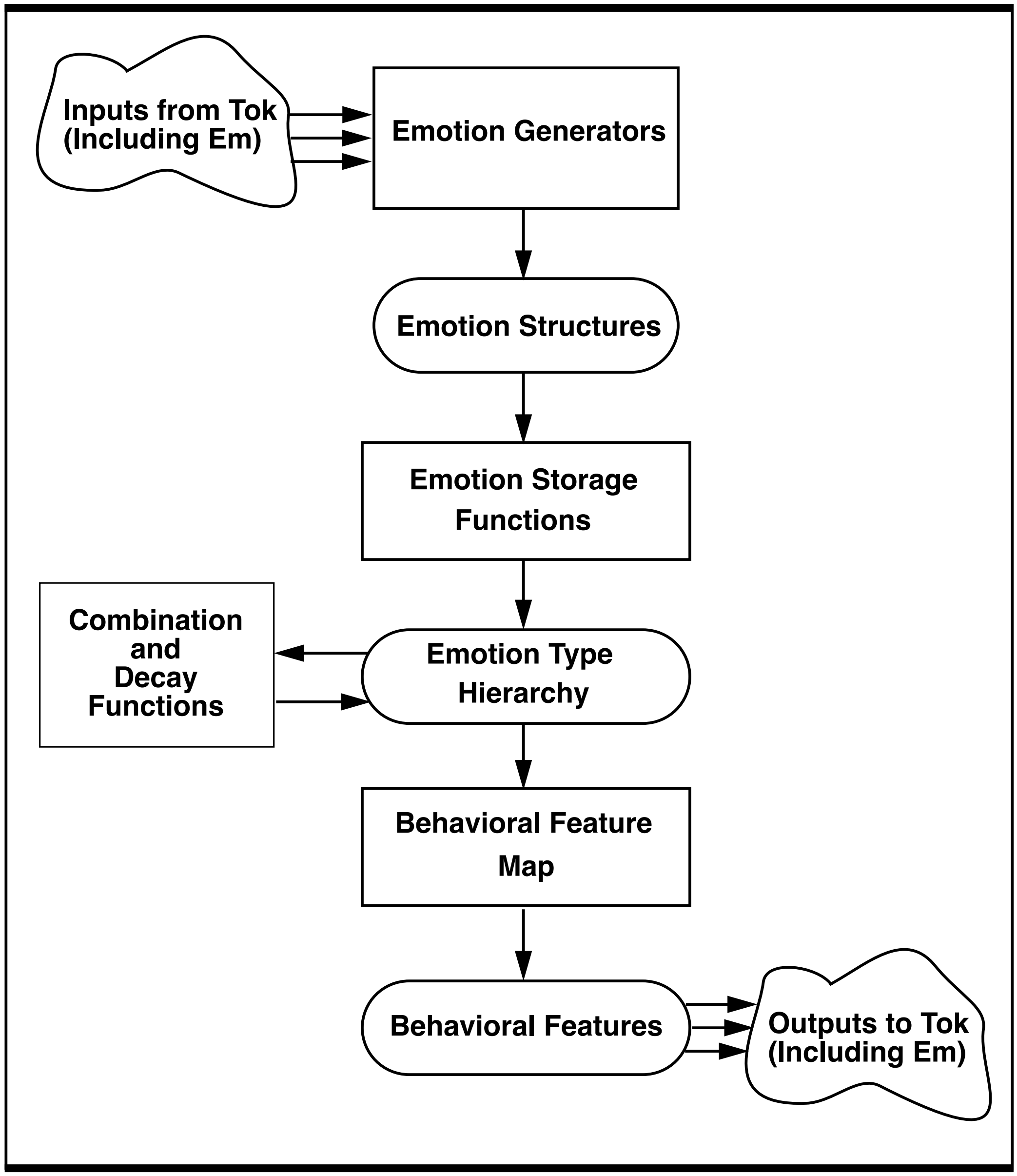
The first thing to notice in Figure 2-1 are the **Inputs from Tok**. These inputs are used to decide when an agent should react emotionally. The Em architecture provides a wide range of inputs to the artist-defined rules that determine what emotions the agent will display.

The **Emotion Generators** box represents this set of rules (called emotion generators) that take the set of inputs and produce a set of **Emotion Structures**. These rules are written in the Hap language. An example emotion generation rule is the following: when an agent has a goal failure and the goal has importance X, generate an emotion structure of type distress and with intensity X. Emotion structures have a type (e.g., fear), an intensity (e.g., 7 out of 10), possibly a direction (e.g., Sluggo), and a cause (e.g., Sluggo is threatening to beat me up). Details of the inputs to the emotion architecture and the emotion generators will be provided in Chapter 3.

A set of **Emotion Storage Functions** takes the emotion structures as they are created and puts them into an **Emotion Type Hierarchy**. Emotion structures are placed in this hierarchy based on what kinds of effects they will have, with higher-level nodes representing more general effects and lower-level nodes representing more specific effects. For example, the hierarchy might have a distress type that represents general forms of distress expression, like frowning, crying, and moving slowly. Below that type might be subtypes, such as grief, homesickness, and lovesickness, which inherit the general effects of their common parent, but that also have more specific means of expression as well, such as thinking about home when homesick.

Each type in the hierarchy (e.g., distress) has an intensity associated with it that is a function on the intensities of the emotion structures of that type. The way that the intensities of the emotion structures are combined is determined by a set of **Emotion Combination Functions**.

FIGURE 2-1 **The Em Architecture**



The intensity of the emotion structures will decay over time at a rate specified by the artist in the **Emotion Decay Functions**. Each structure can have its own decay function if desired (e.g., anger from being insulted decays slower than other emotion structures), or decay functions can be defined at the emotion-type level (e.g., all anger emotion structures decay slowly) or over all emotions (i.e., all emotions decay at the same rate). Details about storing, combining, and decaying emotions will be provided in Chapter 4.

The emotion structures are mapped into **Behavioral Features** via a **Behavioral Feature Map**. This arbitrary mapping is written in Hap. It is the behavioral features, and not the emotion structures, that directly affect behavior.

The final component of the Em architecture are the **Outputs to Tok**. The behavioral features are able to affect a number of different aspects of the agent's processing. More details about the behavioral features and the effects on the agent will be provided in Chapter 5.

2.3.2 The Default Em Emotion System

The Em architecture provides the structure that an artist will work within when creating emotional characters, but none of the content. And because the architecture is so flexible for artistic reasons, it can be hard to know how to begin. For instance, artists have a large amount of flexibility in determining how to map inputs to emotion structures, but coming up with a good mapping is still a hard problem. Similarly, determining the speed of decay, the proper combination functions, a good type hierarchy, and the other decisions to be made are all difficult.

Because of this, I have not only provided an architecture for creating emotional agents; I have also created a default emotion system. An emotion system is essentially a filling in of the architecture to provide behaviors that are reasonable. It may be useful to think of the emotion architecture as a programming language for writing programs that control the emotions of characters and the default system as a sample program. Artists will probably still want to write their own program, but if they already have one to modify or at least to learn from, creating other programs should be simplified.

Chapters 3 through 5 will describe the default emotion system in great detail, but here is a brief overview of what it contains:

- I provide a set of emotion generators based on the cognitive emotion model of Ortony et al. [Ortony88]. These generators will create 24 different types of emotion structures. (Chapter 3)

- These structures are stored in a default emotion type hierarchy. (Chapter 4)
- The intensities of the emotion structures are combined according to a default combination function and they decay using one of two default decay rates. (Chapter 4)
- A default behavioral-feature map creates 33 types of behavioral features based on the current emotional state. (Chapter 5)
- Em provides some default mappings from behavioral features into changes in the way the agent behaves. It is generally difficult to provide default behavior based on the behavioral features because of the wide variety of characters artists might be working on. However, I have created default mappings from behavioral features to a limited set of emotional effects that I have found are somewhat general, including the body state and changes in attitudes towards other agents. I have also built a number of general functions that artists can use for computing the priorities of goals based on input from the behavioral features. (Chapter 5)

2.3.3 Thesis as How-To Manual

Although I provide a default emotion system for agent builders to start from, I also want to teach them (as much as possible) how to create emotion systems specialized for their particular agents. So, the third kind of tool I provide agent builders is a how-to manual. Keeping with the analogy of the emotion architecture as a programming language and the default system as a program, I intend for much of the discussion in this thesis to play the part of the programming manual which describes good programming technique as well as how to use the language to write certain kinds of programs.

Throughout the next three chapters I will not just describe the Em architecture and default system; I will also discuss how to use them. For example, in Chapter 3, I will introduce the architectural support for emotion generators and describe a set of default emotion generators that I provide. I will also, however, discuss ways that the architecture can be used that the default rules don't take advantage of. This will provide artists ideas about how to approach problems and personalities that are not directly handled in the default emotion system.

FIGURE 2-2 **Emotion Terminology (part 1)**

[The terminology in this figure is not meant to be representative of anything but this thesis. Many of the terms used are specific to work I have done and contain implicit assumptions about my approach.]

Emotion architecture. A general framework for creating *emotion systems* for particular agents. I will describe an emotion architecture, called Em, in Chapters 3 through 5.

Emotion system. The emotional makeup of a particular agent. For example, getting violently angry when insulted is an aspect of an agent's emotion system. How important an agent feels certain goals are is another aspect. So are the appraisals for determining how likely a given goal is to succeed. Specific emotional episodes (e.g., getting angry) are a result of the emotion system but are considered part of the *emotion state*. I will describe how to create emotion systems within the Em emotion architecture in Chapters 3 through 5.

Emotion generators. Anything that produces an *emotion structure* is an emotion generator. A demon that creates joy whenever one of an agent's important goals succeeds is an emotion generator. Such a demon will also be referred to as an *emotion generation rule*.

Emotion structures. Structures representing specific emotional experiences. In the Em architecture, emotion structures include type, intensity, cause, and directional information. For example, if Sam is threatening to beat up Tom, then an emotion generator in Tom might generate an emotion structure of type FEAR. The intensity might be 7 (on a 1-10 scale). The direction represents who the fear is of: Sam. In this case, the cause is the fact that Tom doesn't want to be beaten up and may be represented by a goal structure that represents that information.

Emotional state. The current set of all emotion structures present in an agent. For example, a child being threatened by a bully may feel fear, sadness, and anger (to varying degrees) at the same time.

FIGURE 2-3 **Emotion Terminology (part 2)**

[The terminology in this figure is not meant to be representative of anything but this thesis. Many of the terms used are specific to work I have done and contain implicit assumptions about my approach.]

Emotion types. Emotion types represent sets of emotions that are similar in how they affect the behavior of the agent. For instance, distress might have a number of subtypes, including grief and homesickness. All distress emotions will share certain effects, like slowing movement and frowning. Other effects are specific to the subtypes, like thinking about home when homesick but not when grieving.

Behavioral Features (BFs). Intermediate structures between emotion structures and *emotion effects*. BFs allow emotion effects to correspond to the current emotional structures without being tied directly to those structures. BFs have the same components as emotion structures: type, intensity, direction, and cause.

Behavioral Feature Map. A mapping (typically) from emotion structures to behavioral features. The BF Map might map emotion structures of type ANGER to behavioral features of type AGGRESSIVE. The mapping can take non-emotional inputs into account, such as creating an aggressive BF to help achieve a goal instead of for emotional reasons.

Emotion effects. Any way that the agent changes based on the current behavioral features. An effect might be frowning or adding a revenge goal to the motivation system. Effects can also occur in the emotion system itself.

Attitudes. Attitudes are long-term feelings about people or objects. For example, Bill might like Samantha and dislike green beans. These attitudes might give rise to emotion structures but are not themselves emotion structures. Attitudes may change over time but tend to change slowly (as opposed to emotion structures).

Moods. Moods are abstractions based on the current emotional state. For instance, I will refer to a character with a large number of "positive" emotion structures (e.g., joy, hope) as being in a good mood. Moods are determined by the behavioral feature map, so different emotion systems can be built to support different kinds of moods beyond "good" and "bad."

2.4 Key Idea: Broad Emotional Agents

In building the tools described in the previous section, I have derived a lot of power from using a broad, integrated agent architecture (as described in section 1.4). One reason for taking this approach is that I expected that it would provide a good deal of generative and expressive power. In other words, there would be more to have emotions about and more ways to express those emotions.

For example, as I argued in Chapter 1, emotions and social relationships are tightly interconnected. Emotions often arise based on relationships and relationships often change based on the emotions others cause in us. To build emotions apart from the social aspects of agents necessarily limits what emotions can be generated and how they can be expressed.

Other AI researchers study emotions apart from important parts of the architecture, such as a real motivation system, or, when they do have such components, they are often overly shallow or not well integrated with the emotion system. For instance, Elliott's Affective Reasoner [Elliott92] allows emotions about goals, but only permanent goals (i.e., no subgoals). In Tok, emotions can arise in response to the full set of the agent's subgoals. Also, systems with weak motivation systems are limited in the ways that emotions can be expressed through goals, behaviors, and actions. By deeply integrating Em with a full motivation system, Tok provides a rich set of ways of expressing emotions through action.

This broad approach might seem obvious, but most emotion researchers try to focus on emotions apart from the rest of the architecture.[1] To focus on one hard problem at a time seems like a sound research methodology but is inappropriate where a broad set of capabilities are needed to create rich characters.

Once I chose this approach to building believable emotional agents, I found that it was effective for reasons that I hadn't foreseen. I will provide a number of examples, particularly in Chapters 3 and 5, where this broad, integrated approach to building believable emotional agents has allowed me to incorporate aspects of emotions missed by those who took a narrower view. In particular, I will show how integrating emotion with the perception and motivation systems enables the creation of models of emotions that are often simpler and more complete than the cognitive models of Ortony et al. that I started with.

1. For an exception, see [Sloman86] and [Beaudoin94]. Sloman and Beaudoin hypothesize that states of certain kinds of broad mental architectures can be called "emotional." Beaudoin has looked, in particular, at the relationship of motive processing and emotional states.

2.5 Summary

Here are some of the important issues that have come up in this chapter.

- I described the various sources of inspiration I was able to draw on for this interdisciplinary problem, including art, psychology, and AI. Art provides insights into the nature of the problem and psychology and AI provide ideas about how to solve it.

- I introduced the main goal of the first half of the thesis: providing tools that will assist artists in the creation of believable emotional agents. The tools that I will describe over the next three chapters include the following: the Em architecture, which is a framework for building particular emotional agents; the default Em system, which provides default emotional behavior for artists building specific emotional agents; and discussions about how to use the other two tools effectively. It is useful to think about these tools as a programming language for writing programs to control the emotions of believable agents (the architecture), a sample program (the default system), and a language manual. These tools will be described in Chapters 3 through 5.

- While building these tools, I have taken advantage of the fact that I am working within a broad agent architecture, Tok. By using the other Tok subsystems, it is possible to create characters that have emotions about many things and are able to express their emotions in many ways. I will also demonstrate in the next chapter that this approach enables the creation of models of emotion generation that use perception and motivation as sources of emotions, which can subsequently lead to simpler and more complete models of how emotions are generated. Details about the power gained from a broad architecture will be presented in Chapters 3 and 5.

CHAPTER 3

Emotion Generation

By building on work in the arts, psychology, and AI, I have designed a set of tools that enable the creation of interactive characters that appear emotional. In the next three chapters I will describe these tools in depth and in Chapter 6 I will provide empirical evidence that they are effective.

In Tok agents, the following process creates the appearance of emotions:

1. A trigger event leads to the generation of an emotion structure (or structures).
2. The structure(s) is stored and decays over time.
3. The current set of emotion structures is used to generate additional structures (called behavioral features) that determine abstractly how the agent should act.
4. These features are used to affect specific aspects of the agent's behavior.

For instance, if Tom's boss fires him, that trigger event will lead to the generation of an anger structure directed at his boss. This structure is stored and will eventually decay. In the meantime, this structure leads to the creation of a behavioral feature of type aggression directed at Tom's poor dog Skippy, who just happens to be nearby. This feature leads to specific effects, such as Tom yelling at Skippy and getting red in the face.

In this chapter, I will focus on step 1, the generation of emotion structures based on various inputs.

3.1 Key Ideas in This Approach

While building tools to support the creation of believable emotional agents, I had to make a number of important design decisions. Before describing the tools themselves, I will discuss these decisions and why I took the path I did.

3.1.1 Emotion Architecture vs. Emotion System

Previous work in the area of computational emotions has taken a psychological model of human emotions and put it into code. For example, Elliott's Affective Reasoner [Elliott92] is a fairly true model of the Ortony, Clore and Collins model of human emotions [Ortony88]. Similarly, Frijda and Swagerman's AI work [Frijda87] is based on Frijda's previous theoretical work [Frijda86].

In such systems the goal is to build a working model of a psychological theory and to that end these systems are quite effective. By nature of this goal, however, these systems don't have to provide flexibility to allow the creation of non-realistic agents that don't fit the underlying theory. For instance, let's say an artist wanted to create a very odd agent that gets intensely mad every eight minutes. There is no human emotion model that accounts for such an agent since people like this don't exist, and so most computational models of emotions would not be very helpful for creating such a character.

This is a silly example since artists probably wouldn't want to create such an agent, but as I argued in section 2.2.1, *artists will almost certainly want to create agents that are emotionally non-realistic*. So this artistic constraint has led me to a rather different approach to that taken by Elliott and Frijda and Swagerman.

My basic approach is to separate what I call the emotion architecture from the emotion system. The emotion architecture can be viewed as a language for writing programs to control the emotions of an agent and emotion systems can be viewed as specific programs. Using this analogy, Elliott and Frijda and Swagerman have built programming languages that are only suitable for writing programs that fit a set of underlying psychological assumptions. By separating the emotion architecture from the emotion system, I allow artists to build a greater variety of emotional agents.

In section 3.2, I will describe the part of the Em architecture used for generating emotions; in section 3.3, I will describe the default emotion system. The default system will take the form of a set of specific emotion generators that can be extended, modified, or replaced by artists creating specific characters.

3.1.2 Broad vs. Narrow Emotion Architectures

One useful way to study hard problems in AI and cognitive science is to adopt a divide-and-conquer approach. For instance, many AI systems focus on one aspect of intelligence, such as learning, problem solving, or language understanding, and try to model it quite deeply. This has also been the approach of other researchers in computational emotions—they take the problem of understanding human emotions and study it quite deeply, but mostly independently of other aspects of the human mind.

Even when researchers have considered other aspects of the mind, they have treated those other problems rather shallowly. For example, Elliott's Affective Reasoner [Elliott92] was designed, as its name implies, to reason about emotions. His system incorporates a motivation/action module but this module is not modeled very deeply since his goal is to understand emotions and not motivation. I argued in section 1.4 that when building believable agents it is possible to use somewhat shallow capabilities; the capabilities in Tok, however, are still more developed than similar capabilities in other agents designed for emotion research. For instance, there is a motivation system that is deep enough to handle somewhat complex goal processing. It is not as deep as the more complex goal-processing systems that AI has to offer, but it is deeper than the motivation system in Elliott's Affective Reasoner.

My approach has been to take advantage of the breadth of the Tok agent architecture in emotion generation and expression. This approach has a number of advantages. One advantage is that having greater breadth means that there are more things to have emotions about and more ways to express them. For instance, there is a language generation system, so the agent can have emotions about things like not being able to come up with the right word; the agent can also express emotions through language. Another advantage of this broad approach, that I will come back to in section 3.3, is that *breadth often allows us to create simpler and more complete models of how emotions are generated by using parts of the architecture such as the motivation and perception systems*.

3.1.3 An Explicit Emotion Module

Some work in emotion modeling, such as that of Sloman [Sloman86] and Beaudoin [Beaudoin94], does not use an explicit emotion system to model emotions. In Sloman's model, emotions are states of complex, resource-limited, goal-processing systems. That is, there is no explicit emotion system—emotional reactions emerge from the complex processing of certain kinds of goal-processing systems in certain environments.

Again, my goal is not to figure out how human emotions work and create models of those processes, so it doesn't particularly matter, for my purposes, if Sloman is correct or not. *What concerns me is what kinds of models best enable artists to create quality interactive characters*.

I decided, based on these goals, to use an explicit emotion system that artists can directly access. Although the emotion system is still tightly integrated with the rest of the architecture, by making it independent I provide a degree of modularity that allows artists to worry about many aspects of the character's emotional makeup apart from other parts of the architecture.

Having an explicit emotion system also allows decisions about the character's emotions to be explicit. If emotions are emergent properties of complex goal-processing systems, it is difficult to give an artist direct control over the emotional aspects of the character. By making these decisions explicit, I hope to ease the artistic character-design process.

3.2 The Tools: The Em Architecture

In this section, I describe the part of the Em architecture dedicated to generating emotions. In the next section, I describe the default emotion generators. Throughout both sections I also describe some of the ways that the architecture might effectively be used that the default generators do not take advantage of.

There are two main components to the generational part of the Em architecture. First, there is a set of inputs that the emotion generators are able to access when generating emotion structures. Second, there is the language that is used to write the generators.

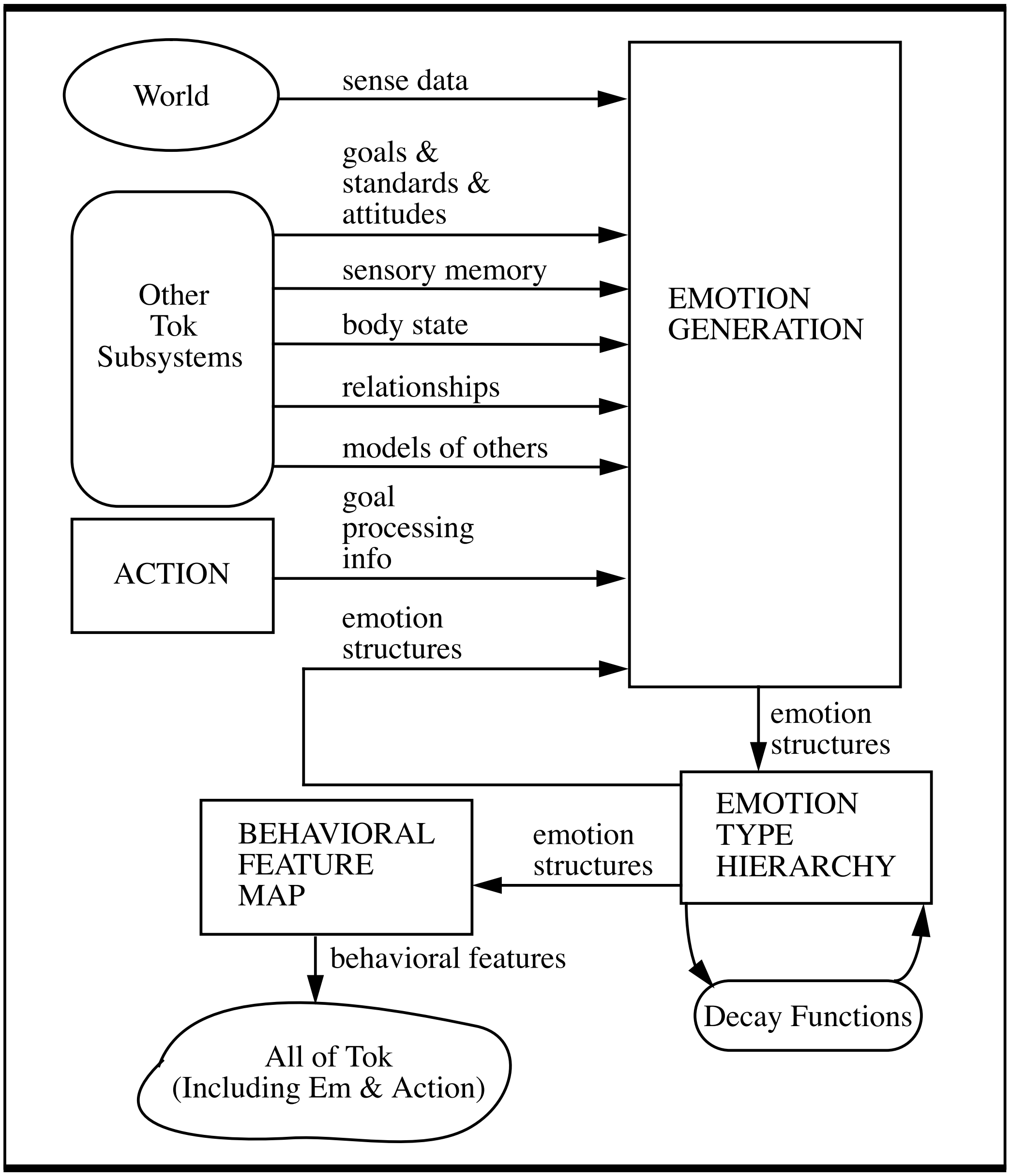
3.2.1 Inputs to the Emotion Generation Rules

Figure 3-1 provides a view of the Tok architecture with an emphasis on the Em component. There are a number of arrows leading into the **Emotion Generation** box. These arrows will be the focus of this section.

- **Sense Data & Sensory Memory**

  Sensing the world can give rise to many sorts of emotions, from the fear associated with noticing a physical threat to the anger that comes from stubbing your toe. Sense data provide important information such as who and what is nearby, what other agents are doing, and what is happening in the environment.

FIGURE 3-1 **An Em-centric View of the Tok Architecture**



Sense data provide different sorts of information depending on the type of physical world simulation. For instance, in the animated Oz system ("The Woggles") sense data encode things like the (x,y,z) position of other agents, the size of other agents, and the current action another agent is performing (e.g., jump, squash). Sense data in the text-based Oz systems encode topographical relationships between objects (e.g., X is on Y, X is in Y), object properties, changes in states of the world, and agent actions (e.g., get-object, speak).

Sensory data are used frequently by the standard emotion generators. The data are used both directly (e.g., loud noises lead to startle emotion structures) and indirectly (e.g., the success of some goal is sensed, which in turn is used to generate a joy emotion structure).

In the text-based Oz systems, emotion generators have access to previous sensory inputs as well as the current sensory inputs.

- **Goals, Standards, and Attitudes**

  These are the basic sources of emotions in the cognitive emotion model of Ortony et al. [Ortony88]. According to this theory, events in the world are appraised relative to the agent's goals; actions of self and other agents are appraised according to a set of standards; and objects are appraised according to attitudes. Variations on these themes lead to a theory of 22 types of emotions (e.g., the shame type covers emotions where the agent disapproves of one of its own actions according to some standard of behavior).

  The goal information flowing into Em can be about either active goals or passive goals. Active goals are those that are actively pursued, like eating. Passive goals are goals that an agent wants to be achieved but that the agent does not actually pursue, such as a fan wanting the Steelers to win the Super Bowl. Both kinds of goals can be important for generating emotions.

  More details about how goals, standards, and attitudes are used in the default Em generators will be provided in section 3.3.

- **Body State**

  I have simple models of the agent's facial expression, state of physical arousal, and muscular state that are available to the emotion generation system. The work of Schacter and Singer [Schacter62] showed that physical arousal can influence the emotional state, even if the arousal isn't actually connected with the apparent cause of the emotion. There are also folk psychology claims that facial expressions can generate emotions—for instance, smiling can make you happier.

- **Social Relationships**

  The social relationships an agent has may be important in generating emotions. For instance, relationships can set up expectations for the behavior of others that lead to emotions—Sue might not normally get mad if someone didn't do what she wanted, but she might get mad if the other person were her employee. Relationships can also be used to heighten or dampen emotion intensities, so you might give a friend the benefit of the doubt and not have a strong reaction, even when that friend does something wrong.

  Stories are often based on relationships (e.g., love stories). In these cases in particular, relationships are important factors in emotion generation. Their role in the default Em generators is, however, only indirect. For instance, if an agent has a goal failure, the inference that determines if another agent was responsible will often take relationship information into account; this, in turn, affects who to get angry at and how intense the emotions structure should be. Relationships happen not to be *directly* used by any of the default Em generators.

- **Models of Other Agents**

  Modeling other agents is an important topic in Part II of this thesis. One reason to model another agent is for emotional purposes. For example, the gunman in *Robbery World* models how violent he believes the police officer is and his fear of the officer is dependent on this model. If he believes the officer is very violent, he will be more scared.

  Some emotional theories, such as that of Ortony et al. [Ortony88], rely on modeling the plans and goals of other agents in order to appraise their emotional states. Em's default generators can use agent models as inputs, though I have found this kind of modeling to be unnecessary for some characters and worlds. I will discuss my approach in more detail throughout section 3.3.

- **Goal Processing Information**

  Having information about the current goals is important, as I just discussed, but it can also be important to have information about the processing of goals. Some previous systems (e.g., [Elliott92]) have generated emotions based only on static goal information, but my experience has shown that having dynamic processing information can also be useful.

  Em has access to information about goal creations, goal successes, goal failures, the likelihood of goals succeeding, the likelihood of goals failing, changes in either likelihood value, the parties responsible for a goal succeeding or becoming more likely to succeed, the parties responsible for a goal failing or becoming more likely to fail, the sources of threats to goals, possible sources of assistance for goals, and plan failures.

- *Goal creations, goal successes and goal failures* are all determined during normal Hap processing. Every time one of these events occurs, a method call[1] is made to record the event for processing by Em. This method stores the goal in question in a slot based on the type of event (creation/success/failure). Default Em generation rules notice goals being added to these slots.

- *Likelihood of goal success and failure* are determined by functions associated with individual goals. I extended the Hap language to allow artists to write such functions as they write the agent's goals. These functions can include arbitrary amounts of inference when necessary. For example, the gunman's goal not to be killed requires inferencing about a number of aspects of the situation (is there a police officer present? does the officer seem violent? does anyone else have a gun? etc.).

  These likelihood functions are continuously updated and the values stored with the goal. When the value changes, the amount of change (delta) is also stored. Emotion generators can match on either the current likelihood value or the delta value. There are default emotion generators that match both of these values.

  Likelihood functions also return information about the possible threats or aids to the goal. This information is also stored with the goal and is used in default Em generators.

  Here's an example goal with accompanying likelihood functions. For simplicity, I will often abbreviate "likelihood of success" as "los" and "likelihood of failure" as "lof." `compute-los` and `compute-lof` are the functions to compute how likely this particular goal is to succeed and fail. They return both a value, from 0 1o 1, and a list of agents to assign credit or blame to (this list can be empty). The importance of this goal succeeding and not failing is 5 on a scale of 0 to 10—this can be a function as well as a static value. The goal itself is `get-card` and is on the last line of the example; all the other code represents various annotations on this goal. The keyword *with* is used in Hap to denote a goal with annotations.

1. Oz uses an object-oriented approach to modeling the simulated world and the agents, so I will occasionally refer to the methods and slots of various objects. This will give the reader a feel for the specific implementation but is not meant to argue that this particular implementation is unique or ideal.

```
(with
   (importance 5)
   (compute-los (lambda () (list 3/4 $$actor)))
   (compute-lof (lambda () (list $$get-card-lof $$actor)))
   (subgoal get-card $$card $$actor))
```

Melvin (in *The Playground*) uses a version of this goal when he's trying to get a card from another character after a trade has been agreed to. `$$card` and `$$actor` are instantiated variables that are previously bound to the card and actor in question. In this case Melvin always feels that there is a 75% chance of things going well, but a varying belief about how likely things are to fail. (Note that success could just be (1 - chance-of-failure) but I chose this example to show that this does not have to be the case.)

In this example, the chance of failure is determined by the value of a variable, `$$get-card-lof`, that is initially set to 0 but that can be changed at various times during the processing of the `get-card` behavior—such as when Melvin has asked for the card and received no response. In other cases, this variable can be replaced with a more complex inference process to determine the value, so it is possible to take both information about the environment and information about the on-going behavior progress into account in determining likelihood values.

- Goals in Hap can also be annotated with inference functions for determining *whom to blame when they fail and whom to give credit to when they succeed*. For instance, here is the same `get-card` goal as above with new annotations (`failure-responsibility` and `success-responsibility`) to denote that `$$actor` is responsible in case of either a success or a failure.

```
(with
   (importance 5)
   (compute-los (lambda () (list 3/4 $$actor)))
   (compute-lof (lambda () (list $$get-card-lof $$actor)))
   (failure-responsibility (lambda () (list $$actor)))
   (success-responsibility (lambda () (list $$actor)))
   (subgoal get-card $$card $$actor))
```

- *Plan failures* are noticed during normal Hap processing. Structures representing failed plans that have a non-zero *importance* will be stored in a pre-determined slot in the agent object that the emotion generators can access.

- **Emotion Structures**

  The Emotion system feeds back on itself, so that previously generated emotion structures can affect the generation of new structures. The default set of Em generators do not happen to use emotion structures as inputs.

3.2.2 Writing Emotion Generators in Hap

Given the inputs just described, artists need to write emotion generators that map these inputs into emotion structures. To do this, they use a version of the Hap language [Loyall93,Loyall96] extended to better support emotion generation. The default set of emotion generators are, of course, also written in Hap.

The Hap language was originally designed to write robust, reactive physical behaviors. It has turned out, however, to also be a reasonable language in which to write emotion generators. Here are some of the features of Hap that make it a good language for creating emotion generators. I will discuss a sample emotion generator for *frustration* later in the chapter, but it provides a simple example of the first three items in this list as well. The Hap code for this generator can be found in Figure 3-3 (pseudo-code for the generator can be found in Figure 3-2).

- **Demons.** The ability to have rules that fire in certain circumstances is very important for creating emotion generators. Every generator I have written is expressed as a demon that waits for a particular emotional situation to occur, and then fires, creating an emotion structure.

- **Flexible match language.** The *match language* is used to code the left-hand sides of the generation rules. That is, it is used to describe the situations in which the generator should fire. It is not always simple to express these situations. I've used a good deal of the flexibility provided by Hap in writing the left-hand sides of the default generation rules. Hap's match expressions are very rich, allowing arbitrary computation.

- **Internal actions.** Hap was designed for controlling external, physical action, but it also allows internal, mental actions to be performed. This is important because the generators don't have any external effects—they just create emotion structures, which is a purely internal process.

- **Flexible control of actions.** Hap also provides a number of standard programming language structures, like if-then and cond, which can often make the internal behaviors that generate the emotion structures simpler to write.

- **Priorities.** Most of the Tok subsystems (with the exception of low-level sensing and some higher-level perception) are written in Hap. Each subsystem has goals and behaviors appropriate to its role in the agent. For instance, the emotion system has goals to update the emotional state and to decay emotions at regular intervals; the motivation system provides what are typically called the “goals” of the agent—things like eating when hungry and sleeping when tired. Each of these goals has a priority that determines when it gets processed relative to the other goals. This allows the artist to make sure things get done in the proper order.

Here are a couple of examples of the kinds of things that can be done with priorities:

- Priorities allow emotions to be updated before actions or vice versa, depending on the particulars of the system. I have used both approaches.

  In the text-based, non-real-time systems, emotions are updated first and then actions are generated. This ensures that the action chosen reflects the current emotional state. Since this is not a real-time system, the agent is allowed to process until an action is chosen.

  In the animated, real-time system, actions are generated with a higher priority than emotions. For example, a woggle jumping through the air sees another woggle about to attack it. Since the agent is mid-air, it typically has time before it has to choose its next action, so there is no goal conflict to be resolved by priorities. In this case, the emotions are updated and when the agent is about to land and needs to choose its next action, the proper emotional action is chosen.

  However, if the threat occurs very late in the jump, when the agent is considering its next action, it is possible that the next action is chosen before the emotional state is updated. Once the emotions are updated, however, if the action chosen to be performed next has not yet started (the previous jump has not quite ended), that action will be retracted and a new one chosen in its place.

  This arrangement means that the agent always has an action to perform, and that it is almost always the appropriate emotional action. In some cases, the emotion might not be expressed immediately, but the agent is never frozen waiting for the mind to choose the next action, which I have found to be crucial for subjective believability. Horswill also provides some evidence that this is true [Horswill94].

  This is an interesting point because it means that even if there were a perfect theoretical model of emotion, if it couldn’t be computed fast enough it wouldn’t be suitable for real-time believable agents (putting aside for the moment the argument that artists probably don’t want theoretically perfect models anyway). By building an emotion system within a reactive language, the artist can make sure that the emotions don't come at the expense of the agent being reactive to the environment.

- Using priorities also allows the agent to update "more important emotions," like fear and anger, before "less important emotions," like gratification and pity. Artists are able to specify which emotions are more or less important for each character if desired.

- Priorities let artists ensure that inferences that are necessary for emotion processing are done before the emotion processing. For instance, inferences about how likely a given goal is to fail need to come before the emotional generation of distress because of that change in likelihood.

These priority examples demonstrate how I achieved some types of integration of the various capabilities of a Tok agent (e.g., emotion, action, inference) and also how such integration can be useful. For more on this subject, see [Loyall96].

3.3 The Tools: Em's Default Emotion Generation Rules

Em comes with a standard library of emotion generation rules based mostly on the Ortony, Clore and Collins cognitive emotion model [Ortony88]. (For brevity, I will refer to this as the OCC model.) When designing a specific agent, artists can use these default rules in their entirety, or they can pick and choose those that fit their agents. Artists can also add to and modify the rules that are provided. My goal is to provide artists a useful set of starting rules without forcing a particular emotion system on all characters.

In this section, I will describe how my work relates to the OCC emotion model, I will show what an emotion generator looks like, and I will describe the set of emotion generators that I have created.

3.3.1 Relationship to the OCC Model

It should be kept in mind that while the Em emotion generators are based on the OCC model, many of them are somewhat different from their OCC counterparts. A difference that occurs in all of the rules is that they don't represent the complex intensity system of the OCC model. The original model postulated four global intensity variables (arousal, unexpectedness, proximity, and sense of reality) and up to four local variables (e.g., goal importance in goal-related emotions such as joy) for each emotion type. Elliott and Siegle [Elliott93] continued to expand and refine the complex emotion intensity model to include 24 emotion intensity variables (such as physical well-being)—and that was claimed to be an incomplete list.

Instead of using the complexity of the OCC or Elliott and Siegle models, I have adopted a subset of the intensity variables of these models. I found that my simpler intensity model was sufficient for all of the characters the Oz group has built so far[1] and was easier to deal with from an agent builder's standpoint. It may be that the complexity of the OCC/Elliott model is necessary for some characters in some worlds, so Em allows new, more complex emotion generators to be added by the artist. Making complex intensity computations the default that artists would have to learn, though, is unreasonable and unnecessary.

Another way that the Em emotion models differ from the OCC models is that Em uses a broad agent architecture to move away from strictly cognitive models of how emotions are generated. The OCC model has a very cognitive slant that requires complex reasoning and modeling of other agents. In many cases, I have been able to come up with much simpler techniques for modeling the same types of emotions.

Em takes advantage of the breadth of the Tok agent architecture to incorporate motivational and perceptual antecedents into the emotion generators. I am not rejecting cognition by doing this, just extending the Em models rely on more than cognition.

Here's an example of what I mean. In the OCC model pity emotions are generated roughly as follows: agent A feels pity for agent B when (1) agent A likes agent B and (2) agent A appraises an event in the world to be displeasing to B according to B's goals. So, if Alice hears that Bill got a demotion, Alice must be able to match this event with a model of Bill's goals, including goals about demotions. This requires Alice to have a pretty good model of Bill's goals and appraisal functions, which is very hard to do—especially in dynamic worlds where goals can change very rapidly.

Instead, I suggest the following antecedent for pity: agent A feels pity for agent B when (1) agent A likes agent B and (2) agent A thinks that agent B is unhappy. Not only is this a simpler description, which can be important if non-technical people are to use this system, but it is also more complete and sometimes even easier to use.

In this case, I have broken the OCC model into two components: recognizing sadness in others and having a sympathetic emotional response. In the OCC model, both components are tied up in the antecedent for pity. Recognizing sadness in others is done, according to the OCC model, only through reasoning and

1. Eleven so far, in five different simulated worlds.

modeling of the goals of other agents, so this inference can be built into the model of how the emotion is generated. Em keeps the recognition of sadness apart from the emotional response, which allows for multiple ways of coming to know about the emotions of others. One way is to do reasoning and modeling, but another way, for example, is to see that an agent is crying.

The Em model is more complete than the OCC model in cases such as agent A seeing that agent B is sad but not knowing why.[1] In the OCC case, when agent A does not know why agent B is unhappy, the criteria for pity is not met.[2] Because the default Em emotions generators require only that agent A believe that agent B is unhappy, which can be perceived in this case, Em generates pity.

It is possible in Em for A to know about B's emotional state through complex reasoning about B's goals, but this isn't necessary. In fact, in the simulations I have built, I have found the perceptual version of this emotion (i.e., seeing B is unhappy) is typically more useful than the cognitive version (i.e., reasoning about B's emotional state). *This means that for many characters, it is possible to dispense with the cognitive modeling code and rely only on the simpler perceptual code*.[3]

Here's another example of how I have been able to create less-cognitive models of how emotions are generated. According to the OCC model, distress is generated when an event is appraised to be unpleasant relative to the goals of the agent. This model relies on evaluating external events. In Em, distress is caused by goals either failing or becoming more likely to fail, which is determined by the motivation/action system. This shifts the emphasis towards the goal processing of the agent and away from the cognitive appraisal of external events. This is useful for two reasons. First, the motivation system is already doing much of the processing (e.g., determining goal successes and failures), so doing it in the emotion system as well is redundant. Second, much of this processing is easier to do in the motivation system since that's where the relevant information is. For instance, deciding how likely a goal is to fail might depend on how far the behav-

1. How an agent notices that another agent is sad will depend on the specifics of the characters and system. For instance, in the animation-based *Edge of Intention*, agents believe other agents are sad when they move slowly, squash down a lot, and don't jump very high. In *The Playground*, it would be a matter of noticing actions, such as sulking or crying, or facial expressions, like frowning. Recognizing the emotions of human-controlled characters relies on similar techniques.

2. It is possible to create an implementation of the OCC model that reasons that a sad-looking agent must have appraised an unknown event relative to an unknown goal and then generate emotions based on the goal appraisal. This approach is more complex than is necessary.

3. The fact that perception is easier than cognition is partly due to the fact that I am working in simulated environments where perception is fairly straightforward. In real-world systems, perception becomes more difficult, but cognition doesn't get any easier.

ior to achieve that goal has progressed or how many alternate ways to achieve the goal are available—this information is already in the motivation system.

By using the breadth of the Tok agent architecture, Em uses models of how to generate emotion structures that are often more complete and easier to use than their strictly cognitive counterparts. This idea of moving away from cognition and towards a hybrid model that incorporates perception and motivation as well as cognition will be repeated throughout the discussion of the individual emotion types.

3.3.2 An Emotion Generator

In the next section, I will describe the default emotion generators I have built somewhat abstractly—in other words, I'm not going to provide a lot of code. The goal is to present what the generators do, not to provide details of the implementation. Nonetheless, it is useful to see what an emotion generator looks like and how it works to get a feel for the process involved in creating emotion generators. Figures 3-2 and 3-3 provide pseudo-code and Hap code for the frustration generator. Recall that emotion structures have a type (e.g., anger), a cause (e.g., being insulted), a direction (e.g., the agent who made the insult), and an intensity (e.g., 7 out of 10).

Readers interested in the details of Hap are referred to [Loyall93, Loyall96]. Otherwise, it is not necessary to understand the Hap code and I suggest looking at the pseudo-code instead.

Frustration is generated when the agent has an important behavior that fails. For instance, it might be really important that an agent not only achieve some goal, but that the goal be attained in some specific manner. If the desired approach to achieving the goal fails, the agent may still be able to achieve the goal, but feel frustrated that the desired approach was unsuccessful.

This generator relies on the fact that the agent architecture records the failure of behaviors in a specific place in memory, which is one of the inputs to Em discussed earlier.

In the frustration example, the emotion generator requires only a single Hap production and the production generates only frustration. This is not always the case as the rules are often keyed off of important events, like goal successes, that can lead to a number of different kinds of emotions.

FIGURE 3-2 **An Emotion Generator (Pseudo-Code): Frustration**

```
Demon:
 Name: em-update-frustration-demon
 if (and (not-empty(plan-failures)))
         (B := first-elmt(plan-failures))
         (Importance(B) > 0))
 then {
   struct = make-emotion-structure {
             type = FRUSTRATION
             cause = B
             direction = NIL
             intensity = Importance(B)
            };
   store(struct);
   remove(B,plan-failures);
 }
```

FIGURE 3-3 **An Emotion Generator (Hap Code): Frustration**

```
;; Production to generate frustration structures
(sequential-production update-frustration ()

   ;; This is a demon
   (demon em-update-frustration-demon

        ;; LHS
        ;; Fire when a failed behavior has been put in
        ;; the $plan-failures slot and the importance of the
        ;; behavior is greater than 0
        (and (match $plan-failures
                    (list-containing ?plan))
             (match (call importance ?plan) ?intensity)
             (> ?intensity 0)

             ;; Create an emotion structure. Set the variable
             ;; ?emotion-structure to the structure
             (match (make frustration-emotion
                          actor self
                          cause ?plan
                          frustration-production ?intensity)
                    ?emotion-structure))

        ;; RHS
        ;; Store the structure
        (mental-act
           (call add-emotion
                 (slot emotion-type-hierarchy $em)
                 $$emotion-structure
                 'frustration))

        ;; Remove the behavior from the $plan-failures slot
        (mental-act
           (setf $plan-failures
                 (remove $$plan $plan-failures)))))
```

3.3.3 The Default Emotion Generators

Some of the Em emotion generators are broad and cover more emotion types than might be obvious from their names. For instance, distress covers the general class of emotion structures caused by goals failing or becoming more likely to fail. There are numerous subtypes that are represented by this general class: grief is a type of distress caused by the failure of goals that loved ones not die; despair is a type of distress with an high intensity. So, if you don't see your favorite emotion on this list, it's possible that it's just an instance of one of the more general classes listed.

Table 3-1 provides a quick overview of the emotion types. Remember, I am not claiming that these are "correct" in any psychological sense—just that they are a strong starting point for creating believable emotional agents. There are a number of choices that I've made along the way, such as how to compute the intensity of various emotion types, that are justified only by their artistic impact. Chapter 6 will present evidence that characters created based on these choices can appear emotional and be believable.

TABLE 3-1 Emotion Generation in Em

Emotion Type	Cause in Default Em System
Distress	Goal fails or becomes more likely to fail and it is important to the agent that the goal not fail.
Joy	Goal succeeds or becomes more likely to succeed and it is important to the agent that the goal succeed.
Fear	Agent believes a goal is likely to fail and it is important to the agent that the goal not fail.
Hope	Agent believes a goal is likely to succeed and it is important to the agent that the goal succeed.
Satisfaction	A goal succeeds that the agent hoped would succeed.
Fears-Confirmed	A goal fails that the agent feared would fail.
Disappointment	A goal fails that the agent hoped would succeed.
Relief	A goal succeeds that the agent feared would fail.
Happy-For	A liked other agent is happy.
Pity	A liked other agent is sad.
Gloating	A disliked other agent is sad.
Resentment	A disliked other agent is happy.
Like	Agent is near or thinking about a liked object or agent.
Dislike	Agent is near or thinking about a disliked object or agent.

Emotion Type	Cause in Default Em System
Other attitude-based emotions	Agent is near or thinking about an object or agent that the agent has an attitude towards (e.g., awe)
Pride	Agent performs an action that meets a standard of behavior.
Shame	Agent performs an action that breaks a standard of behavior.
Admiration	Another agent performs an action that meets a standard of behavior.
Reproach	Another agent performs an action that breaks a standard of behavior.
Anger	Another agent is responsible for a goal failing or becoming more likely to fail and it is important that the goal not fail.
Remorse	An agent is responsible for one of its own goals failing or becoming more likely to fail and it is important to the agent that the goal not fail.
Gratitude	Another agent is responsible for a goal succeeding or becoming more likely to succeed and it is important that the goal succeed.
Gratification	An agent is responsible for one of its own goals succeeding or becoming more likely to succeed and it is important to the agent that the goal succeed.
Frustration	A plan or behavior of the agent fails.
Startle	A loud noise is heard.

- **Distress**

  Distress is generated when a goal fails or the agent believes that a goal has become more likely to fail and it is important to the agent that the goal not fail. The intensity is based on the importance of the goal not failing and the change in the likelihood of failure. (Failure means the goal has changed to a likelihood of failing of 1.) For example, suppose a goal with importance of not failing of 6 and a believed likelihood of failing of 25% actually fails. A distress structure of intensity .75 x 6 = 4 (when rounded down) is generated.

  The factors affecting emotional intensity are taken from the OCC model, though I have taken only some of their factors into account. Choosing to multiply the various factors and truncating the decimal part has proven to be an effective way of computing the overall intensity. The OCC model does not posit an explicit model of how the various factors should be combined. Artists are free to adjust these calculations as they see fit.

Importance of goals[1] is broken down into two parts: importance that the goal succeed and importance that the goal not fail. For instance, imagine Jake hates being late for appointments; being on time doesn't make him happy, but being late does make him upset. This is modeled in Em by giving Jake's goal to be on time a low (or 0) importance of success and a medium-high importance of failure. There might be other ways to explain these sorts of emotional reactions, but I have found this approach simple and effective.

In the OCC model, distress emotions occur when an event is appraised as being unpleasant with respect to the agent's goals. This model makes it hard to model cases where the goal is known to have failed (or become more likely to fail), but the causing event is unknown. By placing emphasis on goal processing instead of the appraisal of events it is much easier to model certain kinds of distress.

For example, Toby owns an expensive vase and comes home one day to find it broken. In this case, Toby presumably has a goal that his expensive vase not be broken; the success and failure of this goal can be determined by simply sensing the world. When Toby sees the broken vase, he knows that his goal has failed, but does not know what specific even caused the failure (e.g., the cat knocked it off the shelf, Greg and Peter were playing ball in the house, or the wind blew it off the shelf). The Em rules handle this case easily; just knowing that an important goal has failed is enough. In the OCC model, a generic vase-breaking event must be created by the agent for the sole purpose of appraising it in a negative way, which leads to unnecessary cognitive processing for creating believable emotions.

- **Joy**

  Joy is generated when a goal succeeds or the agent believes that such a goal has become more likely to succeed and it is important to the agent that the goal succeed. The intensity is based on the importance of the goal and the change in the likelihood of success. (Success means the goal has changed to a likelihood of succeeding of 1.) For example, if an agent has a goal with an importance of succeeding of 8 and that goal goes from 30% likely to succeed to 55% likely to succeed, the agent feel joy of intensity .25 x 8 = 2.

1. Goal importance is used for emotion generation and it is not directly used for driving action; there is a separate priority value for determining the processing order for goals. Priorities, however, can change based on emotions as I will demonstrate in Chapter 5.

In the OCC model, joy emotions occur when an event is appraised as being pleasant with respect to the agent's goals. The Em model focuses on the agent's goal processing instead, which leads to less cognitive processing. This relationship is analogous to the one between the OCC and Em versions of distress.

- **Fear**

Fear is generated when a goal is considered to be likely to fail and it is important to the agent that the goal not fail. The intensity is based on how important the goal is and how likely it is to fail. For instance, a goal with importance of not failing of 4 that is believed to be 75% likely to fail, leads to fear of intensity .75 x 4 = 3. If the goal fails, no fear structure is generated.

There is actually one more condition for the generation of a new fear structure: there should not already be a fear structure present for the goal in question. If the goal in the example were to stay at 75% likely to fail, there would not be a new fear structure generated each tick; if the likelihood of failure were to increase, the intensity of the existing fear structure would rise; if the intensity were to decrease, the intensity would begin to decay as will be described in the next chapter.

Fear differs from distress in that distress is based on changes in the likelihood of failure, so if the goal is staying at a steady likelihood of failure, the distress caused by the initial increase will decay, but fear will not since the underlying cause of the fear is still present.

In the OCC model, fear occurs when a prospective event is appraised as being unpleasant relative to the agent's goals. This model requires some sort of forward-looking simulation in order to have events that can be appraised. The Em rules could use such a system on which to base appraisals of likely goal failures, but they aren't necessary. For example, in *The Playground*, Melvin has a health-preservation goal with a likelihood of failure function that believes the goal is somewhat likely to fail when Sluggo is around and very likely to fail when Sluggo is around and looking angry. This doesn't require any forward simulation of how Sluggo would go about hurting him and is, therefore, much simpler to create. Furthermore, the Em model makes it easy to model irrational fears of goals failing; even if the agent is not capable of imagining an event that would cause such a failure, fear can exist.

This is another example of the shift away from cognition based on appraisals of external events and towards a model that focuses on internal goal-processing events. As with distress and joy, the focus shift here is away from appraisals of (possible future) events and towards goal processing information (that can be based on appraisals or simple perceptions of the world).

- **Hope**

  Hope is generated when a goal is considered to be likely to succeed and it is important to the agent that the goal succeed. The intensity is based on how important the goal is and how likely it is to succeed. For instance, suppose a goal with importance of succeeding of 5 has a 90% believed likelihood of succeeding. This leads to hope of intensity .9 x 5 = 4 (when rounded down). No hope is generated if the goal succeeds. Also, if the likelihood of success stays steady, no new emotion structures are generated; the first structure generated is simply kept and not decayed.

  In the OCC model, hope occurs when a prospective event is appraised as being pleasant relative to the agent's goals. The relationship between the OCC and Em models of hope is analogous to the relationship between the OCC and Em models of fear.

- **Satisfaction**

  Satisfaction is generated when a goal that was likely to succeed does and it was important to the agent that the goal succeed. The intensity is based on the importance of the goal succeeding and the likelihood of success prior to success. For instance, if a goal with importance of succeeding of 8 is believed to be 75% likely to succeed and then does succeed, the satisfaction will be of intensity .75 x 8 = 6.

  In the OCC model, satisfaction occurs when a pleasing prospective event occurs. This requires some sort of idea about what prospective events would be pleasing. The Em model replaces the idea of external events with internal goal processing information, so satisfaction occurs when a goal succeeds that was hoped would succeed. The initial hope does not depend on anticipating actual external events (as just discussed for hope and fear) and the success of the goal does not depend on the appraisal of external events (as discussed with joy and distress), though in both cases appraisals of events can be involved.

  The OCC model also includes the concept of how much effort was expended towards achieving a goal. This is used to affect the intensity of the emotion, so working hard to achieve a goal leads to more intense satisfaction when it is met. This is not modeled in the default Em generators, but it is possible to write emotion generators that do take it into account if that is desirable for a certain character. This is done by keeping an *effort-expended* variable with each goal that can be incremented during the behavior when some amount of work has been expended; then the satisfaction generator will need to take this new variable into account when computing the intensity of the emotion structure.

- **Fears-confirmed**

  Fears-confirmed is generated when a goal that was likely to fail fails and it was important to the agent that the goal not fail. The intensity is based on the importance of the goal not failing and the likelihood of failure prior to failing. If a goal with importance of not failing of 7 is 80% likely to fail and then does fail, the fears-confirmed will be of intensity .8 x 7 = 5 (rounded down).

  In the OCC model, fears-confirmed occurs when a displeasing prospective event occurs. The relationship between the OCC and Em models of fears-confirmed is similar to the relationship between the two versions of satisfaction.

- **Disappointment**

  Disappointment is generated when a goal that was likely to succeed fails and it was important to the agent that the goal succeed. The intensity is based on the importance of the goal succeeding and the likelihood of success prior to failing. If a goal with importance of succeeding of 3 is 90% likely to succeed and then fails, the disappointment will be of intensity .9 x 3 = 2 (rounded down).

  In the OCC model, disappointment occurs when a pleasing prospective event fails to occur. The relationship between the OCC and Em models of disappointment is similar to the relationship between the two versions of satisfaction.

- **Relief**

  Relief is generated when a goal that was likely to fail succeeds and it was important to the agent that the goal not fail. The intensity is based on the importance of the goal not failing and the likelihood of failing prior to succeeding. If a goal with an importance of not failing of 7 is 45% likely to fail, but it succeeds, the relief will be of intensity .45 x 7 = 3 (rounded down).

  In the OCC model, relief occurs when a displeasing prospective event fails to occur. The relationship between the OCC and Em models of relief is similar to the relationship between the two versions of satisfaction.

- **Happy-for**

  Happy-for is generated when an agent likes another agent who is happy. The intensity is based on how much the other agent is liked and how happy that agent is believed to be. If an agent is liked[1] to a degree 8 (out of 10) and that agent is believed to be happy at an intensity of 5, the happy-for emotion will be of intensity 8 x 5 / 10 = 4.

  The relationship between the OCC version of happy-for and the Em version is very similar to that of pity which was discussed earlier. In the OCC model, happy-for occurs when an event is appraised as being pleasing to a liked agent. This does not account for the emotion associated with seeing a friend who is happy, but not knowing why that character is happy. The Em model only requires some appraisal about the happiness of the other agent. This can come about through inferences about how that agent might appraise certain events, but is not limited to that case. For example, simple physical cues can be used. In fact, physical cues might be preferred in most cases in which they are available since inferences are often more likely to be wrong than direct perceptions (at least in simulated worlds).

  The OCC model also has a concept of deservingness. If a friend receives a tremendous, but undeserved, award, happy-for may not arise. The default Em rules ignore this case, though it might be necessary to add in the future.

  It should also be noted that besides the emotions happy-for, pity, resentment, and gloating, it could be useful to add other emotions about other agents. For example, fear-for could be an emotion generated when an agent believes that a liked agent is likely to have an important goal fail. Although fear-for (and similar emotions) are not part of the OCC model or the default Em emotion system, it is possible to add them for particular characters if they are needed.

- **Pity**

  Pity is generated when an agent likes another agent who is distressed. The intensity is based on how much the other agent is liked and how unhappy that agent is believed to be. If an agent is liked at a level 6 and they are believed to be unhappy at an intensity of 9, the pity felt towards that agent will be of intensity 6 x 9 / 10 = 5 (rounded down).

  The relationship between the OCC and Em versions of pity was discussed above and is similar to the relationship between the OCC and Em versions of happy-for just discussed.

1. Like (and dislike) are attitudes. They are reasonably stable feelings about other agents that can give rise to emotions, but are not emotions themselves.

- **Gloating**

  Gloating is generated when an agent dislikes another agent who is unhappy. The intensity is based on how much the other agent is disliked and how unhappy that agent is believed to be. For instance, if an agent is disliked 6 and is thought to be distressed at intensity 5, then the intensity of the gloating will be 5 x 6 / 10 = 3.

  In the OCC model, gloating occurs when an event is appraised as being displeasing to a disliked agent. The relationship between the OCC and Em versions of gloating is very similar to the relationship between the OCC and Em versions of happy-for and pity.

- **Resentment**

  Resentment is generated when an agent dislikes another agent who is happy. The intensity is based on how much the other agent is disliked and how happy that agent is believed to be. If a disliked (degree 8) agent is happy (intensity 7), then resentment towards that agent will be of intensity 8 x 7 / 10 = 5 (rounded down).

  In the OCC model, resentment occurs when an event is appraised as being pleasing to a disliked agent. The relationship between the OCC and Em versions of resentment is very similar to the relationship between the OCC and Em versions of happy-for and pity.

- **Like, dislike, and other attitude-based emotions**

  Like, dislike, and other attitude-based emotions are generated when the agent comes in contact with or thinks about any object (including other agents) that the agent has an attitude about. For example, if John likes Sara (an attitude), then he will have an emotional reaction (also called "liking") when she's around him. By default, Em's rules match against all objects in sight, but other behaviors can easily filter this list or add new objects to it for consideration. In such cases, new emotion generators would not have to be written.

  The intensity is based on the strength of the attitude. So, if John has a liking attitude of degree 8 towards Sara, then when she is around he will have a liking emotion of intensity 8.

  The OCC model postulates only two attitude-based emotions: love, caused by appraising an object pleasing according to a like attitude, and hate, caused by appraising an object displeasing according to a dislike attitude.[1] The Em system allows this set to be extended just by creating new attitudes—no new generators need to be written. For example, agents may feel attitude-based fear just by being around another agent, even if there is no particular goal being threatened. Melvin feels attitude-based fear (and goal-based fear) towards Sluggo in the playground simulation.

- **Pride**

  Pride is generated when an agent finds its own action praiseworthy according to some standard of behavior. Standards can be of moral quality: thou shalt not kill. Or they can be of performance quality: I should be able to do well in school.

  The intensity is based on the strength of the standard. So, if an agent has a strength 6 standard to do well on a test and it does so, the agent will feel pride at intensity 6. The strength of the standard is set by the artist and should be based on the difficulty of fulfilling the standard and the personality of the character.

  Like the goal-based emotions, standards have two kinds of importance: that they be met and that they not be broken. For example, a standard to not steal is important not to break, but an agent shouldn't feel pride every time it fails to steal something. Similarly, an agent might feel pride at some exceptional accomplishment, but not feel ashamed at not accomplishing extraordinary feats.

  The OCC model is quite similar, although the idea of two kinds of importance is never made explicit in their theory.

- **Shame**

  Shame is generated when an agent finds its own action blameworthy according to some standard of behavior. The intensity is based on the strength of the standard. Breaking a strength 5 standard results in shame of intensity 5.

  The OCC model is quite similar.

- **Admiration**

  Admiration is generated when an agent finds another agent's action praiseworthy according to some standard of behavior. The intensity is based on the strength of the standard. If another agent is judged to have met a standard of strength 7, admiration towards that agent of intensity 7 will be generated.

  The OCC model is quite similar.

1. I have decided to use the words "like" and "dislike" for both attitudes and emotions instead of using "like" and "dislike" for attitudes and "love" and "hate" for emotions. The confusion of reusing terminology, exists, but "love" and "hate" had inappropriate connotations that I didn't want either. I will try to be clear about whether I am talking about emotions or attitudes if there is some chance of confusion.

- **Reproach**

  Reproach is generated when an agent finds another agent's action blameworthy according to some standard of behavior. The intensity is based on the strength of the standard. Another agent breaking a standard of strength 3 will cause reproach of intensity 3.

  The OCC model is quite similar.

- **Anger**

  Anger is generated when another agent is considered to be responsible for a goal failing or becoming more likely to fail and it is important to the agent that the goal not fail. The intensity is based on the importance that the goal not fail and the degree to which the other agent is held to be responsible. If agent A has a goal fail and that goal has an importance of not failing of 6 and agent A believes that agent B is fully responsible for the failure, then agent A will feel anger towards B of intensity 100% x 6 = 6.

  Because of the goal failure and because of an implicit standard for others to not cause other people's goals to fail, Em also generates distress and reproach when it generates anger. When computing anger intensities, the generator does not factor in how likely the goal was to fail beforehand; this gives an edge in intensity to anger over distress as the intensity of distress is based on the change in likelihood of failing.

  The OCC model describes anger as a compound of distress and reproach. That is, when another agent performs an act that is blameworthy and displeasing, anger is generated towards that agent.

  Here's an example demonstrating how the Em and OCC models work. A basketball player playing on Doug's favorite team takes a bad shot and misses the final shot of the game causing the team to lose. The OCC model posits that Doug's reproach feeling towards the player for taking a bad shot and his distress at the team losing combine to form anger towards the player. In Em, the goal of the team winning the game fails, and the responsibility is assigned to the player taking the bad shot, which also results in anger, distress, and reproach.

  According to Elliot's implementation of the OCC model [Elliott92], when reproach and distress are combined to form anger, the two component emotions are removed from the system. Em keeps the distress, reproach, and anger emotion structures. The emotion expression part of the architecture decides which (possibly more than one) to display.

- **Remorse**

  Remorse is similar to anger, but the agent is responsible for its own goal failing or becoming more likely to fail. The intensity is based on the importance of the goal not failing and the degree to which the agent holds itself responsible.

  In the OCC model, remorse is a compound emotion with shame and distress as its constituents. In remorse situations, Em will also generate distress and shame.

- **Gratitude**

  Gratitude is generated when another agent is responsible for a goal succeeding or becoming more likely to succeed. Intensity is based on the importance of the goal succeeding and the degree to which the agent is held responsible.

  In the OCC model, gratitude is a compound emotion with admiration and joy as its constituents. In gratitude situations, Em will also generate joy and admiration.

- **Gratification**

  Gratification is similar to gratitude, but the agent is responsible for its own goal succeeding or becoming more likely to succeed. The intensity is based on the importance of the goal succeeding and the degree to which the agent holds itself responsible.

  In the OCC model, gratification is a compound emotion with pride and joy as its constituents. In gratification situations, Em will also generate joy and pride.

- **Frustration**

  Frustration is generated when an agent's behavior fails, even when there is no corresponding goal failure. This means that the agent's goal may not have changed in how likely it is to succeed or fail, but the temporary setback can still be emotional. For example, when hammering in a nail, I typically take about three bent nails before I accomplish my task. When I bend the first two, I know that sooner or later I'll get the nail in, but the failures are still frustrating. The intensity is based solely on the importance of the plan not failing, which will often take the importance of the goal, the difficulty of the task, and the number of failures into account. The importance is an arbitrary function that can be tied to the plan (instead of being static values).

  Frustration does not fit into the OCC model. Although quite complete, the OCC model does not seem to be all-encompassing. This is one reason to make sure that the Em architecture is flexible enough to handle extensions to the emotion generators. I will show how frustration was added to the Em system in section 4.5.

- **Startle**

  Startle is generated when an agent hears a loud noise. This emotion type was added to improve the cashier in the gunman scenario. For the cashier, the intensity of the emotion is always 7 and the decay is very quick (the intensity halves every turn). Variations on this simple form of startle are possible. For instance, emotion structures can be generated based on a variety of intense, unexpected sensory inputs. I only needed startle for my characters, so it was the only one that I built.

  Startle was not part of the cognitive OCC model. It is an example of an emotion that could conceivably be modeled cognitively, but that has a simpler and more intuitive perceptual model that works well for my artistic ends.

3.4 Discussion: Types of Emotion Generation Rules

One of the goals of this thesis is to provide some ideas to artists about how to use the tools I provide them. Much of the discussion throughout this chapter has been towards that end and this section will provide more.

By exploring the emotion research literature and trying to create emotional agents, I have developed a list of emotion types based on their underlying causes. I have not included all of these types in the default set of emotion generators, so I thought it important to provide artists with some ideas about other ways they might want to use the Em architecture.

- **Cognitive-Appraisal Emotions**

  Cognitive-appraisal emotions, as in the OCC model, are based on appraisals of events, actions, and objects in the environment. These appraisals tend to use reasoning and modeling of other agents to determine when to generate emotions. Though I have pointed out that Em's default rules are (mostly) not strict cognitive-appraisal rules, they are still based on many of the ideas from the OCC cognitive-appraisal model.

- **Reflex Emotions**

  Emotions that are driven directly from sense data are called reflex emotions. Startle is an example of such an emotion.

- **Episodic Memory-Based Emotions**

  Memories of past emotional events can recreate the remembered emotion. For example, remembering the death of a loved one can cause grief. As Tok has a very limited episodic memory system, I have not incorporated memory-based emotions into the default set of Em emotion generators.

- **Daydream-Based Emotions**

  Daydreaming about events can lead to emotions. For example, imagining getting revenge on an enemy or being on vacation are often enough to cause emotional reactions. Daydreams would have to be created by another part of the architecture and then passed on to Em where they can be appraised. Because there is currently no daydreaming module in Tok agents, these kinds of emotions cannot currently occur. Whether or not new generators would need to be written will depend upon the structure of the daydreaming system. It could be that the daydreams are handled by the already existing cognitive-appraisal rules.

  Eric Mueller [Mueller90] has done a good deal of work on the relationship between emotions and daydreaming and this work would provide a good starting point for extending Tok/Em to handle daydreaming and daydream-based emotions.

- **Sympathetic Emotions**

  People often have emotions about the emotions of other agents. As I described earlier, there are a number of ways that an agent can come to know what another agent is feeling, such as reasoning and modeling (in which case these emotions can be viewed as cognitive-appraisal rules) or by perception.

  The default system has some sympathetic-emotion rules. One such rule is to feel happy for a friend who is happy and feel pity for a friend who is sad. The rule set also includes rules for feeling resentment towards a happy agent who is disliked and gloating when a disliked agent is sad. Other rules, like being afraid when another person (friend or not) is angry, can be added by modifying these rules.

- **Social Contagion**

  Being around people who are in a particular mood can be enough to cause a similar emotional reaction. If everyone around you is happy, you might tend to be happy, even if you don’t have any other reason to be happy. Some psychology research has been done in this area by Hatfield et al. [Hatfield94]. The default emotion generators do not have any such rules.

- **Body-Feedback Emotions**

  The state of the body can lead to emotions, such as feeling happier when smiling. Similarly, general arousal (e.g., an adrenaline rush from physical exertion) can affect the intensity of emotions. Psychological evidence has pointed to this being true in humans [Schacter62]. The default Em system does not currently have any body-feedback rules.

I have included this discussion because I feel that explaining the potential breadth of the architecture might be useful for artists. For example, when building *Robbery World*, I found that I wanted the cashier to be startled when anyone fired a gun. Instead of trying to force this into a default cognitive-appraisal rule, I found it fit most easily into the category of reflex emotions. I added a rule and some other support code (details on how this is done can be found in section 4.5) and had a new reflex-based emotion. Being able to identify new types of emotions may be more productive than trying to fit all possible emotions into the default set provided.

3.5 Summary

Here are some of the important issues that have come up in this chapter.

- I described some important decisions about how to build tools to support artists in creating believable emotional agents. In particular, I separated the emotion architecture from the emotion system, chose to use an explicit emotion system instead of creating a system with emergent emotions, and took a broad approach to the problem instead of a narrow-and-deep one.

- I developed an emotion architecture, Em, which sits within a broad agent architecture and supports the generation of emotion structures. The architecture provides a language, Hap [Loyall93,Loyall96], and a set of inputs which can be used to determine which emotion structures to generate and when. I found the Hap language appropriate for the emotion generation language because it has a number of features that are important for writing emotion generators, such as a flexible match language, demons, flexible control structures, the ability to perform internal (mental) actions, and priorities.

- I developed a standard set of emotion generators based on the cognitive emotion model of Ortony, Clore, and Collins [Ortony88]. This set of generators provides a starting point for artists, who might otherwise find it very difficult to create a good emotional agent with only the Hap language and the set of inputs. Em's set of default generators creates 24 different emotion types.

- I do not expect that the set of emotion generators I provide will be sufficient for all artists and have tried to aid the process of extending this set of generators by discussing possible extensions. The discussion about the various inputs and how they could be used and the discussion about other types of potentially useful emotion-generation rules, like reflex rules and rules based on episodic memory, should provide important information to artists who want new emotion generators.

- One of the key ideas I presented is the importance of working within a broad agent architecture. For instance, a broad architecture is necessary to provide the various inputs that emotion generators can use; using a narrow architecture limits the reasons for having emotions. Also, I showed how using a broad architecture can lead to some simpler and more complete models of certain emotions than the OCC model provides. In particular, I showed how to use processing in the motivation system and the perception system to help create emotions which the OCC model explains by using more complex cognitive processes and the modeling of other agents.

CHAPTER 4 Emotion Storage

Once the emotion generation part of the Em architecture produces a set of emotion structures (see Figures 2-1 and 3-1), they are passed to the parts of the architecture concerned with storage, combination, and decay. I describe each of these in this chapter. I also briefly discuss the query interface to the storage system that allows the behavioral feature system to access the current emotional state. I conclude with a description of how to add new emotion types to an emotion system.

4.1 Storage

The emotion generators can produce many emotion structures, often at the same time. Em provides mechanisms for storing those structures; these mechanisms have been designed to ease the task of processing the effects that the emotion structures have on the rest of the architecture. In particular, as I explain in Chapter 5, artists will often want to express emotions at various levels of generality—more general forms of expression are things like crying and smiling; more specific forms of expression are things like fighting with a particular agent or recalling a specific emotional event. In order to support both general and specific forms of emotional expression, the Em architecture stores emotion structures in an *emotion type hierarchy.*

4.1.1 The Emotion Type Hierarchy

An emotion type hierarchy is made up of nodes that represent emotion types, such as *anger* or *distress*. Each node can contain emotion structures of that type and can also point to other nodes which represent emotion subtypes. For example, the *distress* node might contain an emotion structure that was generated when the agent was insulted and it might point to a subtype node, such as *grief*, that contains an emotion structure that was generated when the agent's pet weasel died.

Given this structure, I have been able to write query functions that return information about emotion structures of a given type and of all of that node's subtypes. For instance, it is possible to query the *distress* node for all emotion structures that are of type *distress*; this query will return structures that are subtypes of *distress* as well. I have also been able to write queries that return combined intensity data. For instance, it is possible to query the *distress* node for the combined intensity of all of the emotion structures of type *distress*; this query combines the intensities of any structures stored in the *distress* node and any structures in subtype nodes, like *grief*. How the intensities are combined will be discussed in the next section.

Being able to get information about all structures of a type and its subtypes is useful for determining general effects on the rest of the agent. For example, a general *distress* reaction might be frowning. Whether the cause is being insulted or a dead pet weasel, this is a reasonable reaction. Instead of having to query each possible subtype to determine if there are any *distress* structures, artists are able to make a single query to the *distress* type and get information about all of the structures in the tree below it.[1]

Because Em doesn't store all distress types in a single node, it becomes easier to express more specific emotional reactions. If the artist wants emotion structures of type *grief* to result in more specific responses, like thinking about the loss, it is possible to tie emotion structures of this subtype to such effects without requiring all *distress* emotions to produce similar effects.

Here's an example from one of the simulations: the cashier in *Robbery World* has a *fear* type and a *startle* type which is a subtype of *fear*. When the cashier hears a loud noise, like a gunshot, his emotion generators create a *startle* structure. Because *startle* is a subtype of *fear*, the normal *fear* reactions ensue, such as turning pale and trembling. However, *startle* can also lead to more specific responses, like dropping what he's holding, which I don't want to happen when he has other

1. Note that this explanation is a bit simplified since the emotion structures do not directly affect the processing of the architecture. They are first mapped through a set of behavioral features that also have various levels of generality. In the examples give, distress would be mapped to a feature, like sulking, that would directly cause the frowning. More about behavioral features can be found in Chapter 5.

kinds of *fear* structures. By separating out this subtype, I can create more appropriate emotional reactions.

The default hierarchy provided with Em is shown in Figure 4-1. The top level node, *total*, is used mainly to create a single hierarchy, though this isn't necessary. It could be considered the total level of emotional arousal, but because the arousal of emotions like *joy* and *distress* are so different in how they affect the architecture, this will typically be an uninteresting value. The second level of the hierarchy, which contains *positive* and *negative*, is used to determine the general *mood* of the agent, whether good or bad. This *mood* level provides a level of generality even above the standard emotion types, which can greatly simplify the process of creating emotional effects. I come back to this in section 5.1. The next level of the hierarchy represents all of the standard OCC emotion types. Each type can have its own effects on the architecture. At the lowest level are *startle*, which both contributes to *fear* but also has some of its own effects on the architecture, and *frustration* which is in a similar relationship with *anger*.

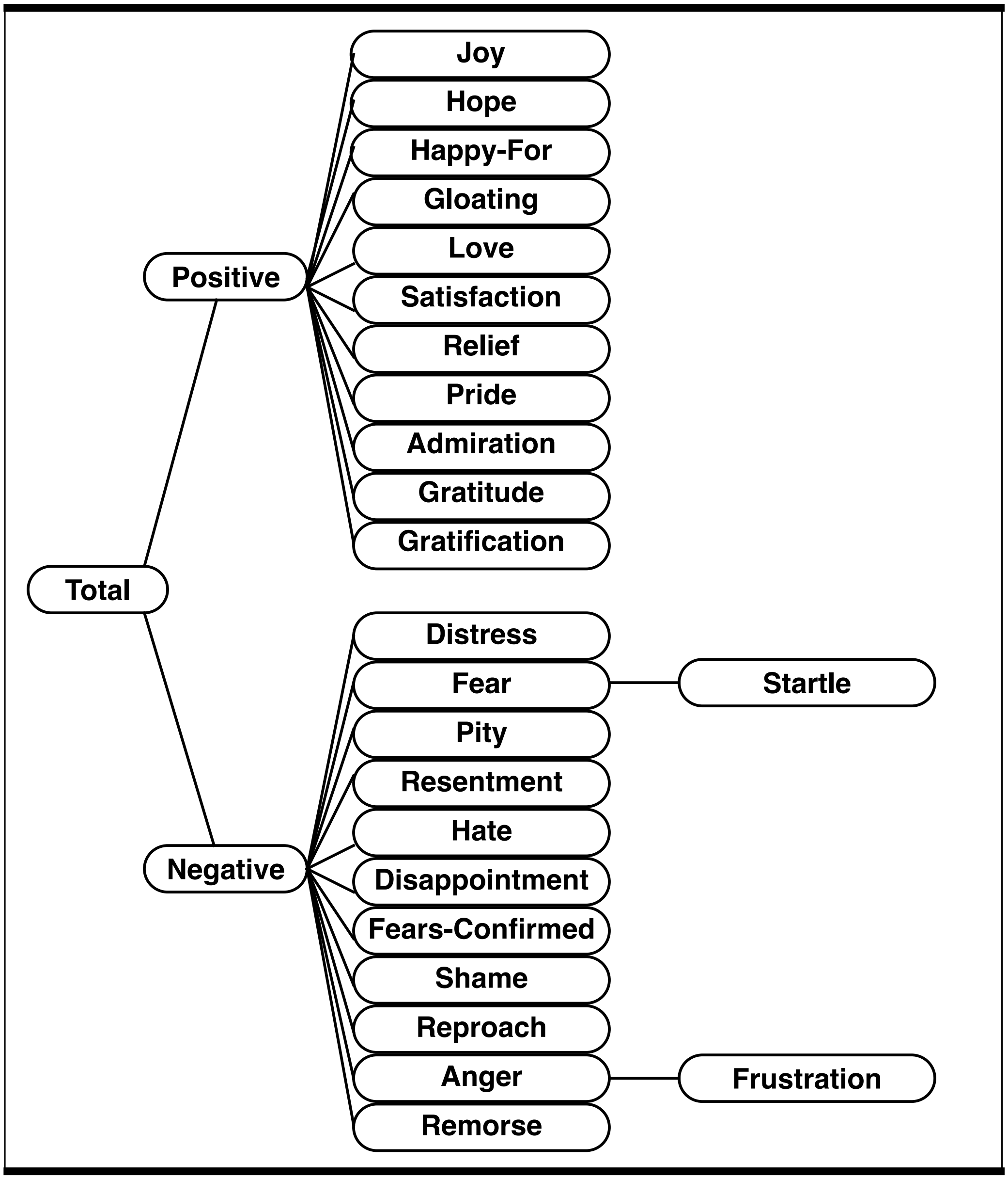
The typical structure for the emotion type hierarchy is a tree, though it is possible to have nodes with multiple parents. For example, *frustration* could be a subtype of *distress* and *anger* if that were deemed useful by the artist. In this case, *frustration* intensity would increase the intensity of both the *distress* and *anger* nodes.

Artists who create their own generation systems will often need to modify the standard hierarchy. Taking nodes out of the hierarchy is simple and uninteresting; I show how to add new types of nodes in section 4.5. Although the default hierarchy is quite shallow, I expect that artistic extensions will tend to make it deeper instead of wider. Because Ortony, Clore, and Collins have done a reasonably good job of covering a large number of emotion types, I expect that most artists will be more interested in refining one of these general types than creating a whole new type.

4.1.2 Generative Types vs. Expressive Types

Ortony et al., who introduced the notion of emotion types [Ortony88], defined them in terms of their antecedents. So, distress emotions are defined as those caused by appraising an external event to be the cause of the failure or increased likelihood of failure of a goal that is important to the agent. Subtypes, such as homesickness, have the same basic cause, but are differentiated by aspects of the emotion structure, such as the type of goal in question. I call this form of typing emotions *generative* typing.

FIGURE 4-1 **Default Em Emotion Type Hierarchy**



The goal in creating a useful storage mechanism, however, was to provide a way to organize emotion structures to ease the task of expressing emotion structures. In other words, by the time the architecture gets to storing emotion structures, how and why they were generated is less important than how they will affect the agent's behavior. To this end, emotion types are defined not in terms of how they are generated, but in terms of their effects. So, a type represents how an emotion structure can be expressed. I call this form of typing emotions *expressive* typing.

These two approaches often have similar results. For example, grief is both a generative subtype of distress and an expressive subtype of distress. It is a generative subtype because it is generated by the standard distress rule with the distinction that the goal in question is of a specific type. It is an expressive type because the general ways of expressing distress (e.g., crying, pouting, moving slowly) are all appropriate to grief as well, while grief may also have some specific ways to be expressed.

Generative and expressive typings, however, are not always the same. The relationship between *fear* and *startle* is an example where this effect-based type system is different than the OCC antecedent-based type system. According to antecedent typing, *startle* wouldn't be a subtype of *fear*. *Fear* is generated when an important goal is threatened, so antecedent subtypes would, for example, include *fear* of specific goals failing. *Startle* is generated by a reflex rule based on hearing loud noises, so it is rather different in how it is generated. However, the general ways in which *fear* is expressed (e.g., turning pale, focussing attention) are appropriate to *startle*, making it an appropriate expressive subtype of *fear*.

As I stated above, the design of the storage mechanisms for Em was based on the goal to make expression easiest. This influenced my decision to group emotion structures by expressive type instead of generative type.

4.2 Combination

When different emotion structures of the same type are stored together, they will often need to be combined in some manner. For example, say an agent has three distress structures present; to determine how distressed the agent is, the intensities of the three structures are combined to come up with a single value.

The Em architecture supports artist-defined combination functions at every level of the emotion type hierarchy, so the combination rules can be fairly specific, including options like anger structures combining differently than gratitude structures. As I mentioned before, I tried to keep choices like this in the hands of the artist, while still providing useful defaults that produce good behavior in practice.

In this section, I will briefly lay out three options for how to combine emotion intensities, including the one that is supplied as the default combination rule. The default rule is used for combining emotion structures within a type node and for combining the intensity of nodes within a type and a set of subtypes.

1. **Winner-takes-all.** If an agent has two distress structures and one grief structure with intensities 3, 3, and 4, the intensity of the agent's distress will be 4. This has the advantage that lots of small emotions don't suddenly cause the agent to have a violent emotional reaction. It has a related disadvantage, however, that a large number of medium-intensity structures will never result in a strong reaction.
2. **Additive.** It is possible for intensities to be added together. Distress structures of intensity 3, 3, and 4 would result in a combined intensity of 10. This has the advantage that multiple emotions can result in strong reactions. It has the disadvantage, however, that a few minor incidents can lead to intense reactions. Since I typically make the scale of intensity run from 0 to 10, it is possible for 3 medium-low structures (intensities of 3, 3, and 4) to have the same sort of intensity as the most intense of emotions (intensity of 10).
3. **Logarithmic combination.** This is the default option in the Em system and the one used in all of the Tok agents created to date (except the first Oz agent, Lyotard the cat [Bates92a], which used option #2). In this case, emotions are added together logarithmically, so to combine intensity I and J would be computed as $\log_2 (2^I + 2^J)$. Combining structures of intensity 3, 3, and 4 results in a combined intensity of 5. The disadvantage of this approach is that it is a bit more complicated. The advantages are that it allows multiple emotions to lead to more intense reactions without the reactions getting out of hand too quickly.

4.3 Decay

The decay process might be quite different from emotion to emotion. Some emotion structures might decay quite slowly, such as *anger* in characters known for holding grudges. Some might decay quite quickly, such as *startle*. In Em, each emotion structure is provided with its own decay function. In this way, artists can have control over emotional decay at the individual-emotion level. For example, if an artist wanted an agent whose anger structures decayed very slowly, they could create that effect. If the artist wanted only very specific instances of anger to day slowly (anger of intensity greater than 8, directed at Rick, that is a result of Rick calling me a "goober"), that can be done as well.

The default decay function is to lose one level of intensity each tick[1] until the intensity drops to 0. The two exceptions are startle emotion structures, which decay by one half of their total intensity each tick, and fear and hope emotions, which stay at the same intensity until the cause of the fear or hope is removed, at which point they decay at a rate of 1/tick. For instance, in *Robbery World*, the gunman gets scared when the officer arrives on the scene and he stays scared as long as the officer is threatening to capture him. If the officer wanders off, the gunman's fear will subside. (It might be possible for the gunman to infer that if the officer leaves, he may be going for backup; in this case, the intensity of the fear would remain high. This particular inference rule does not happen to exist in the gunman character.)

In earlier versions of the gunman, his fear started to decay immediately after being generated, which caused him to be almost emotionally neutral after a short time. This lead to behavior that I found unbelievable.

A similar effect of keeping fear high can be created by using the standard decay function and letting the emotion generators notice that the emotion intensity is too low for this given situation and updating it. The advantage to such a set up is simplicity. It doesn't require a new decay function and it would be handled automatically by the standard fear-generation rule. However, the disadvantage is that raising and lowering the fear intensity is not only computationally more expensive (at least in my system), but can also lead to odd behavior.

For instance, let's say the gunman has a `shoot-officer` behavior that is triggered when his fear intensity is 7 and turns off when the fear drops below 7. The behavior begins with the gunman aiming his gun at the officer and then shooting. If the fear intensity starts at 7, the behavior begins and the gunman aims his gun. When the emotion decays, the behavior stops. When the fear rises again, the behavior will (probably) be started again. This restart may or may not be accomplished smoothly. If it is not, he will aim his gun again instead of shooting. By simply maintaining the fear at level 7, this kind of behavioral problem doesn't arise.

The default decay behavior may not be desirable for every agent in every environment, which is why the decay mechanism has been designed to allow artists to develop decay functions that fit their needs.

1. In the text-based Oz worlds, each character acts in a round-robin order. A tick is a cycle through each character. In the real-time Oz worlds, emotion structures decay one level of intensity each second.

4.4 Querying

Em provides a number of emotion query functions that have proven to be useful for accessing the emotional state. As I will describe in the next chapter, emotion structures in Em affect the rest of the architecture through an intermediate system of behavioral features. This means that I haven't had to provide general-purpose query functions for use by the rest of the architecture, but I have provided a set of functions that have proven useful for generating the set of behavioral features. In order to make the mapping from emotion structures to behavioral features quite flexible, I have made the query interface to the set of emotion structures reasonably rich.

The query functions that I have built for the emotion type hierarchy can return the following information:

- The combined intensities of all emotion structures of a given type (all of these queries return subtype information as well). Example: How happy am I?

- The combined intensities of all emotions of a given type and direction. Example: How grateful do I feel towards Tim?

  This kind of query only makes sense for *directed* emotion types. Directed emotion types are those that can be focused at a particular agent or object. The fact that one can be "angry at" means that it is a directed emotion. *Non-directed* emotion types still have causes, just no direction. For example, agents can be "happy about" but not "happy at." Some directed emotion structures do not always have an object they are being directed at, so agents can be angry without being angry at anyone in particular.

  The set of non-directed emotion types in the standard Em system is the following: joy, distress, pride, shame, frustration, gratification, remorse, relief, satisfaction, disappointment, fears-confirmed, frustration, and startle. The directed emotion types are the following: fear, hope, like, dislike, happy-for, pity, gloating, resentment, admiration, reproach, anger, and gratitude.

- The combined intensities of all emotions of a given emotion type and cause. Example: How afraid am I that I will fail the test?

- The combined intensities of all emotions of a given emotion type, direction, and cause. Example: How angry do I feel towards Ralph for hitting me?

- A list of all directions and intensities associated with a given emotion type. Example: Who am I angry at and, for each, how angry?

- A list of all directions and intensities associated with a given emotion type and cause. Example: Who am I grateful to for throwing me a party and, for each, how grateful?
- A list of all causes for a given type. Example: What goals am I currently afraid are in danger of failing?

  For standard-based emotions (pride, shame, admiration, reproach), the “cause” is a little ambiguous, so there are three different functions, each of which performs a variation on this query.

  - A list of standards associated with a given emotion type. Example: What standards of behavior have I broken that I am currently feeling ashamed of breaking?
  - A list of actions associated with a given emotion type. Example: What actions have I performed that I am currently feeling proud of?
  - A list of actors associated with a given emotion type. Example: Who am I currently feeling admiration towards?
- A list of emotion types sorted by order of overall intensity.

This set of queries has proven to be rich enough for creating behavioral feature maps, which is how I have used it. I have used almost all of the queries above for various agents, so the richness has been important. Also, this set of queries is all that I needed or wanted when building characters, so it is complete enough to be useful, at least for the tasks that I have needed it for. It could be that this set would need to be extended if an approach were taken where behaviors needed to directly access the emotion structures.

4.5 How to Create a New Emotion Type

Now that I have described emotion generators, emotion storage, and emotion decay, I can describe how to create a new emotion type. The ability to extend the emotion system is critical to my approach, which is to place as few constraints on the artist as possible. I expect that artists won’t need to add new emotion types often, but it is still critical that they be able to when they need to. This section will describe how to create a new emotion type and the support that Em provides for this task.

As an example, I will demonstrate how to add frustration to Em. Clearly, this isn’t necessary since it’s already in the system. But, for the moment, let’s just pretend that it isn't. In order to understand this process, it is important to know that emotion structures are objects in an object-oriented system.

Here are the steps that an artist has to go through to create a frustration emotion type[1]:

1. The artist needs to define an emotion-structure type to represent *frustration*. There is already a generic *emotion-structure* type in the system, so the artist simply creates a subtype of that generic structure type and calls it something like *frustration-structure-type*.
2. This new type needs to have two methods associated with it: *get-emotion-intensity* and *decay-emotions*. The *get-emotion-intensity* method takes an emotion type name (like JOY) as input and returns the intensity of the emotion structure if that type name is FRUSTRATION; otherwise it returns 0. The emotion intensity will typically be stored in a slot in the emotion structure. *Decay-emotions*, when called, simply decrements the intensity slot of the structure if it is above 0.
3. Once this new class of emotion structures is created, this type of emotion needs to be placed in the emotion type hierarchy. The artist places this as a child of *anger*, which makes all *frustration* intensity turn into generic *anger* effects on the agent's behavior. (The artist can also add *frustration*-specific effects as I will demonstrate in the next chapter.) The artist creates a new node for the hierarchy, which is an instance of the *emotion-type* type and names it *frustration-emotions*. This new node is placed in the emotion type hierarchy as a child of *anger* and with no children of its own.
4. The last thing the artist needs to do is to create an emotion generator for this emotion—a Hap demon that fires when an instantiated behavior structure is placed in the agent's *plan-failures* slot. The generation rule creates an emotion structure of type f*rustration-structure-type*, sets the intensity and cause slots of the structure to represent the importance of the failed plan and the plan itself respectively. Then this structure is placed in the *frustration* node in the emotion type hierarchy. See Figures 3-3 and 3-2 for this generator.

That's it. This new emotion type will be generated and stored just like the other emotion types. It will even immediately affect the agent's behavior because it contributes to the effects of anger. It is also now possible to use emotion structures of this type to generate specific frustration behaviors and other effects on the architecture.

1. This description is for adding new emotion types to the non-real-time Em. Adding new structures to the real-time Em is similar, though some of the details differ.

4.6 Summary

Here are some of the important issues that have come up in this chapter.

- I presented the Em architectural support for storing, decaying, combining, and querying an agent's current set of active emotion structures.
- I introduced an emotion type hierarchy which stores emotions by expressive type and drew a distinction between expressive and generative emotion types.
- I described the default Em emotion type hierarchy, combination functions, and decay functions. I also discussed some possible variations on these defaults.
- I described the Em query system used to provide information about the current emotional structures to the behavioral feature system.
- I explained how to add a new emotion type to an agent's emotion system in Em.

CHAPTER 5 Expressing Emotions

Generating the right emotion structures at the right times is an important step towards creating a believable emotional agent. However, if those emotion structures just sit in memory and decay, they don't do much good. Unfortunately, there is less work in psychology and AI related to computational models of emotional expression as there is work on emotion generation. One such model is Gilboa and Ortony's model of "action responses" which is described in [Elliott92], but that is less-well developed than the Ortony, Clore and Collins model of emotion generation [Ortony88].

Recall from Chapter 2 that four important lessons to learn from that arts about emotions are [Thomas81]: (1) emotions (including the expression of emotions) are critical to creating believable agents, (2) emotions should permeate behavior (i.e. they should be expressed through facial expression, motion, speech, thought, etc.), (3) emotional expression should reflect the individual, quirky personality of the character, and (4) believability is the goal, not realism.

I have no elegant theory for how to do this. I'm not even sure that an elegant theory is appropriate to a task that demands quirky behavior. Instead, I have designed a two-part system for expressing emotions in believable agents that I have found to be suitable to the artistic nature of this task.

The first part of the system maps the set of emotion structures into a set of behavioral features. For instance, an *anger* emotion structure might be mapped to an *aggressive* behavioral features. As I will describe, the structure of the behavioral features, the default set of features that I supply, and the default mapping from emotion structures to behavioral features are all heavily influenced by the goals of believability and character-specific expression. For instance, the behavioral feature system will often choose to exaggerate the expression of certain emotion structures and not express other structures at all. These techniques are drawn directly from the arts and are used to create clearer expressions of emotion.

In the second part of the emotional-expression system, the set of features are expressed by other parts of the agent architecture, such as in the way the agent speaks and moves. When designing the emotion generation part of Em, I was able to provide useful defaults because I was able to find a useful set of emotion types to build upon. For expressing emotions, such defaults are harder to find. For instance, it would be useful to provide default emotional effects on agents' behaviors, but I have no set of standard behaviors to build upon. Since I have found that emotions are expressed in rather different ways from behavior to behavior, providing default ways of expressing emotions through behaviors proved to be an unfruitful approach.

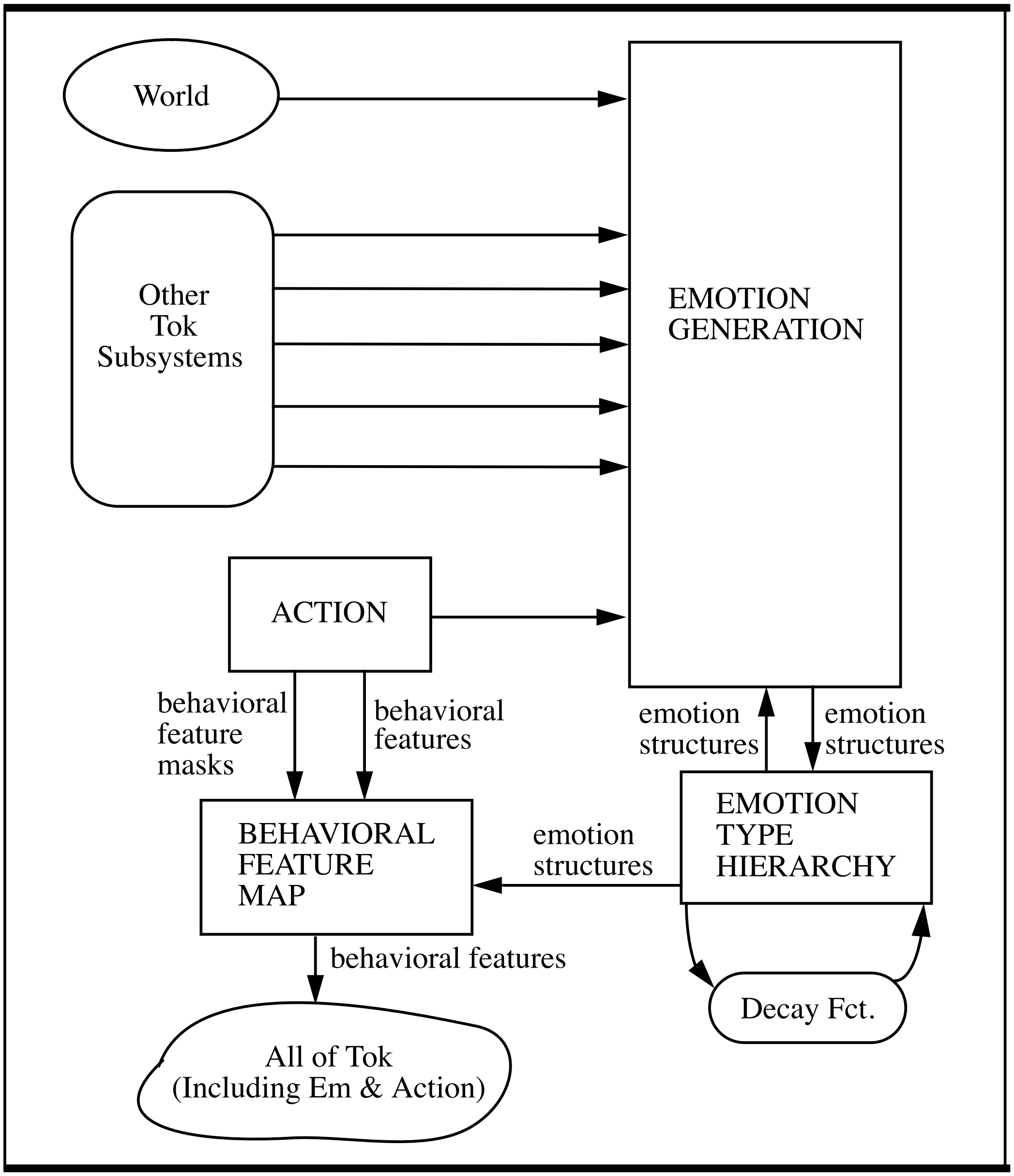
Instead, I have focused on the need of artists to create characters where the expression of emotions permeates the character. I describe a set of mechanisms for expressing emotions in believable agents that artists can use when creating specific characters. In a few cases, where it has been feasible, I have built default expression rules that the artist can use. For instance, there are default rules for updating the agent's facial expression and attitudes about other agents.

5.1 Behavioral Features

When creating emotional agents, I have found that it is useful to have a level of indirection between the emotion structures and the expression of emotion. I will describe with this is so in section 5.1.1. To meet this need, Em provides a set of structures called *behavioral features*. Figure 5-1 shows where behavioral features sit in the larger Em architecture. (This figure is similar to Figure 3-1, with some detail removed and other details added in order to show the role of the behavioral features more clearly.) Basically, there is a *behavioral-feature* map (or an *emotion-to-feature map*) that is written in Hap and that maintain a set of behavioral features based on the current set of emotion structures.

The figure also shows inputs to the behavioral feature system from the action module. Agents can choose to act in a particular way if it suits their needs. For instance, an agent might act aggressively even though the agent is not angry. The action system can also create behavioral feature masks, which mask out certain features. So an agent that would otherwise act aggressively, might suppress that feature based on input from the action system.

FIGURE 5-1 **Behavioral Features in Em/Tok**



In section 5.1.1, I motivate the choice to have a behavioral features system at all. Sections 5.1.2 through 5.1.5 are related to creating specific behavioral feature systems within this framework. Section 5.1.2 provides some practical and artistic suggestions for choosing a particular set of behavioral features to use; section 5.1.3 includes the default set of features that Em provides; in section 5.1.4, I discuss what makes a good mapping from emotional structures into the behavioral features; finally, section 5.1.5 includes the default emotion-to-feature map, some examples of mapping rules, and the emotion-to-feature map for Melvin from *The Playground*.

5.1.1 Motivation for Behavioral Features

It may not be immediately obvious why artists would want a level of indirection between the emotional structures and the expression of those structures. If the mappings were simply *anger* $\rightarrow$ *aggression* and *joy* $\rightarrow$ *cheerful*, the behavioral features wouldn't be useful. It turns out, though, that behavioral features provide artists with the ability to do a number of interesting things with the expression of emotions.

Here are a number of effects that artist might want to achieve along with an explanation of how to achieve them using behavioral features. Many of these effects are related to creating characters with personality-specific ways of expressing emotions. These effects can be achieved by other mechanisms, so I propose that the reader judge the value of the behavioral feature mechanism according to how easily it seems to support encoding these effects.

1. **Individual personality.** Brad might act aggressively when angry, but Sam might get quiet and withdrawn. Artists can achieve these different effects by mapping the *anger* emotion to different behavioral features in different characters. Brad would map *anger* to *aggressive*; Sam would map *anger* to *withdrawn*.
2. **Repression of emotions.** Nancy might never show her anger, even when she has a good reason to be angry. If an emotion is mapped to no feature or to a feature with a much lower intensity, the expression is eliminated or muted, thereby repressing the emotion.
3. **Feigned emotions.** Bill might feign anger to get the store manager to give him a discount. This is accomplished by the action system adding an aggressive feature. Also, the action system can mask out features. For example, in *Robbery World*, the gunman is scared when the police show up, but masks his fear (except for turning pale). The arrows from action to the behavioral features in Figure 5-1 show these two effects.

4. **Redirected behavior.** If Tammy is mad at her boss, she might not act aggressively towards her boss, but she might go home and take it out on her pet iguana. This can be achieved by mapping *anger* to *aggressiveness*, but changing the direction slot. In the Woggles, if Wolf (the bully) is angry at Bear (who is bigger than he is), he will take it out on Shrimp (who is smaller).

5. **Atypical behavior.** Some agents might just be atypical—they act happy when angry and grateful when insulted. The character might be from a different culture, or from a different planet, or maybe the artist is trying to achieve the effect of a mentally unbalanced character.

6. **Creating a coherent emotional state.** There can be lots of emotional structures at any one time and trying to express all of them at once would be both difficult and would produce unwanted results. By mapping the set of emotional structures to features before they affect behavior, a coherent set of features can be chosen to express.

7. **Mixing emotions.** Sometimes an agent will have a few strong emotions that work together reasonably well. For example, an agent could have bittersweet feelings towards some event that is both positive and negative. Instead of trying to express *joy* and *distress* at the same time, the feature map can recognize these kinds of situations and create a feature to represent the mixed emotion, which can be expressed in its own unique way.

8. **Allowing goals to affect other goals.** In languages (like Hap) that don't provide many mechanisms for inter-behavior communication, the behavioral features can provide a crude mechanism for accomplishing such interactions. This is similar in spirit to Blumberg's work [Blumberg94] which uses a blackboard mechanism for inter-goal communication. In Tok, a goal can create a feature that is used to affect processing of other goals. I will discuss this in more detail (relative to social goals taking other goals into account) in Part II of this thesis.

5.1.2 Choosing a Set of Behavioral Features

The behavioral feature architecture is quite flexible. Artists can have whatever features they want and those features can have any arbitrary structure. Here are a number of artistic and practical considerations that I have used in designing the default structure of features and feature sets for the characters I have built.

Specificity of Emotional Expressions & The Structure of Features

In building agents, I have generally found that the more specific the response to a situation is, the more believable it is. For instance, instead of just getting bug-eyed when scared, agents should have specific responses based on the specific cause of the fear. Clearly, the type and intensity of the feature should affect how it is expressed; by also storing directional and causal information in behavioral features, Em allows artists to create even more specific expressions. That is, the structure I chose for the behavioral features (i.e. type, intensity, direction, cause) supports the artistic expression of emotions.

Here are examples of using the cause and direction information to create specific emotional reactions:

- **Specificity based on cause.** Imagine Larry is walking down a city street at night and is confronted by a masked gunman. Larry's fearful response (indicated by a *defensive* feature) might be general, such as sweating, trembling, and focussing his attention on the gunman. But his response might also be more specific, such as running away. Running is a particular type of reaction that is appropriate for some causes of fear but not for others (e.g., fear of failing a test).
- **Specificity based on direction.** By keeping directional information with features, expression can also be directed at specific agents. Instead of just acting *aggressively*, agents can act *aggressively* towards particular agents. Also, features that are directed at different agents can be expressed differently. For instance, *aggression* towards your boss and *aggression* towards your dog might be expressed differently.

What Makes a Good Behavioral Feature?

From a practical standpoint, there are at least two ways to simplify the process of creating a believable emotional agent by means of the behavioral features. First, generating some abstract features, like *good-mood*, *bad-mood*, and *energy* can make expressing emotions through behaviors simpler for the artist. Em requires artists to build the expression of behavioral features explicitly into the agent's behaviors. For instance, if an artist wants to create a character that can play cards emotionally, the playing cards behavior will need to be written such that the agent plays differently when different features are present. Furthermore, if an artist wants to create a character that is broadly expressive (that is, it expresses emotions through a broad collection of channels), this requires the features to affect much of the agent's processing.

I have found that many behaviors (or aspects of behaviors) might need to be influenced only by the presence or absence of, say, a *good-mood* feature, where *good-mood* is generated by the *positive* emotion structures in the emotion type hierarchy. This allows for some emotional expression without the need to explicitly work a large number of features into every aspect of every behavior. Taking this approach, though, means that the agent's expression is not as specific as it could be and I just argued in the last section that specificity is a good thing. I have found, however, that for many instances of expressing emotions in behaviors, the loss of specificity is not noticeable, while the decrease in the amount of work that needs to be done by the artist is.

The second way to simplify the work associated with the behavioral features is to not use more features than are necessary. As I have noted, each behavioral feature has to be integrated with the processing of the rest of the agent. Although there is some architectural support for this integration, the artist still needs to consider how each goal of the agent should be affected by the current set of features. By limiting the number of features to those that are really important for expressing the personality of the agent, this task is simplified. In particular, when choosing a set of features for a particular character, there is no need to incorporate all of the features in the default set if they are not appropriate for that character.

5.1.3 The Standard Behavioral Feature Set

Now that I have discussed some the elements of a good set of features, I will describe the default set that I provide with Em. Artists are also able to modify and extend the default set of features arbitrarily. In Table 5-1, I briefly describe each of the 33 standard features. In section 5.1.5, I will describe how these features are generated from the agent's set of emotion structures.

Recall that each features has a type. Features also have an intensity that ranges, by default, from 0 to 10. For example, an *energy* feature of 0 means that the agent is very lethargic; if the feature is 10, the agent is very energetic. For some features, it also makes sense to have direction and cause information. For instance, an agent can be acting generally *unfriendly* or singling out a particular agent to act *unfriendly* towards. Because the direction is usually another agent, I have labelled the direction of the features in Figure 5-1, "agent." If a feature type in Table 5-1 has nothing in parentheses next to it, that refers to the feature with the cause and direction being empty. Similarly, if the cause or direction is left out of the parentheses, that denotes that slot is empty.

The set of features I provide reflects a large number of features that are designed to express the types of emotions that Em supports (e.g., emotion structures of type *joy* are expressed by acting *cheerful*ly). Many other features were created to support specific agents and domains (e.g., *energy* has been used by Lyotard the cat and the Woggles). Others have been added for practicality reasons (e.g., *good-mood*).

TABLE 5-1 The Default Set of Behavioral Features

Feature Type	Description of General Effect on Agent
cheerful	the agent acts in a cheerful manner
friendly	the agent acts friendly towards everyone
friendly(agent)	the agent acts friendly towards another agent
unfriendly	the agent acts unfriendly towards everyone
unfriendly(agent)	the agent acts unfriendly towards another agent
generous	the agent acts generously towards everyone
generous(agent)	the agent acts generously towards another agent
grateful(agent)	the agent acts gratefully towards another agent
sulking	the agent sulks or acts despondent
withdrawn	the agent acts withdrawn
aggressive	the agent acts generally aggressive
aggressive(agent)	the agent acts aggressively towards another agent
aggressive(cause)	if cause is an active goal, the agent aggressively pursues it; if the cause is a failed goal, the agent acts aggressively but in a manner appropriate to the specific goal having failed
aggressive(agent, cause)	the agent acts aggressively towards another agent and for some particular reason (cause)—if the cause is an active goal, the agent aggressively pursues it; if the cause is a failed goal, the agent acts aggressively but in a manner appropriate to the specific goal having failed
defensive	the agent acts defensively or fearfully
defensive(agent)	the agent acts defensively because of another agent
defensive(cause)	the agent acts defensively towards an active goal (cause)
defensive(agent, cause)	the agent acts defensively towards one of the agent's goals (cause) because of another agent
avoid(agent)[a]	the agent prefers to physically avoid another agent
approach(agent)	the agent prefers physical proximity to another agent

Feature Type	Description of General Effect on Agent
proud	the agent acts proudly
ashamed	the agent acts ashamed
gloat(agent)	the agent gloats at another agent's expense
gloat(agent, cause)	the agent gloats over a specific event (cause) at another agent's expense
console(agent)	the agent consoles another agent
console(agent, cause)	the agent consoles another agent when other agent needs consoling because of a cause
congratulatory(agent)	the agent acts congratulatory towards another agent
congratulatory(agent, cause)	the agent acts congratulatory towards another agent for a specific cause
contempt(agent)	the agent acts with contempt towards another agent
awe(agent)	the agent acts with awe towards another agent
anticipation	the agent acts excited/anticipatory
anticipation(cause)	the agent acts excited about some goal succeeding in the future (cause)
anticipation(agent, cause)	the agent acts excited that another agent is going to cause a goal (cause) success
good-mood	the agent acts as if in a good mood
bad-mood	the agent acts as if in a bad mood
energy	the agent acts lethargically/energetically

a. Features like *avoid* and *approach* may seem similar, in some senses, to goals. Agents can have demons that create an approach goal based on an approach feature, but this isn't necessary and will often be undesirable to interpret the feature so narrowly. These features can also be used in much more general ways, such as when deciding which agent to ask for help or which agent to play with. Also, as mentioned above, features can be used as intermediaries between goals. For example, Melvin has a goal to become friends with the user. This goal creates approach and friendly features which are used in other behaviors, such as negotiation, to modulate those behaviors to reflect the friendship goal.

5.1.4 Choosing a Behavioral Feature Map

Now let's look at some issues to consider when designing a mapping from emotion structures into behavioral features. I begin by presenting artistic factors to consider and then discussing some of the options to consider when trying to create a coherent set of features to express.

Artistic Considerations in Behavioral Feature Maps

Artists claim that exaggeration is an important aspect of traditional acting and animation (e.g., [Thomas81]). By exaggeration, they mean that certain aspects of the character are emphasized to make them clearer. For instance, an actor on a stage uses exaggerated movements because they are easier to see from the audience. More subtle forms of exaggeration are used in movies and television acting as well. This knowledge can help artists create effective feature maps. The map should, for instance, exaggerate feature values in artistically appropriate ways. One simple way to exaggerate the emotional state of the agent is to use a highest-feature-wins approach to features that are opposites, like *good-mood* and *bad-mood*.

For example, let's say Melvin has just finished a successful trade with the player but was also insulted by Sluggo. The trade success leads to a *joy* structure of intensity 7 and the insult leads to a *distress* structure of intensity 5. Instead of combining these emotions in a way that mostly cancels both of them out, let the positive win out and create a *good-mood* feature of intensity 7. In an artistic sense, this exaggerates the expression of joy to make it clearer.

Another thing to be learned from other media, like animation [Thomas81], is that it can be very difficult to express conflicting thoughts or emotions at the same time. In animation, a common technique is to express one thing and express it strongly. Again, this is an artistic point because it reflects the need for the character to be understood by an audience. I have found a similar problem with expressing inner conflict in interactive characters. The highest-feature-wins approach seems to be a rough approximation of the animators' solution. In the example above, instead of trying to express the mixed *joy-distress* state, the *distress* is repressed for the sake of clarity of expression.

Creating a Coherent Set of Behavioral Features

One advantage to using a behavioral feature map is that it can be used to create a more coherent state than the full set of emotion structures provides. For example, by using the highest-feature-wins approach just described, a number of conflicting emotion structures are not passed along to the behavioral features; this means that there is a smaller, more coherent set of features to be expressed than if there were a simple one-to-one mapping from emotion structures to behavioral features. Exactly how coherent to make the state, however, is still a matter of preference.

The option that I have mostly used has been to make the state somewhat coherent, but not completely so. In the default mapping I provide, it is still possible to have a large number of potentially conflicting behavioral features (such as *cheer-*

ful and *aggressive*) present at any one time.[1] This does not have to be the case, as the mapping could be much more selective in which features are generated.

There are advantages to both approaches. The advantages to allowing conflicting sets of features are the following:

- It is easier to write and maintain the mapping rules when there are fewer interactions between rules. For instance, the *aggressive*-feature rule doesn't have to be concerned with whether *cheerful* is also being generated.
- Some emotional effects can take advantage of a diverse set of features. For instance, a *sulking* agent might want to find a shoulder to cry on. The choice of who to turn to could be based on a *friendly* or *approach* feature. If *friendly* and *approach* were not generated because they conflicted with *sulking*, this would be more difficult to achieve.

The advantage to not allowing conflicting sets of features is the artistic one that it is typically more effective for a character to display one emotion very clearly instead of trying to display inner turmoil. Artists can make agents that are more clear in their expressions by using the feature map to ensure a completely coherent set of features to express.

Individual artists will have to decide which of these approaches is most appropriate for their characters and style. The default mapping adopts the first approach of allowing conflicting features. In Melvin, however, I used a hybrid approach that is mostly based on the first approach of allowing conflicting features, with some elements of the second approach incorporated, such as not allowing *cheerful* and *withdrawn* features simultaneously.

5.1.5 The Default Behavioral Feature Map

I now turn to the default Em mapping from emotion structures to behavioral features. Although it will be important for artists to fine tune the emotion-to-feature map for their particular characters, Em provides a standard mapping to start from. Many artists may want to throw this out entirely; others may find it just requires some tweaking. Like the emotion generators, the emotion-to-feature map is written as Hap rules. New rules can be added to the set or old ones can be modified or removed as desired.

1. I use "conflicting" to mean features that are somewhat at odds, but not directly opposed. Opposing features, like *good-mood*/*bad-mood* and *approach*/*avoid* should be handled as discussed in the previous section.

Here's an example. Say that an artist wants an agent to act generally *unfriendly* when feeling *distressed* or *angry*. Because this is a general unfriendliness, the direction and cause information can be left out. The intensity of the *unfriendly* feature is a function of the current intensities of the *distress* and *anger* types in the emotion type hierarchy, such as combining one-third of the *distress* intensity and one-half of the *anger* intensity using the logarithmic combination function described in section 4.2. Figure 5-2 provides a pseudo-code description of such a rule. In these rules, there is an understanding that the intensity of the features is cut off at 10. I do this by having the values returned by sum_intensities cut off at 10.

I will use a shorthand notation for such rules:

```
unfriendly := distress/3 + anger/2
```

The "+" refers to the default logarithmic combination function. So, if the agent is *distressed* at intensity 6 and *angry* at intensity 4, the agent will be generally *unfriendly* at intensity $\log_2 (2^{(6/3)} + 2^{(4/2)}) = 3$.

In some cases, single shorthand rules are implemented as more than one Hap rule; in other cases, multiple shorthand rules are implemented as single Hap rules. So, the shorthand rules are not a perfect representation of the underlying code, but they nonetheless provide a useful way to describe feature maps without having to resort to lots of code.

Here's another example. The default rule for calculating the intensity of the *unfriendly* toward another agent a is:

```
unfriendly(a) :=  dislike(a) + anger(a)/2 + distress/3 +
                  reproach(a)/3 + resentment(a)/5
```

This rule takes into account general emotions, like distress, as well as emotions specific to the agent in question, like dislike. The *a*'s represent particular agents. There are also examples where the particular goal is important and this is represented as a parameter *g*.

FIGURE 5-2 **Pseudo-Code for Undirected *Unfriendly* Behavioral Feature**

```
Demon-Name: bf-update-undirected-unfriendly-demon

;; The lefthand side of the rule computes what the
;; current intensity of the general unfriendly feature
;; should be and compares it against the actual current
;; value. If they are different the demon fires.
LHS:
 AND(
  ;; distress_int is the current undirected intensity of
  ;; all of the agent's distress emotion structures
  ;;
  ;; anger_int is the current undirected intensity of
  ;; all of the agent's anger emotion structures
  ;;
  ;; value combines fractions of these two values and
  ;; represents what the intensity of the undirected
  ;; unfriendly feature should be
  distress_int := em_intensity(DISTRESS,nil);
  anger_int := em_intensity(ANGER,nil);
  value := sum_intensities(distress_int/3, anger_int/2);

  ;; bf_value is the current value of the unfriendly
  ;; behavioral feature -- nil indicates the undirected
  ;; version of the feature -- bf_value is 0 if there
  ;; is no current undirected unfriendly feature
  bf_value := current_feature_value(UNFRIENDLY,nil);

  ;; value and bf_value are not equal (i.e., the actual
  ;; feature value is not what it should be) -- without
  ;; this test the demon would fire repeatedly
  (value != bf_value)
 )

;; The righthand side of the rule creates a feature with
;; the new intensity value. If this feature already exists
;; with a different intensity value, it is overwritten.
RHS:
 add_feature(UNFRIENDLY,nil,value)
```

The format of the directed-*unfriendly*-feature rule is very similar to the undirected version shown in Figure 5-2, except that there are additional emotions to take into account and the emotions and feature are directed towards an agent. Figure 5-3 shows pseudo-code for the directed version of the *unfriendly* feature. In both cases, the code creates a demon that computes the correct value of the feature and compares it to the actual value. When they are not the same, the demon fires and the feature is updated. Most of the code is to determine what the current value of the feature should be and to compare it to the current value of the feature. (Although the feature-generation rules are not long, they are long enough and repetitive enough that I have introduced the shorthand above.)

In the text-based version of Em, the implementation of these rules is not ideal as it requires a number of demons to constantly recompute feature values. A better implementation would have feature changes that are driven by changes to the intensities of emotion structures or nodes in the emotion type hierarchy. This is closer to what happens in the real-time version of Em; the matching is done by an incremental Rete matcher [Forgy82], which automatically optimizes the matching to limit the needed computations.

FIGURE 5-3 **Pseudo-Code for Directed *Unfriendly* Behavioral Feature**

```
Demon-Name: bf-update-directed-unfriendly-demon

;; The lefthand side of the rule computes what the current
;; intensity of the unfriendly feature should be towards
;; nearby agents and compares it against the actual current
;; value. If they are different the demon fires.
LHS:
 AND(
  ;; agent can be bound to any agent nearby (in the same
  ;; location as) self
  agent := know-of-ob(type=agent,relative-location=nearby)

  ;; emotion values are retrieved -- most are the intensity
  ;; of emotions directed at the given agent
  distress_int := em_intensity(DISTRESS,nil);
  anger_int := em_intensity(ANGER,agent);
  dislike_int := em_intensity(DISLIKE,agent);
  reproach_int := em_intensity(REPROACH,agent);
  resentment_int := em_intensity(RESENTMENT,agent);

  ;; value is set to what the unfriendly feature towards
  ;; agent should be set to
  value := sum_intensities(distress_int/3,anger_int/2,
                           dislike_int,reproach_int/3,
                           resentment_int/5);

  ;; bf_value is the current value of the unfriendly
  ;; behavioral feature -- agent indicates the version of
  ;; the feature directed at agent -- bf_value is 0 if
  ;; there is no current unfriendly feature directed at
  ;; agent
  bf_value := current_feature_value(UNFRIENDLY,agent);

  ;; if value and bf_value are not equal (i.e. the actual
  ;; feature value is not what it should be) -- without
  ;; this test, the demon fires repeatedly
  (value != bf_value)
 )

;; The righthand side of the rule creates a feature with
;; the new intensity value. If this feature already exists
;; with a different intensity value, it is overwritten.
RHS:
 add_feature(UNFRIENDLY,agent,value);
```

Here is the standard emotion-to-feature map provided with Em. It requires 32 hap rules and roughly 775 lines of hap code. I have developed this mapping from my experiences creating agents and it has shown itself to be reasonably useful as a starting point for creating emotionally expressive agents. I have modified the map for every agent I have built, but I have found it much easier to start with this map and modify it than to work from scratch. For example, I describe Melvin's feature map below, which is scaled down and somewhat modified but required no new feature map rules.

There is no psychological foundation to this mapping; there are practical and artistic considerations, however, that I described in section 5.1.2.

```
cheerful := joy
friendly := joy/2
friendly(a) := like(a)/2 + gratitude(a)/2 + joy/5 +
               admiration(a)/3 + happy-for(a)/5

unfriendly := distress/3 + anger/2
unfriendly(a) := dislike(a) + anger(a)/2 + distress/3 +
                  reproach(a)/3 + resentment(a)/5

generous := joy/2
generous(a) := gratitude(a) + joy/2
grateful(a) := gratitude(a) + joy/2

sulking := distress
withdrawn := distress

aggressive := if (anger>0 AND fear>5)
               then anger + fear
               else if (anger>0)
                     then anger
                     else 0
aggressive(a,g) := if (anger(a,g)>0 AND fear(a,g)>5)
                     then anger(a,g) + fear(a,g)
                     else if (anger(a,g)>0)
                           then anger(a,g)
                           else 0
aggressive(a) := if (anger(a)>0 AND fear(a)>5)
                     then anger(a) + fear(a)
                     else if (anger(a)>0)
                           then anger(a)
                           else 0
aggressive(g) := if (anger(g)>0 AND fear(g)>5)
                     then anger(g) + fear(g)
                     else if (anger(g)>0)
                           then anger(g)
                           else 0
```

```
defensive := fear
defensive(a,g) := fear(a,g)
defensive(a) := fear(a)
defensive(g) := fear(g)

temp-avoid(a) := max (fear(a), dislike(a))
temp-approach(a) := max (hope(a), like(a))
avoid(a) := if temp-avoid(a) > temp-approach(a)
            then temp-avoid(a)
            else 0
approach(a) := if temp-approach(a) > temp-avoid(a)
               then temp-approach(a)
               else 0

proud := pride + gratification/2
ashamed := shame + remorse/2

gloat(a) := gloating(a)
gloat(a,g) := gloating(a,g)

console(a) := pity(a)
console(a,g) := pity(a,g)

congratulatory(a) := happy-for(a)
congratulatory(a,g) := happy-for(a,g)

contempt(a) := resentment(a) + reproach(a)
awe(a) := admiration(a)

anticipation := hope
anticipation(a,g) := hope(a,g)
anticipation(g) := hope(g)

;; recall that positive and negative are default nodes
;; in the emotion type hierarchy
good-mood := if (positive > negative)
               then positive
               else 0
bad-mood := if (negative >= positive)
               then negative
               else 0

{0 is very lethargic -- 10 is very excited}
energy := 5 + (min(10,(joy + fear + hope + anger
                            + pride + frustration)) / 2)
            - (min(10,(distress + shame)) / 2)
```

For some agents and worlds, the complete feature mapping will not need to be used in its entirety. By limiting the number of features that an agent has, the artist can simplify the process of building character-specific ways of expressing the features. For example, here's the mapping that is used for Melvin in *The Playground*. It requires 13 hap rules and 310 lines of hap code. When building Melvin's negotiation behavior, for example, I did not have to consider how it should change based on the presence of a *proud* feature since Melvin will never have such a feature. As mentioned previously, features can also be created by non-Em behaviors. In Melvin, for instance, his behavior for making friends creates a *friendly* feature. These do not show up in this mapping, which represents only the effects of emotions on behavioral features.

```
sulking := if (distress > joy)
            then distress
            else 0
withdrawn := if (distress > joy)
               then distress
               else 0

defensive := fear
defensive(a,g) := fear(a,g)
defensive(a) := fear(a)
defensive(g) := fear(g)

cheerful := joy
generous(a) := gratitude(a) + like(a)

temp-avoid(a) := max (fear(a), dislike(a))
temp-approach(a) := max (hope(a), like(a))
avoid(a) := if (temp-avoid(a) > temp-approach(a))
            then temp-avoid(a)
            else 0
approach(a) := if (temp-approach(a) > temp-avoid(a))
               then temp-approach(a)
               else 0

console(a) := pity(a)
congratulatory(a) := happy-for(a)

good mood := if (positive >= negative)
               then positive
               else 0
bad mood := if (negative > positive)
               then negative
               else 0

anticipation(a,g) := hope(a,g)
```

```
friendly(a) := max(0, like(a) - dislike(a))
unfriendly(a) := max(0, dislike(a) - like(a))
```

This mapping does not include the entire set of mapping rules because they aren't necessary for Melvin. Also, some of the rules that have been adopted have been modified to better suit Melvin. For example, the default *good-mood* and *bad-mood* rules favor Melvin being in a good mood, unlike the default rules. Despite these differences, Melvin's emotion-to-features map requires only about 20 lines of new or modified code. The rest is taken from the standard set of feature mapping rules.

Adding new types of features not in the default mapping is reasonably simple. To add a new feature, the artist just needs to write a new rule that adds a feature of a particular type. The Em architecture provides the mechanisms to store and query all such features. For instance, if I wanted a new behavioral feature of type *insulting* to be used in Sluggo when he was angry, I would write a new rule (like the one in Figure 5-3) that creates a feature of type *insulting* towards whomever he is angry at; the intensity would probably be some function of his anger towards that agent.

5.2 Emotional Expression in Em

Once the set of features has been computed, they need to affect the agent's behavior. As I discussed in the introduction to this chapter, I have tried to take a cue from traditional arts, where the artists talk about the importance of emotion permeating the character: how they move, how they act, how they speak, what their face looks like, what their body stance is, and much more [Thomas81]. Based on these suggestions, I provide artists a large set of mechanisms for expressing emotions in their characters. As more characters are designed and interactive drama progresses as a field of art, I believe that artistic techniques for using mechanisms like the ones I describe will develop, just as they have in other media.

I argued in section 1.4.2 that using a broad architecture is appropriate for creating believable agents. The expression of emotions provides additional evidence for this being a useful approach. In the rest of this section, I will present a broad set of mechanisms that artists have at their disposal when creating emotionally expressive agents using Tok. These mechanisms allow artist to create characters where the emotional expression permeates the character. I will also provide examples to illustrate the kinds of situations where each of these techniques can or has been used.

This section contains five parts, each corresponding to a part of the Tok architecture that the emotional state can affect: the action system, natural language processing, the emotion system, the social system, and the body state. Throughout these sections, it is important to remember that none of the effects described are hard-coded into the Em system. They are all options available to artists to use as they see fit. Since the expression of emotions needs to be character-dependent and I don't want to stifle to creativity of the artists, my approach is to support a large number of ways of expressing emotions without forcing any of them.

Since the art of using these mechanisms still needs to develop, it is hard to say which of these techniques will prove extremely useful and which will be less useful. In an architecture like Tok, though, I believe that these mechanisms provide the primary ways of creating the kinds of emotional expression that artist say are needed.

These ideas come from a variety of sources, including the emotion literature (e.g., [Elliott92], [Oatley92], [Frijda86]), AI (e.g., [Carbonell79]), the arts (e.g., [Thomas81]), and personal experience. The arts suggest that all of these ways of expressing emotions are important, though they don't make such suggestions at the low level of description that I use here. One of the most detailed examinations of emotional expression is given by Gilboa and Ortony in [Elliott92]. They postulate a set of "action responses" for emotional expression. The set of mechanisms that I describe below allows artists to duplicate all of the Gilboa and Ortony action responses as well as expand on their list to include a large number of new ways to express emotions.

5.2.1 The Action System

I break the influence that emotions[1] can have on the action system into three parts: the relationship between emotions and goals, emotions and plans, and emotions and actions.

In Tok, emotion structures (via the behavioral features) can influence goals in a number of ways. I have personally found the first two the most useful so far.

1. Emotions can cause the addition of new goals. This is accomplished with demons that match on conditions that are partially or totally emotion based.

1. As I have described, emotions do not directly have any affect on the action system or any other system. They act only through the behavioral feature system. In this section, however, I will often refer to emotions and the emotional state as affecting aspects of the architecture, which is not technically correct. Where I believe there is the chance of confusion, I will be more specific and refer to the behavioral features.

Example: Cathy got so mad at Doug that she vowed to get revenge. Or: Cindy was so happy after passing her test that she decided to go out dancing. In both cases, the agent has a demon that fires in particular emotional states and creates new goals for the agent to pursue.

One way I used this mechanism was for Sluggo to create a new goal to beat up another character when he got sufficiently angry at them.

2. Emotions can cause the priority of a goal to rise or fall. The original Hap language only allowed static priority values, but I extended the language to allow functions as well and wrote three standard priority functions that take the features *defensive(goal)* and *aggressive(goal)* into account when computing the priority of the goal. The simplest of the three standard functions is a simple linear function from some low-priority value when the defensive and aggressive features for a goal are 0 to some high-priority value when one (or both) of the features is 10.

```
priority(goal) =
   low-priority +
   ((high-priority - low-priority) *
      (1/10 * max(defensive(goal), aggressive(goal))))
```

   The other two functions are similar in nature, but are step functions instead of linear functions, so the artist can have more control over when the priority increases and by how much.

   Example: In *Robbery World*, the gunman's goal not to be killed initially has a lower priority than the gunman's goal to hold up the convenience store. When the goal not to be killed is threatened, however, the gunman gets *angry* at the officer and *fears* that he will be killed; these emotion structures are mapped into *aggressive* and *defensive* features; and the features lead to the goal being reassigned a higher priority. If the priority surpasses the priority of the goal to hold up the store (which happens when the aggressive or defensive feature is 9 or 10), the gunman will stop holding up the store and defend himself either aggressively (by shooting at the police officer or taking a hostage) or defensively (by running or giving up).

3. Emotions can make it easier for goals to succeed. Some goals may not have strict success criteria and emotions may make a goal succeed more easily. This is accomplished through success-tests.

   Example: Paul wrote his essay in a bad mood and it showed, but he really didn't care to make it better. In this case, Paul would have a goal to write an essay and a behavior to achieve that goal that would continue to work on the essay until the associated success-test for the goal determined the essay was of acceptable quality. This test could take Paul's behavioral features into account.

4. Emotions can make it harder for goals to succeed. Again, this is mostly for goals without strict success criteria where emotions can increase the difficulty of judging the goal successful. This is also accomplished with success-tests.

   Example: Gary really liked Beth and wanted to make the picture he was painting for her perfect—he was even more obsessive about the tiniest details than usual. As in #3, Gary would have a goal with a success-test to indicate when the goal was accomplished that takes behavioral features into account.

5. Emotions can make it easier for goals to fail. One way to accomplish this in Hap is to make the context conditions of the plans to achieve such a goal sensitive to emotions.

   Example: When Nate fell a little behind in school, he got so depressed that he figured he could never catch up, so he just gave up trying. In Tok, Nate would have a goal to do well in school and a behavior (or set of behaviors) for achieving that goal. These behaviors would have context-conditions associated with them that would indicate when the behaviors no longer made sense to use. These context-conditions could take Nate's behavioral features into account and be more likely to indicate failure when Nate was distressed.

6. Emotions can make it harder for goals to fail. This can be accomplished by making the context-conditions of the behaviors for this goal aware of the emotional state.

   Example: Even though it was clear to everyone else that the children had been killed, their mother refused to accept it and pressed for the investigation to continue. As in #5, the context-condition for the mother's behaviors might be less likely to indicate that a behavior was no longer worth using because of input from the behavioral features.

7. Emotions can affect the importance of a goal, making its success or failure a matter of less or more concern to the agent. I have made the importance of a goal an arbitrary function that can take emotional information about the goal into account.

   Example: The more angry Phil got at Cathy, the more he wanted to get revenge. In this case, Phil's goal to get revenge has an importance based on his anger towards Cathy. Notice that the greater the importance of the revenge goal, the more intense his emotional reactions associated with the goal will be, such as joy when he successfully enacts his revenge.

Here is a list of ways that emotions can affect behaviors in Hap. I have personally found the first three the most useful.

1. When choosing a behavior for a particular goal, emotions can influence the choice. For example, emotions can make some behaviors potential candidates that would otherwise not be chosen, they can make a possible candidate more likely, they can make a behavior automatically chosen, they can decrease the likelihood of a behavior being chosen, and they can make an otherwise potential behavior no longer an option. All of these effects are created by modifying the preconditions of the behaviors to take emotional information into account.

   Example: Dan was so mad that he kicked the door open on his way out, knocking it clean off its hinges. Or: Lilly would normally have gone for the complicated combination shot, but she was so nervous that she took the simpler straight shot on the 2-ball.

   One way that I have used this mechanism is to have Melvin respond to trade offers differently based on his emotions. For example, if he is happy, he will trade and be generous. If he is neutral, he will trade, but not be especially generous. If he is angry or scared, he will not trade at all.

2. Emotions can cause a shift from an ongoing behavior to a different behavior. The shift is accomplished by having a context-condition on the first behavior that checks for emotional conditions for failing. When this behavior fails, a new behavior for the goal can begin. By using the mechanisms described in #1, which behavior is taken up can also be determined taking the emotional state into account.

   For example: Doug was so upset that he decided to walk home from the party, but half a mile later he had calmed down enough that he decided to go back and ask Zeke for a ride.

   I use this mechanism in the gunman. He has a number of plans for protecting himself. One is to take the cashier hostage; another is to give up peacefully. He will take a hostage if he is not overly scared. If he should become scared after taking the hostage, he will release the hostage and give up.

3. Emotions can affect the choices about which agent or object to use in a behavior when there are multiple possibilities.

   Example: Rich's favorite band, *Morp, Morp, Morp*, was in town and he was looking for someone to go with him. He decided to ask Scott instead of John since he was sill mad at John for breaking a promise. In this case, Rich has a behavior that would work with any of a number of people and his choice of which person to use is based on input from his behavioral features.

I use this mechanism in Melvin to have him prefer to trade with people that he is feeling friendly towards.

4. Emotions can affect how many times a behavior is attempted before giving up on it.

   Example: When Ted is happy, he'll practice free-throw shooting all afternoon. When he's angry, he gives up after about 10 shots.

Finally, here are some of the ways that emotions can affect actions. I have found all of these approaches to expressing emotions useful.

1. The style with which an act can be performed can change, so generic acts like walking across a room can be modified to be stomping, shuffling, or any of a number of variations on the same action.

   I use this mechanism in many of the characters to affect, for instance, how the character speaks (e.g., angrily, excitedly).

2. Acts can be viewed as simple behaviors in the sense that emotions can affect them in many of the same ways they affect behaviors, such as affecting action choice and persistence in repeating an action.

   I use many of these mechanisms in the characters I have created. For instance, when responding to a greeting, Melvin will say, "Greetings" if he's in a neutral emotional state, and "Salutations, Vulcan ambassador" if he's in a *good-mood*.

3. Acts can be viewed as simple goals in the sense that emotions can affect them in many of the same ways that they affect goals, such as adding new actions and affecting the priority, importance, and success criteria of acts.

   I use many of these mechanisms in the characters I have created. For example, then Sluggo is *angry* he will sometimes interrupt what he is doing to glare at the agent he is *angry* at.

5.2.2 The Body State

The Em system (for characters in text-based Oz worlds) comes with a standard set of rules that map the current features into changes in the body appearance (e.g., tensing), facial expressions (e.g., frowning), and face color (e.g., pale). These body-state features are modeled as variables that are described to the user in the text description of the world. For instance, agents have a facial-expression attribute that can be set to a facial expression. When this value changes, the change is described to the user.

As I argued previously, it is often confusing for the user when other characters express conflicting features concurrently. Because of this I occasionally use the notion of the current dominant feature to determine expression. The current dominant feature, for the purposes of these default rules, is the rule with the highest intensity other than *good-mood* or *bad-mood*. However, if no other feature is above intensity 3 but *good-mood* or *bad-mood* is, then *good-moo*d or *bad-mood* is the dominant feature. This encourages specific emotional displays when possible, but chooses a general display over no display of emotions when there is an appropriate general response. For example, if the agent's most intense feature is anger, the agent's body will tense and the agent will scowl and turn red in the face.

Table 5-2 describes the defaults I use to map from features to body state.[1] These rules can be added to, modified, or discarded as the artist desires. In fact, as I have suggested repeatedly, I expect that artists will want to change many of these rules to create unique, personality-rich characters.

Tok also models simple adrenaline-like effects. Agent have a behavioral feature denoting how much *energy* is currently available for things like moving quickly and lifting heavy objects that is affected by the emotional state.

1. The body effects described in table 5-2 are modeled by changes in the set of attributes of the agent. In text-based Oz systems, an agent can take one action per turn but have multiple attribute changes. Unfortunately, the current Oz natural language narration system for text-based worlds makes it difficult to describe attribute changes with two-part features, such as smiling-at. Because of this, the italicized agent-directed features in table 5-2 would have to be implemented as actions instead of state changes, which would eliminate the ability to perform another action that turn. I typically modify this mapping and settle for the non-directed version of the feature, which gives the right idea without ruling out other actions. For example, aggressive(agent) leads to scowling, but not at any particular agent.

TABLE 5-2 Default Mapping from Behavioral Features to Body State Effects.

Dominant Feature	Facial Expr.	Face Color	Body State
Bad-Mood, Sulking, or Withdrawn	Frowning	*default*[a]	*default*
Good-Mood, Cheerful, Grateful, Gloating or Friendly	Smiling	*default*	*default*
Friendly(agent), grateful(agent) or gloating(agent)	*Smiling-At (agent)*	**default**	**default**
Aggressive	Scowling	Red	Tense
Aggressive(agent)	*Scowling-At (agent)*	*Red*	*Tense*
Unfriendly	Scowling	*default*	*default*
Unfriendly(agent)	*Scowling-At (agent)*	**default**	**default**
Defensive (intensity <= 5)	*default*	Pale	Tense
Defensive (intensity > 5)	Bug-Eyed	Pale	Trembling
No Features or other	*default*	*default*	*default*

a. *default* means that the agent will look “normal” in this regard.

5.2.3 Natural Language

Natural language understanding and generation problems are still being studied for believable agents (see [Loyall96] and [Kantrowitz96]) and I don’t have a lot to say about this topic. However, because the language understanding and generation are being done in Hap, I expect many of the same techniques that are used to affect processing in the action system to apply here.

For example, in section 5.2.1, I discussed a number of ways that emotion can be expressed through physical action. These same sorts of mechanisms should also apply to low-level natural-language decisions, like lexical choice. The mechanisms that I described to handle emotional expression in goals and plans should also apply to higher-level language decisions, like whether to answer in speech or gestures and whether to speak loudly or softly. In the agents that currently use language, I use emotions to choose a style of speaking and to choose between speech and gesture. Since the language generation capabilities of these agents are currently restricted to templates, however, lexical choice is simulated by choosing between a variety of templates that have been written to express emotions through word choice.

For language understanding, emotions can influence, among other things, how much attention to pay to a particular speaker and how to interpret what they are saying. I suggest using emotions to affect the priority of language-processing goals to achieve the former effect. I suggest using emotional influence on inferencing to achieve the latter. I have used the former technique to build characters that ignore other characters when in certain emotional states, which is a simple version of this idea. I have not used the latter idea at all. Once a deeper Hap-based language understanding system has been built[1], the interactions between emotion and language understanding will be able to be explored further.

5.2.4 Inferences

Emotions can color the way an agent thinks and reasons. Only limited work on generic inferencing in Tok agents has been done, with the focus being primarily on special-purpose inferencing mechanisms required for particular agents in particular situations. Because of this, results in this area are limited. Nonetheless, emotions and special-case inferencing have been integrated through the standard Hap mechanisms in a number of characters.

For example, in *Robbery World*, the gunman needs to infer how violent and trustworthy the police officer is. This is used to determine whether or not to turn himself in. If the gunman is angry at the police officer, this will affect his judgements about the officer, as will things like the fact that the agent is a police officer. This is accomplished by checking the aggressive feature directed towards the officer when making this inference.

5.2.5 Emotion Processing

Emotions can affect emotions. For instance, in Em, emotions are often based on appraisals of things like the likelihood of a goal failing, which are essentially inferences. As just noted, emotions can affect inferences of all kinds, including appraisals related to emotion generation. These inferences can include judgements about such things as likelihood of goals succeeding or failing and the responsibility for a goal success or failure.

As mentioned in the action section, the importance functions associated with goals can also take emotional information into account, which will change the intensities of the emotions based on those goals.

1. The one I use is based entirely on keyword matching.

It is also possible to write emotion effects into emotion generators directly and have emotions directly affect both the types and intensities of emotions being generated. Furthermore, emotions can be used to affect when and how quickly emotions decay.

Because emotions are so tightly tied up with the other parts of the architecture, there will also be many indirect emotional effects. For example, if an emotion makes some of the agent's goals more likely to fail, the failure of those goals may result in new emotions. An artist that uses the standard emotion generation system provided with Em does not have to do any additional work in cases like this because the rules the generate emotion structures based on goal failures (and many other situations) are already in place.

5.2.6 Social Factors Affected by Emotion

There are a wide variety of things that can legitimately fall under the heading of "social factors." They include things like what relationships the agent has with others, what the agent thinks about other agents, and how the agent interacts with others. I will discuss just a few that are supported by Tok. I have found that being able to express emotions in social ways is important for creating believable social agents as well. This will be discussed in greater detail in Part II of the thesis.

- **Social Inferences.** Social inferences (e.g., inferring that two agents are friends) are simply inferences about social situations. Social inferences are affected by emotions using the same mechanisms that non-social inferences use.

  In *Robbery World*, the gunman's inferences about the police officer (such as how trustworthy he is) are based on the gunman's emotional state.

- **Attitudes about other agents.** Attitudes agents have about each other will often change based on emotions. Example: Wally was always so nice to Glinda that she eventually came to like him—at least a bit.

  In the Tok agents created so far, I have used demons that match on anger and gratitude in the agent's current emotion state, and use those to change the like and dislike attitudes towards those agents. Attitudes have a threshold value associated with them, which helps determines when emotions will change attitudes and how much. The formula Em uses by default is:

  ```
  change = max(0, (gratitude-intensity - threshold)/2)
  new-like-att-intensity = old-like-att-intensity + change
  ```

  A similar formula is used for dislike and anger. By default, Em also allows only one of the like or dislike attitudes to be above 0, whichever is greater.

  For instance, agent A has a like attitude for agent B with intensity 7 and threshold 8 (they are old friends). Agent A becomes grateful to Agent B with intensity 6. A's attitude towards B does not change. If the attitude were intensity 2 and threshold 2 (they were passing acquaintances), then the like attitude would change to 4 because of the gratitude.

In *The Playground*, Melvin uses this mechanism to change his attitude towards the player if the player is mean to him.

Em does not have rules for changing attitude thresholds since most interactive drama interactions are fairly short. If a story were to extend over a long period of time, the artist would have to write rules to change attitude thresholds.

- **Interpersonal relationships.** Dynamic relationships are a very important part of many stories. One way that relationships change is through emotions. I typically choose to model relationships as type-value pairs, such as (friend 8), which indicates close friends. (The value ranges from 0 to 10.) Artists can write Hap rules to change these structures based on the agent's emotional state. There are no default rules for making these changes.

  Example: After Jill was elected class president, she became insufferable. Eventually even Mary stopped hanging out with her because she was always unpleasant to be around. In this case, Jill's friend Mary will have a structure that represents her relationship with Jill. This structure is changed when Mary's attitude towards Jill changes from like to dislike, as just discussed. Note that Jill might still believe that she and Mary are friends since she has her own structure to represent their relationship.

- **Social norms.** Agents typically have a certain amount of cultural knowledge to know how to act appropriately. Emotions can influence such behaviors both intentionally and unintentionally. Example: Rick was so flustered that he walked out of the restaurant without paying.

  In Tok, behaviors are written to take social norms into account. Emotions modulate those behaviors using the techniques described in the section on action above. For instance, Rick's behavior for paying in a restaurant could be ignored in favor of a more emotionally important behavior with a higher priority, causing him to walk out of the restaurant without paying.

  In *The Playground*, Sluggo will be very rude and break a number of social conventions if he is feeling aggressive.

- **Social roles.** What role an agent plays in society should affect how they act. Emotions can affect what role an agent is playing at any given time and how they act relative to that role. Example: When Jack finally met the murderer face-to-face, fourteen years of police training went out the window and he lost his cool like his partner had never seen before.

Social roles can be handled in at least two ways with Tok. First, behaviors can implicitly take the role of the character into account. In this case, the behaviors need to take emotions into account in the usual ways. For example, if an artist is creating a police officer character, that character's behaviors are written knowing the character's role and are suitable for a police officer. These behaviors are affected by emotion just like any other behavior the agent has would be.

Second, the role can be explicit. This can be used when an agent plays different roles depending on the situation, but has the same set of underlying behaviors. For instance, if the police officer character is not always on the beat, then the artist might want to write behaviors that change based on whether the character is on the beat or not. The artist creates an explicit role structure for the agent that can change over time (e.g., when on the beat or not) and behaviors match on this structure.

In this explicit case, there are two ways to have emotions and roles interact. The first is similar to the implicit-role case: build emotions into the behaviors. The second is to take emotions into account in the rule that determines the current role(s) the agent is taking on. For example, if the officer were really angry, he might "forget" that he was a police officer on the beat and not act according to police procedure. A demon that sets the role structure to the proper value would be responsible for the agent "forgetting" his role in emotional circumstances.

- **Social goals and behaviors.** Emotions can affect social goals and plans in many of the same ways they affect goals and plans for physical actions. Example: Sal was so mad that he vowed to get revenge, but he was a bit too intent on getting immediate revenge and didn't think his plan out very well.

  Social goals and plans are very much like any other sort of Hap goal or plan, except that it might often be even more important to make sure that they are modulated by emotions. The same mechanisms used in other Hap goals and plans can be used here as well. This topic will be discussed in greater detail in Part II.

5.3 Summary

In this chapter, I described a two-part mechanism for expressing emotions in believable agents.

- I motivated and described Em's behavioral feature system, which provides a level of indirection between the emotion structures and their effects on the agent's processing.

- I described the default set of behavioral features and the default emotion-to-feature map that are provided by Em. Both are influenced by the artistic constraints of the task. I also showed the emotion-to-feature map used by Melvin in *The Playground* to show that I was able to reuse a significant amount of code while maintaining the ability to create a personality-specific map.

- I described many of the ways that emotions can affect the processing of an agent. Where possible, I described default mechanisms that can be used by artists; these include ways of expressing emotions through the body, ways of changing a character's attitudes about other characters based on emotions, and three default functions for modifying goal priorities based on emotion. Where I couldn't provide default solutions, I described the architectural capabilities Em and Tok provides artists for expressing emotions through many of the agent's subsystems. This approach provides artists flexibility to create characters with distinctive personalities where emotional expression permeates the agent's behavior.

CHAPTER 6 Validation

6.1 Emotion Claims

In the previous three chapters, I have described a set of tools, collectively called Em, that are designed to support artists in the creation of believable emotional agents. In this chapter I will present evidence that Em is successful in that it does support the creation of believable emotional agents.

Before showing that Em is effective, it is also important to note the claims I am not making:

1. Not all believable characters must have emotions. Spock and the Terminator are examples where emotions are explicitly not appropriate. I claim that artists will almost always want their characters to seem emotional, but that this is not necessary for creating successful characters. I base this claim on work in other media where emotions are an important element of many quality characters.
2. I do not claim that Em is unique. There may be many other ways of achieving the effects that Em achieves. I claim only that Em is *a* way to approach the problem and that it is sensitive to the true needs of artists in ways that much existing AI research is not.
3. I do not claim that Em is easy to use or completely flexible. In the descriptions of the Em architecture, I described a number of design decisions that were made to increase flexibility (such as user-definable emotion generators, decay functions, and emotion types) but I do not claim that Em is completely flexi-

ble. Also, I described the default emotion system provided with the Em architecture, which is a means of making it easier for agent builders to set up specific emotion systems. Nonetheless, this is still a difficult task and there are probably many ways that the task could be further simplified.

4. I do not claim that Em ensures that the agents will be either emotional or believable. That is up to the artist, the user of the tool.

6.2 Validation of the Em System

In this section I will provide evidence that the Em system makes it practical to create agents that both appear to be emotional and are believable.

It is important to note that I am showing that Em supports creating characters that appear to be emotional *and* are believable. It would be fairly simple to create characters that just added emotional adverbs with every action (e.g., “Sluggo spits angrily,” “Sluggo gets in the tree house threateningly,” “Sluggo looks at a baseball card aggressively”) that might make users judge the characters to be emotional, but if these adverbs are not controlled, it is likely that the characters would not be very believable. Also, if characters express emotions that are inconsistent with their personality or the situation, it is likely that the user’s suspension of disbelief will be broken, even if the characters seem emotional. Therefore, it is important to show characters that are both emotional and believable.

This claim is validated by providing an example of the Em tool being used to build an agent that users feel is both emotional and believable. I built seven agents for this thesis, so validating the success of Em became a matter of evaluating at least some of these characters through user studies.

I presented a set of users with two versions of the Playground simulation. One version is the normal version I have described with Melvin and Sluggo. The other version has Melvin replaced with an Em-less version of the same character, named Chuckie. By doing this, I am able to make claims about the role of Em in creating the appearance of emotionality and believability.

Removing the emotions of a character is not a completely trivial task since the emotions tend to be very tightly integrated with the rest of the character. In fact, I couldn’t have run such tests with the gunman in Robbery World because taking out his emotion system completely breaks him; that is, he will not perform *any* actions. Because Melvin is not as emotional a character to begin with and because the world he inhabits is generally a less emotionally intense one, I was able to create a version of Melvin without emotions that works reasonably well.

6.2.1 Methodology

In order to produce well-formed studies, I sought the assistance of Professor Sara Kiesler of Carnegie Mellon's department of Social and Decision Sciences. (Of course, I take responsibility for any flaws in the studies.)

17 users were given introduction and instruction sheets to read. (These sheets are included in Appendix B.) There are two variations of the system; one includes Sluggo and Melvin; the other includes Sluggo and Chuckie, who is the same as Melvin but without an emotion system. Users interacted with each version of the system for 20 minutes or until a natural conclusion was reached (e.g., Sluggo beats up the user). I alternated the order in which the two versions of the system were presented—8 of the 17 users interacted with Chuckie and then Melvin; the other 9 interacted with Melvin first. After interacting with each system, the user was asked to fill out the questionnaire in Appendix B. The users were not shown the questionnaire ahead of time, so they were not aware of the kinds of questions they would be asked—in particular, they did not know that they were to look for emotions in the characters.

The biggest problem with the test is that the pool of users is not very diverse. I advertised around the Carnegie Mellon campus, which almost necessarily provided a set of users with an above-average education level. Also, because of the makeup of the CMU student body, 7 of the 17 users are from computer science or related engineering fields. The age range is 22 to 41. The male-to-female ratio is 9:8. Because of the bias of the pool of users, it is impossible to make general claims that all users will respond similarly to our pool. I hope the technical competence and educational level of the users means that they were *more* critical than the general population, but that is only speculation.

6.2.2 Results

In this section I will discuss the results I obtained from the study, including the specific questions I asked, the data I gathered, and some statistical evaluation of the data.

Emotions

Users felt that Melvin was more emotional than Chuckie. Users were provided a scale of integers from 1 to 7 (called a Likert scale) where 1 was labelled "unemotional" and 7 was labelled "emotional." The users were asked to, "circle the number that indicates your impression of X." This questions was asked for each character, where X was the character's name.

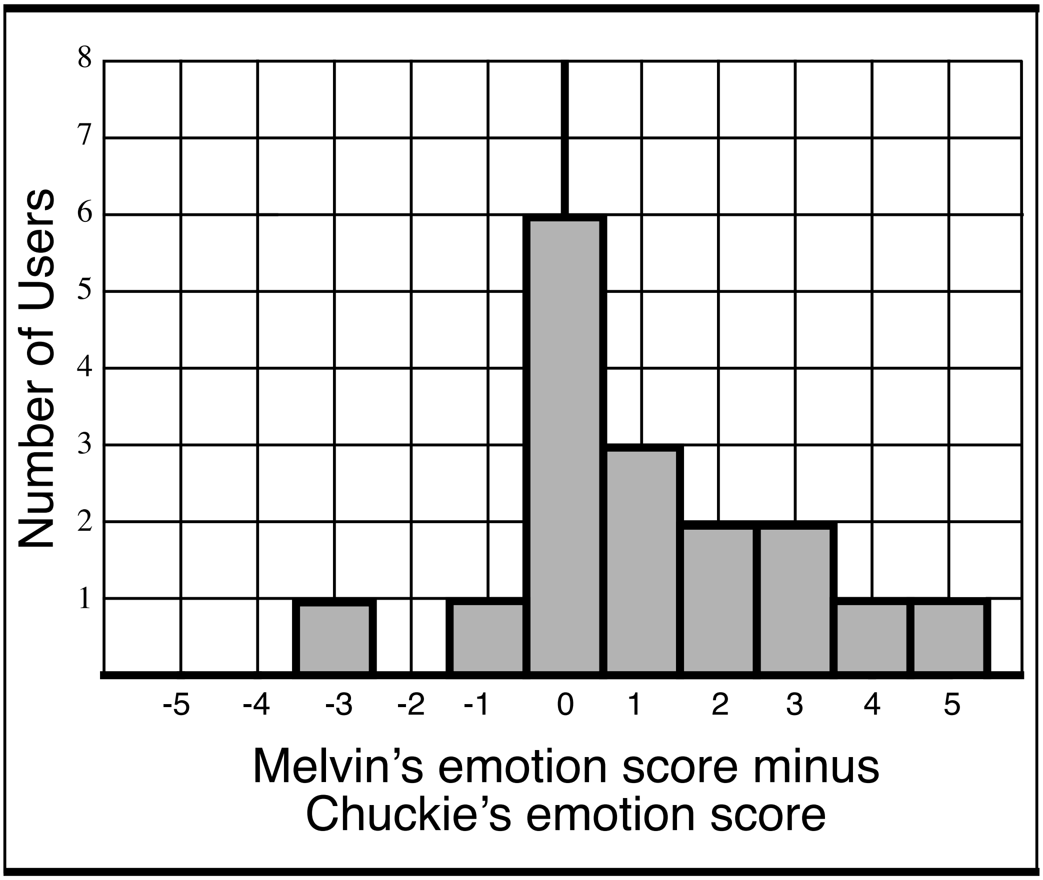
The mean score for Melvin was 4.41. The mean score for Chuckie was 3.35. Also, 9 of the 17 users felt Melvin was more emotional; 6 felt they were evenly emotional; 2 felt Chuckie was the more emotional. The fact that 6 felt they were equally emotional may be a side effect of the fact that Melvin is not an overly emotional character, so his emotions are not obvious in all interactions.

In order to analyze this data more rigorously, I used a statistical procedure called a t-test. A t-test is used to make statistical claims about the actual mean of some population given the results of some sample of the population. For instance, if I know that the mean of a sample is x, and I choose some statistical significance that I desire, say 95%, I can determine an interval around x (i.e. $[x-y, x+y]$) that contains the actual mean score of the population with confidence of 95%. I can also find one-sided intervals, which allow me to make claims like, the actual mean is greater than some value with a given confidence. The t-test is used in this case because it does not require the standard deviation of the larger population to be known; also it is reasonably robust in cases where the distribution is not normal as long as the sample size is at least 15, the distribution is not significantly skewed, and there are no strong outliers. Also, the presence of outliers means that the claims that I am able to make are not as strong as they otherwise could be, not that they are wrong [Moore89].

To compare the emotion scores of Melvin and Chuckie, I applied a t-test to the difference of the scores each user gave Melvin and Chuckie. In other words, for each user, I determined the difference in emotion scores given to Melvin and Chuckie. If each user gave them the same scores, then the mean of the distribution would be 0. This actual sample has mean of 1.06 and standard deviation of 1.95. This data confirms the hypothesis that the mean of the population is greater than 0 with confidence >98%. In other words, I can show with high probability that, given this sample, if I were to sample the entire population, Melvin's mean score for emotion would be greater than Chuckie's.

The data for the differences in emotion scores is provided in histogram form in Figure 6-1.

FIGURE 6-1 **Melvin vs. Chuckie: How emotional?** This histogram shows the differences in how Melvin and Chuckie were scored on a scale from 1 (unemotional) to 7 (emotional). The total number of users was 17. For example, this chart shows that 2 of the 17 users scored Melvin 2 points higher than Chuckie on the 1-7 scale.



Believability

As mentioned above, the fact that Melvin is more emotional than Chuckie is important, but not if it comes at the expense of believability.

To find out how Melvin and Chuckie compared in terms of believability, the users were asked a number of questions related to believability.

- "How good a character is X?" where X was either "Melvin" or "Chuckie." They were given a scale from 1 (awful character) to 7 (great character).

- "Did X have a clearly conveyed personality?" They were given a scale from 1 (not at all) to 7 (very much). One of my goals is to make emotions fit within (and, hopefully, enhance) the character's personality. If the emotions make the character's personality less clear, I would fail in this goal. For example, it is possible to imagine a character with randomly generated emotions who seemed emotional, but whose emotions were inconsistent with other aspects of the personality, which made the personality of the character less distinct. Because artists tell us that a clearly defined personality is vital to believability, it is important that characters' emotions do not detract from the clarity of their personalities.

- "Did X do anything to disrupt your 'suspension of disbelief'?" They were given a scale from 1 (never) to 7 (all the time). In a good movie, the members of the audience forget that they are watching a movie because they are caught up in the story and characters. This is what is meant by the user's "suspension of disbelief."[1] If I was able to make Melvin seem more emotional, but his emotions came at the expense of users' ability to suspend their disbelief, I would have been unsuccessful.

Let's evaluate the data for each of these cases.

First, does adding emotion to a character detract from the quality of the character? In the study, I found Melvin's mean quality of character score was 4.94 and Chuckie's was 4.29. 10 of 17 users found Melvin to be a better character, 5 thought Melvin and Chuckie were equal in this regard, and only 2 thought Chuckie was the better character.

To test the statistical significance of these results, I applied a t-test to the values derived by subtracting Chuckie's quality of character score from Melvin's. This sample has a mean of 0.65 and a standard deviation of 1.80. This means I am

1. I did not explicitly define this term on the questionnaire. I gave this definition to the few users who were not familiar with the phrase.

able to claim that the actual mean is greater than 0 with >90% confidence. If the outlier at -4[1] is thrown out, this confidence level jumps to >99%. That is, with somewhat high confidence, I can claim that users really do find Melvin to be a better character than Chuckie. Figure 6-2 shows the various values of these difference scores.

To get an idea of the level of influence emotion has on the quality of character in the study, I mapped quality of character scores for all three characters (i.e., I included scores given for Sluggo in this test) against their emotion scores, as seen in Figure 6-3. I also performed a linear regression[2] and found that the line best fitting this data has the formula: $F(x)=0.33x+3.37$. The positive slope of 0.33 indicates a positive correlation between emotion and quality of character. It also means that for every 3-point increase in the emotion score of a character, the overall quality of the character increases by roughly 1 point. The coefficient of correlation in this case is 0.46 on a scale of 0 to 1, indicating that scores for emotion and quality of character have a rather high correlation [Moore89][3].

1. It is standard statistical practice to throw out data that is clearly skewed if there is a reason for the skewing unrelated to the test in question. In this case, a coding bug led to problems that made Melvin not respond to the player for a large portion of this interaction.

2. Linear regression does not assume that the underlying population has any specific distribution (e.g., normal) [Moore89].

3. To get an idea for what this value means, a study relating high school and college success of computer science students at a large midwestern university found that the relationship between math and verbal SAT scores for these students were correlated with coefficient 0.46 [Moore89].

FIGURE 6-2 **Melvin vs. Chuckie: How good a character?** This histogram shows the differences in how Melvin and Chuckie were scored in terms of quality of character. The scale ranged from 1 (awful character) to 7 (great character). The total number of users was 17. For example, this chart shows that 6 of the 17 users scored Melvin 1 point higher than Chuckie on the 1-7 scale.

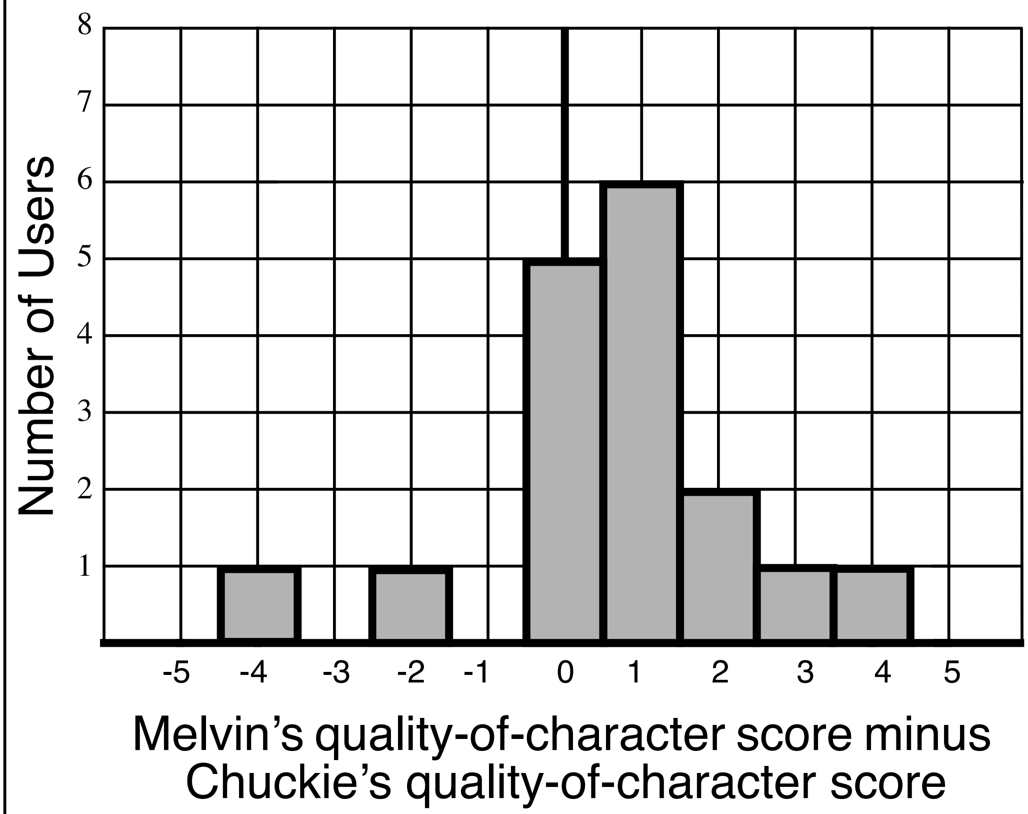
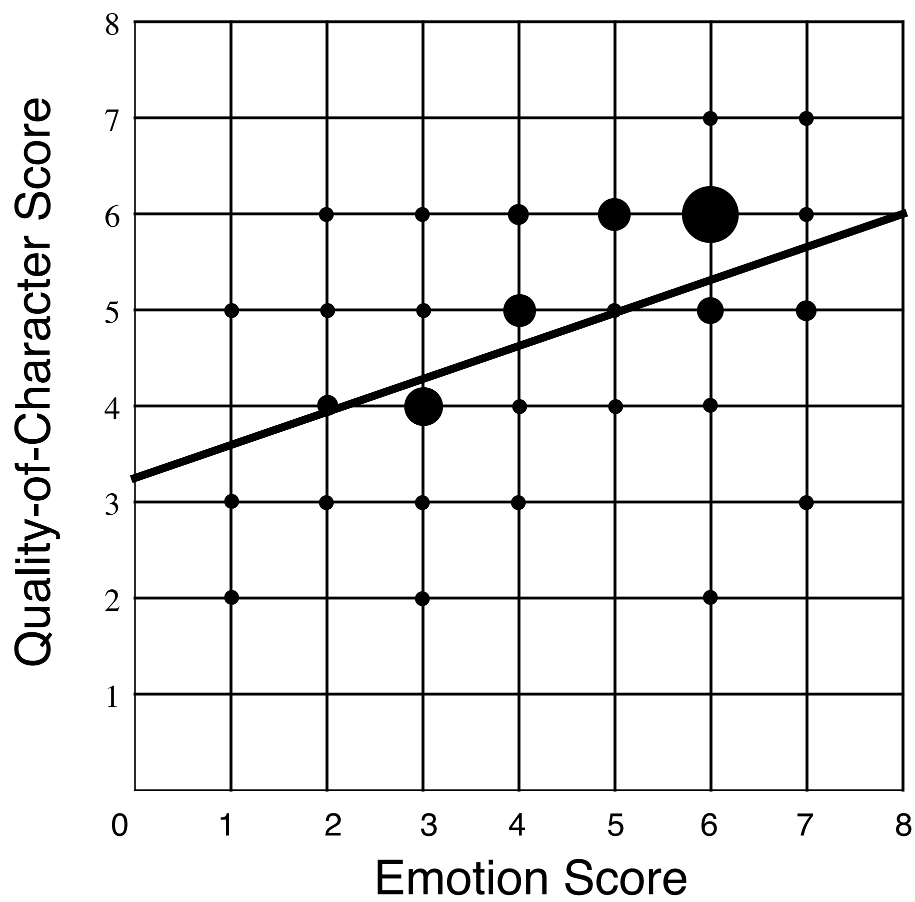


FIGURE 6-3 **Emotion vs. Quality of Character.** This graph shows the relationship between how users scored the emotional expressiveness of the characters (1=unemotional; 7=emotional) and how users scored the overall quality of the character (1=awful character; 7=great character). I have plotted each user's score as a point, with larger dots representing multiple users giving the same values. The line represents a linear regression of the data. The formula representing the line is F(x)=0.33x+3.37. The slope of the line is positive, showing that the more emotional the characters seemed, the better they were as characters. The correlation coefficient is r=0.46 on a scale of 0 to 1.



The next question is whether adding emotions makes the personality of the character less clear to the user. The mean score for how clearly Melvin's personality was conveyed was 5.59 as compared to Chuckie's 4.71. 9 of 17 users thought that Melvin had a more clearly conveyed personality, with 4 each indicating that Chuckie's personality was as clear or clearer than Melvin's.

To test the statistical significance of these results, I applied a t-test to the distribution derived by subtracting Chuckie's clarity-of-personality score from Melvin's, which has a a mean of 0.88 and a standard deviation of 1.87. I am able to claim that the actual mean is greater than 0 with confidence >95%. That is, with high confidence, I can claim that users really do find Melvin to have a more clearly conveyed personality than Chuckie. Figure 6-4 shows the various values that the differences in scores takes on for the users.

To get an idea of the level of influence emotion has on the clarity of a character's personality in the study, I mapped clarity-of-personality scores for all three characters against their emotion scores as seen in Figure 6-5. I also performed a linear regression and found that the line best fitting this data has the formula: $F(x)=0.24x+4.32$. The positive slope of 0.24 again indicates the positive correlation between emotion and quality of character. The coefficient of correlation in this case is 0.33 on a scale of 0 to 1, indicating that the linear regression does a fairly good job of fitting the data[1].

The final test is whether adding emotions makes it harder for users to suspend their disbelief. The mean score for how often Melvin broke the users suspension of disbelief was 3.12 as compared to Chuckie's 2.76. 7 of 17 users thought that Melvin was more likely than Chuckie to break the user's suspension of disbelief, with 5 each indicating that Chuckie was just as likely or more likely than Melvin to disrupt the user's suspension of disbelief.

These numbers aren't as promising as the others, since they seem to indicate that Melvin is more likely to disrupt the user's disbelief than Chuckie. These numbers, however, are much less statistically significant than the others.

1. To get an idea for what this value means, a study relating high school and college success of computer science students at a large midwestern university found that the relationship between high school science grades and overall high school grades scores for these students were correlated with coefficient 0.33 [Moore89].

FIGURE 6-4 **Melvin vs. Chuckie: How clear a personality?** This histogram shows the differences in how Melvin and Chuckie were scored when users were asked, “Did X have a clearly conveyed personality?” The scale ranged from 1 (not at all) to 7 (very much). The total number of users was 17. For example, this chart shows that 2 of the 17 users scored Melvin 1 point higher than Chuckie on the 1-7 scale.

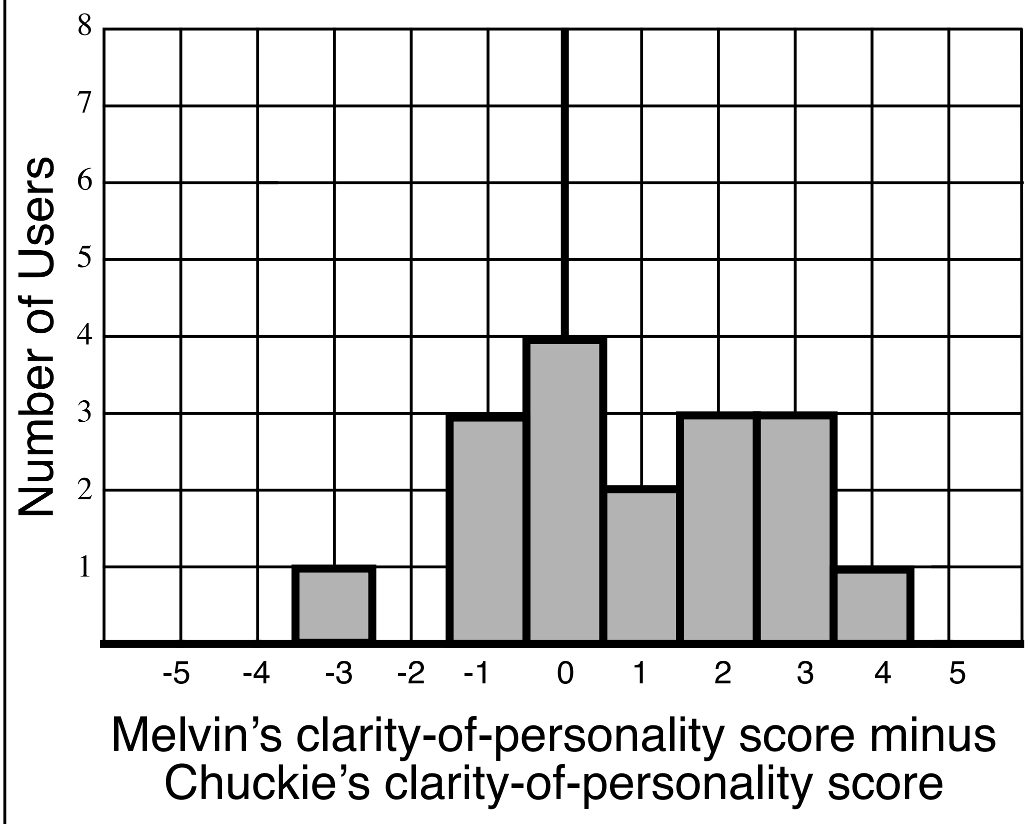


FIGURE 6-5 **Emotion vs. Clarity of Personality.** This graph shows the relationship between how users scored the emotional expression of the characters (1=unemotional; 7=emotional) and how they answered the question "Do you think X had a clearly conveyed personality?" (1=not at all; 7=very much). I have plotted each user's score as a point with larger dots representing multiple users giving the same values. The line represents a linear regression of the data. The formula representing the line is F(x)=0.24x+4.32. The slope of the line is positive, indicating that the more emotional the characters seemed, the more clearly their personality was conveyed. The correlation coefficient is r=0.33 on a 0 to 1 scale.

To test the statistical significance of these results, I applied a t-test to the data derived by subtracting Chuckie's disruption of disbelief score from Melvin's. This distribution has a mean of 0.35 and a standard deviation of 2.42. This means I am only able to claim that the actual mean is greater than 0 with confidence <75%. That is, I cannot claim with any statistical significance that Chuckie or Melvin is more likely to disrupt the user's suspension of disbelief. Figure 6-6 shows the various values that these differences in values take on for the users.

Comparing the level of influence emotion has on the suspension of disbelief, I once again find that the results are inconclusive. I mapped the suspension of disbelief scores for all three characters against their emotion scores as seen in Figure 6-7. I also performed a linear regression and found that the line best fitting this data has the formula: $F(x)=0.03x+2.61$. The positive slope of 0.03 indicates a very slight positive correlation between emotion and disrupting the suspension of disbelief. The coefficient of correlation in this case, however, is 0.03 on a scale of 0 to 1, indicating that the linear regression does not fit the data at all well and that there is no strong claim to be made about this relationship.

In conclusion, Em is moderately successful. Even though this was the reasonably difficult task of showing emotion in an only moderately emotional character, I was still able to use Em to create the appearance of emotion while increasing the character's overall quality and clarity of personality. I also showed that the presence of emotion did not significantly affect the users' ability to suspend their disbelief.

FIGURE 6-6 **Melvin vs. Chuckie: Breaking the suspension of disbelief?** This histogram shows the differences in how Melvin and Chuckie were scored when users were asked, “Did X ever do anything to disrupt your suspension of disbelief?” The scale ranged from 1 (never) to 7 (all the time). The total number of users was 17. For example, this chart shows that 2 of the 17 users scored Melvin 2 points lower than Chuckie on the 1-7 scale.

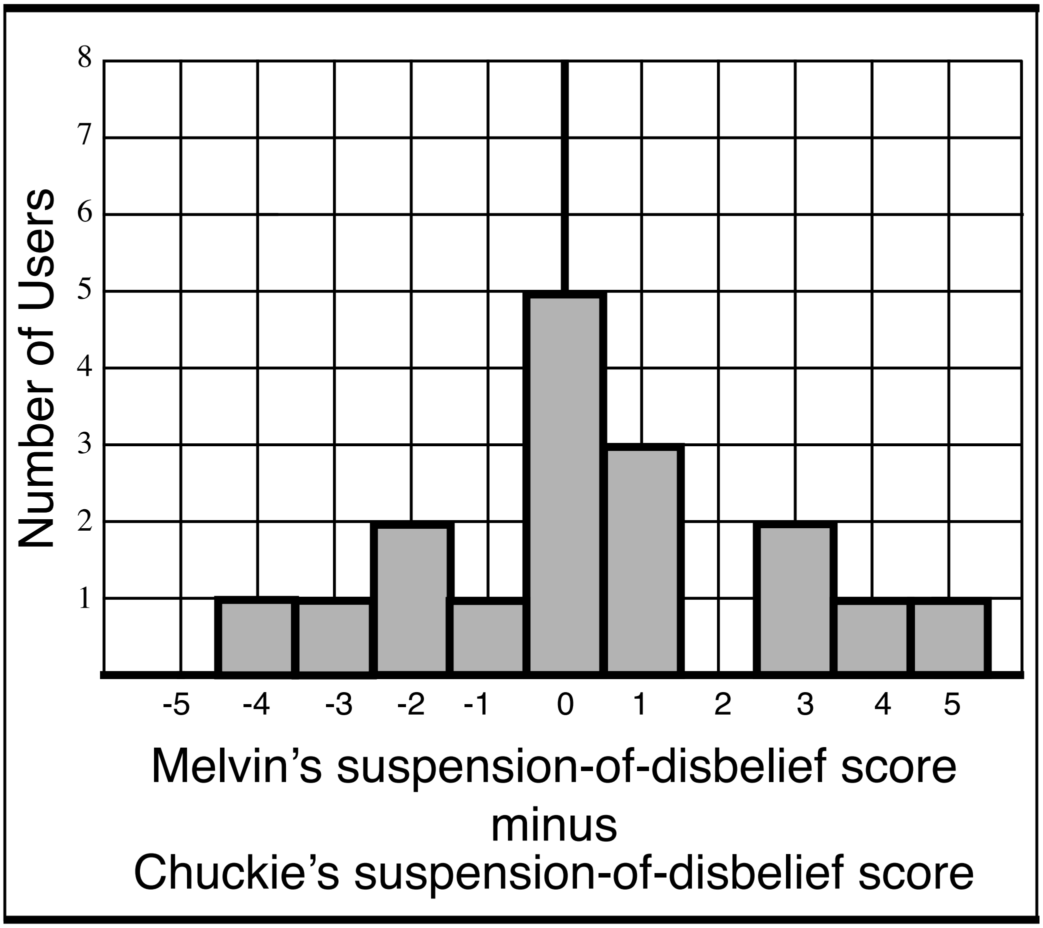
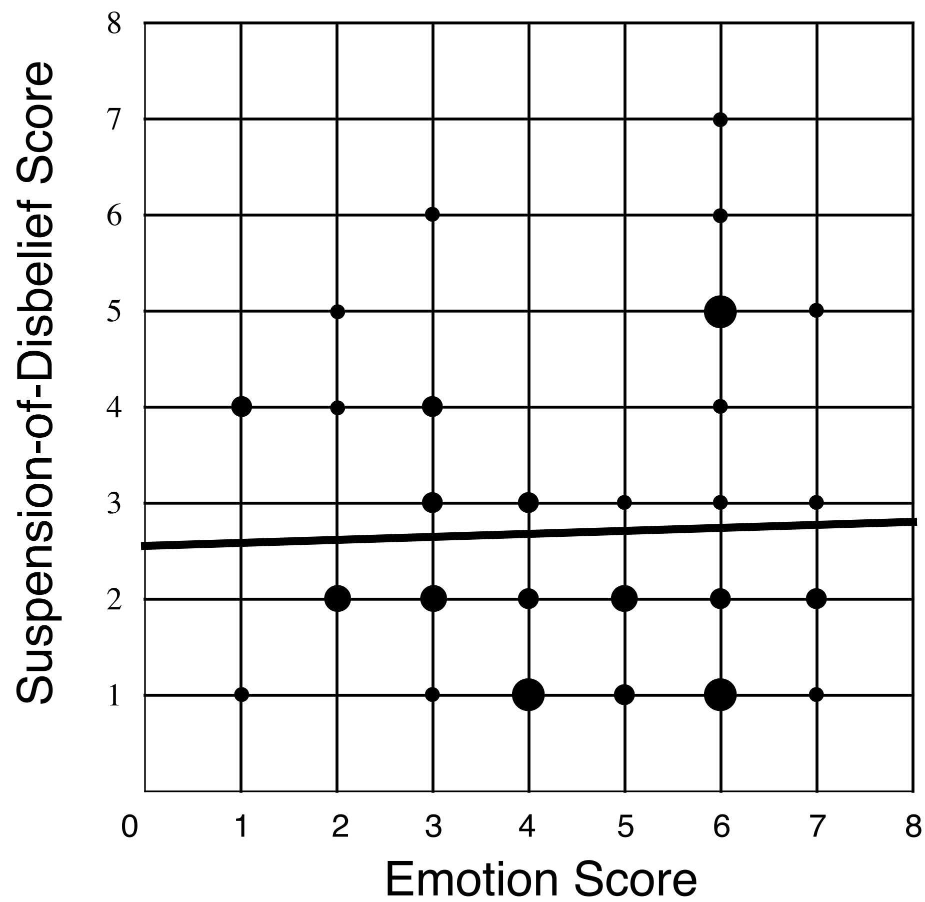


FIGURE 6-7 **Emotion vs. Ability to Suspend Disbelief.** This graph shows the relationship between how users scored the emotional expression of the characters (1=unemotional; 7=emotional) and how they answered the question “Did X ever do anything to disrupt your ‘suspension of disbelief’?” (1=never; 7=all the time). I have plotted each user’s score as a point, with larger dots representing multiple users giving the same values. The line represents a linear regression of the data. The formula representing the line is F(x)=0.03x+2.61. The slope of the line is positive, showing that the more emotional the characters seemed, the more likely they were to cause disruptions in the user’s suspension of disbelief. However, the correlation coefficient is r=0.03 on a scale of 0 to 1, indicating that the data is almost uncorrelated.



6.3 Testing the Internals of Em

Although my main concern was to show that Em could be used effectively, I also wanted to find a way to test the individual parts of the Em system. Starting with the assumption that Em can be made better, my goal was to identify those parts of the architecture that needed the most improvement and some ways to improve them. I decided to examine the gunman because that world and character were built for the explicit purpose of pushing the emotion technology by providing emotional situations for the characters to react to.

Methodology

I started off by providing each of three subjects with the Em architecture diagram in Figure 6-8. I explained, in simplified form, what happens inside the agents using this diagram: inputs are processed, a set of emotion structures are created that have a number of fields, these structures are mapped into behavioral features that have a similar structure, and these features are used to affect the agent's behavior.

I then showed each subject three traces from *Robbery World* with emotion processing output included. This output includes what emotion structures and behavioral features are present at any given time. (One of these traces is included in Appendix B.) The subjects were then asked to go through each trace a line at a time and mention anything they thought about the processing. In particular, they were asked to consider: are there any emotion structures present that shouldn't be? are there any not present that should be? are there any present that are too intense or not intense enough? do the features make sense given the emotional state? are there features present that shouldn't be? are there features not present that should be? are any of the features too intense or not intense enough? and do the gunman's action make sense given the behavioral features?

I kept written notes of the comments made by the subjects, although I did not keep complete records of everything said, which could be a source of bias in the data. Nonetheless, since the main point of this study is to suggest future work and not to prove any specific claims, I feel this potential bias is acceptable.

The subjects were chosen on the basis of their backgrounds. I had a subject with a computer science background (subject #1), one with a writing background (subject #2), and one with a psychology background (subject #3). I chose these three areas to try to get a range of expertise and because subjects in these areas seemed appropriate for evaluating this system.

Results

Most of the changes that the subjects suggested are simple to make. In some cases, the choices I made were artistic ones that could have been made otherwise, but that would have made for a different character. For instance, I made the gunman aggressive; subjects #1 and #2 said that they would expect more fear than aggression.

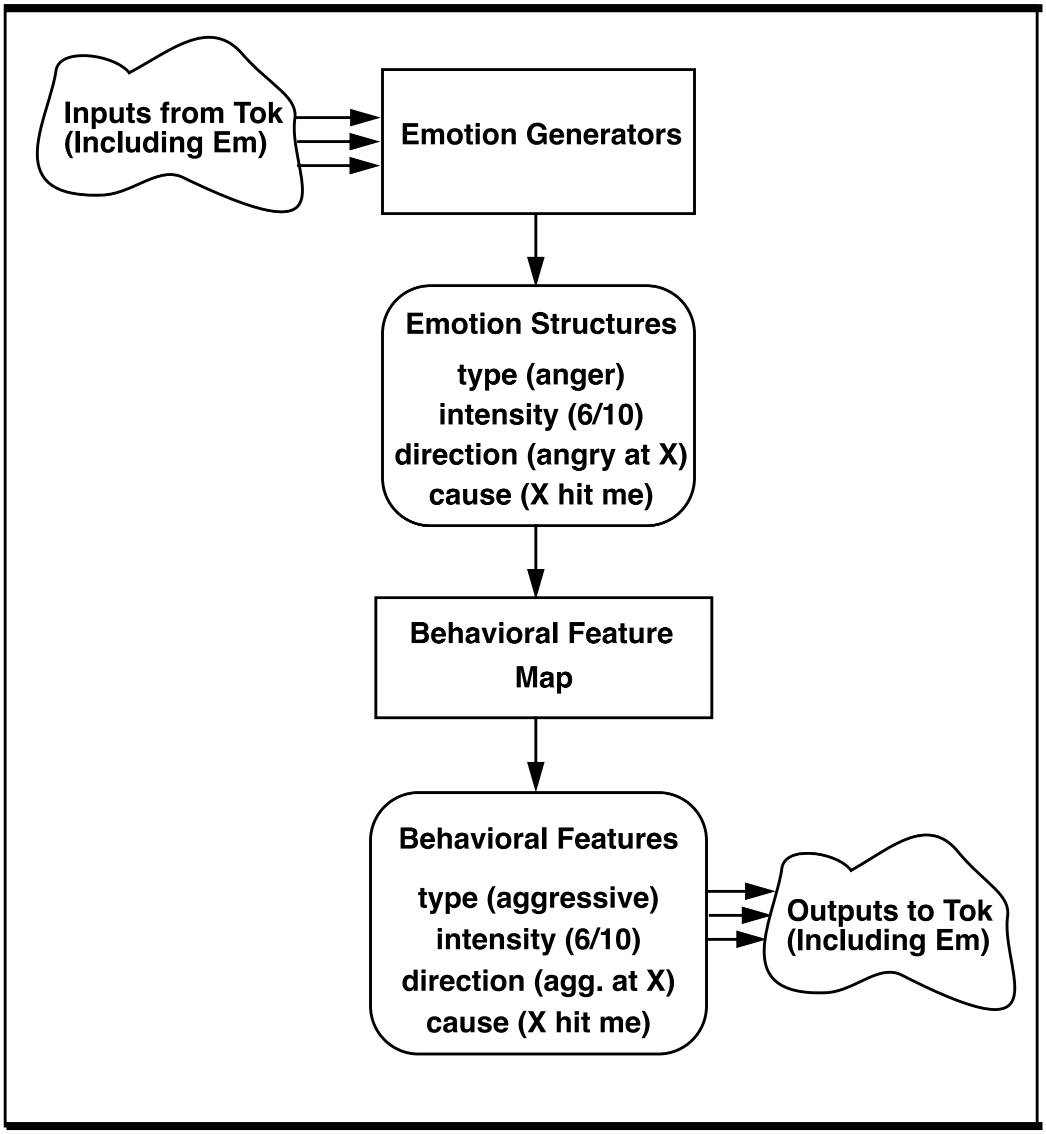
The subjects also provided a number of more interesting insights, some of which may provide ideas for future research. It is those insights that I list here (in no particular order).

- **Goal-specific emotion generators.** Subject #1 suggested that when emotions are generated because the `execute-holdup` goal becomes more likely to fail, those emotions should include high levels of anger and distress but low levels of fear. This can be accomplished by writing emotion generators that look at the goal in question when determining how to react to states such as likely goal failure. This is more flexibility than is provided by the default emotion generation system, but it can be accomplished within the Em architecture. Emotion generators are fairly simple bits of code (see section 3.3); the only difficulty would be the coding time required if an agent required a separate generator (or set of generators) for each of a large number of goals.

- **Goal-specific decay.** Subjects #1 and #2 both wanted to be able to have faster decay of emotions related to some goals than others. For instance, subject #1 suggested a fast decay of the anger associated with being insulted, but gradual decay of the anger associated with being shot at. Since each emotion structure has its own decay function, this is a simple matter of modifying the code that decrements the emotional intensity associated with a particular goal.

- **Computing Joy and Distress Intensities.** The intensity of joy and distress structures is based on the change in how likely the goal is to succeed or fail. For instance, if a goal of importance 8 goes from being 50% likely to succeeding to actually having succeeded, the joy will be of intensity 4. Anger, fear, gratitude, and hope, however, are based on the absolute likelihood of success and likelihood of failure values, meaning that they will generally be more intense emotions than joy and distress. I chose this approach because I generally wanted fear and anger to be more intense than distress when they were present. The subjects, however, found the discrepancies in intensities odd. Possible variations on how the intensity for joy and distress are computed is an area for future exploration. Both subjects #1 and #2 raised this issue in a number of places. Subject #3 did not mention the different intensity levels being unusual.

FIGURE 6-8 Figure Shown to Subjects in Em Analysis



- **Context-Dependent Decay Functions.** Subject #1 suggested that after being shot, the gunman's emotions about things other than being shot should decay quickly. This means that the decay function for emotion structures should take into account the other high intensity emotion structures that are present. The idea is that high-intensity emotion structures should become most important while the other structures fade away.

  I generally achieve a similar effect either in the emotion-to-feature map, by mapping low-intensity emotion structures to zero-intensity features when stronger emotion structures exist, or in the way that I decide which features should affect behavior, by using the highest intensity features to make most action decisions and only relying on lower intensity features that don't conflict with the choices made by the high-intensity features. Having these effects be handled by the decay mechanisms is another possible approach that could be explored. Em's decay mechanism is able to achieve these effects; the question would be whether this approach has specific advantages or disadvantages over other approaches.

  The choice of mechanism will depend, in part, on whether the lower intensity emotions are still important in terms of overall emotional intensity; if the artist wants these emotions to increase overall intensity, then the decay mechanism shouldn't be used.

- **Extending the Distress and Joy Structures.** All three subjects mentioned that the decay of emotions will often need to be based on the environment. For instance, the gunman's distress and anger at the officer for disrupting the hold-up shouldn't decay while the officer is still around. By default, Em delays the decay of fear and hope emotion structures until the cause of the emotion structure is removed. Em does not delay the decay of anger, distress, gratitude, and joy, partly because the "cause" of the emotion is less well defined. For instance if an agent is angry because he was hit, the cause of the anger is being hit, but the emotion intensity should probably be based on the presence of the offending agent. One approach to this problem is to use the direction information in the emotion structures to determine if the target of the emotion is still present; while the target is present, don't decay. This approach, however, would not work for joy and distress since they are undirected. A possible solution is that in the cases where there is an agent responsible for the joy or success—this information is already determined and used to generate gratitude and anger—this information could be stored with the joy or distress structure as the "direction" of the emotion. This direction information could then be used by the decay functions to delay until the target of the emotion is no longer present. The implementation and evaluation of such a technique is left for future work.

- **Goal-Dependent Feature-Mapping Rules.** Subject #2 pointed out that feature-mapping rules could be made goal dependent. For example, being afraid that one goal might fail could lead to defensive actions while similar fear of another goal failing might lead to aggressive actions. A related possibility is to keep the feature-mapping rules general but take importance information into account, so that fear of more important goals failing might cause different features than fear of less important goals failing. It may be, however, that importance is not enough information and the feature-map has to be fit to each individual goal. This is not difficult, just potentially time-consuming if there are many goals to be handled.

- **Goal-Based Exclusion in Feature-Mapping Rules.** Subject #2 sometimes wanted the features associated with goals to follow a winner-takes-all rule. In other words, the gunman should act aggressively or defensively about a particular goal, but not both. I used both and made different decisions based on whether one or both was high, so I found the mapping I had worked well. For instance, when aggressively pursuing the goal not to be killed, the gunman may try to shoot the police officer; when acting defensively, the gunman may give up or run away; when both features are present, the gunman will often take the cashier hostage. An artist that wanted to take a different approach to mapping emotion structures to behavioral features, such as a winner-takes-all approach, could modify the current feature-mapping rules to generate only a single feature based on any goal.

One of the themes in many of these suggestions is the need for various aspects of emotions (e.g., decay, the emotion-to-feature map) to be context-dependent (e.g., what goal is the cause of the emotion, what other emotions are present). This is not surprising and lends support to the broad approach that I took. While Em does not provide general solutions to these problems, the fact that Em was built with breadth in mind means that artists are able to accomplish the effects that the subjects suggest for particular characters. It may be that more general mechanisms can be created to provide even more support for such tasks; this is left for future work.

6.4 Summary

Here are some of the important issues that have come up in this chapter.

- I showed that the Em emotion architecture can be a useful tool for creating believable emotional agents. I had users compare the Melvin character in *The Playground* to an Em-less version of the same character, called Chuckie. I found that Melvin was more emotional than Chuckie. I also found that these emotions did not come at the expense of the overall quality of character, the clarity of the character's personality, or the users' ability to suspend his or her disbelief. In fact, I showed a positive correlation between emotions and both overall character quality and clarity of personality.

- I asked three subjects to review the processing of the Em system in the gunman character in *Robbery World*. I found that most of the things that the subjects thought should be done differently were in the details of the emotion system. In other words, the Em architecture is flexible enough to support their suggested approaches, though the default system I built happens to handle them differently.

Part II:

Believable Social Agents

CHAPTER 7

Believable Social Agents

7.1 Introduction and Overview of the Problem

An important subgoal along the path to interactive drama is the ability to create believable characters that can interact with each other and with the user. Traditional artists have been creating social characters for millennia—the goal of this work is to create social characters that can interact with a character that the artist can't control.

Until now, many computer and video games have provided "social" characters where the extent of the social interaction is fighting of some sort. Others have provided "social" characters that are not very interactive. For example, the games use video clips of live actors so the characters can interact with each other but not with the user or they provide characters that can have very controlled (e.g., menu-based, point-and-click) interactions with the user.

The goal of this work is to enable artists to create believable interactive characters that can have much richer social interactions than has previously been possible.

The approach I have taken is to provide a methodology for building believable social behaviors for particular characters. The methodology supplies a set of heuristics for artists to help them create personality-rich social behaviors. The methodology also proposes a minimalist approach to modeling the other characters in the environment. I have used this methodology to create effective social behaviors with a surprisingly small amount of representation.

In this chapter, I introduce this methodology for creating social behaviors and discuss some related work. Chapters 8 and 9 are devoted to expanding on and exploring the methodology through a number of case studies. In Chapter 10, I describe a user study that provides evidence that this methodology can be used to create believable social behaviors.

7.2 The Goals

I broke the goal of supporting artists in the creation of social behaviors for believable agents into three subgoals. The first goal was to design a methodology for building believable social behaviors. The second goal was to use the methodology to create specific instances of some interesting social behaviors, like negotiation and making friends. The third goal was to evaluate the behaviors that had been created with the methodology.

7.2.1 A Methodology for Building Social Behaviors

The first goal was to develop a methodology for building specific social behaviors for specific characters in specific worlds. An artist creating a specific world should be able to use this methodology to help create the necessary social behaviors for the characters in that world. For example, when I built *The Playground*, I needed to build two characters that had very distinct personalities that were able to engage in a set of social behaviors, such as negotiation. I used the methodology that I will describe to build those social behaviors. Each character has its own version of the behavior tailored to fit its specific personality.

Another possible approach that has often been suggested to me is to create "universal" social behaviors that account for variations of personality within the behavior. For instance, I could create a universal negotiation behavior that had a set of knobs on it that could be set to fit the personality of the character in questions. I rejected this approach for artistic reasons. Like the problem of creating believable emotions, *creating believable social behaviors is inherently an artistic problem with artistic constraints*.

The problem with the universal-behavior approach is that I do not believe that it is possible to create a behavior that expresses personality in a set of "knobs" without stifling the artistic creation of quality characters. For instance, imagine Hamlet, Bugs Bunny, and Captain Kirk negotiating. I believe that any knob-based social behavior is going to have considerable difficulty expressing such a diverse set of personalities. After building a number of characters and seeing how much personality permeates the character's behavior, I believe this even more than when I started.

It might be possible to create a universal behavior that could be a useful starting point for artists. The universal behavior would exhibit a basic social behavior that is roughly what the artist wants and the artist would modify the behavior to really fit the personality of the characters. This might ease the creation of behaviors without a loss of artistic freedom. The reason that I did not attempt this approach is that the set of social behaviors is undefined. In other words, I would have to create some finite set of universal social behaviors for artists to use and they would have to confine their characters to that set of behaviors. Because social behaviors are so varied (for instance, the gunman and cashier in *Robbery World* need to know how to hold up a store and be held up respectively) this would again stifle the artists. By providing a methodology, I provide a tool that helps artists create whatever social behaviors they need.

The methodology I provide can be looked at as a two-part set of heuristics. The first part of the methodology is a set of heuristics for incorporating the personality of the character into the behavior. Users find that characters built using these heuristics have distinctive personalities and are believable. The second part of the methodology provides a heuristic for handling the problem of modeling other agents. The methodology suggests a minimalist approach to the problem of modeling other agents. I have found this approach can lead to surprisingly small representational schemes that (according to users) produce believable, robust behaviors.

The methodology will be introduced in section 7.3, but many of the details of the methodology and how to use it are presented through case studies in Chapters 8 and 9.

7.2.2 Examples of the Methodology and Social Behaviors

The second goal was to use the methodology I had developed to build several social behaviors. There were three reasons for doing this. First, I provide artists with examples of how to use the proposed methodology. Second, implementing a varied set of behaviors demonstrates that my methodology can be applied fairly broadly. Third, I was able to ask users to evaluate some of these behaviors to provide evidence that the methodology can be used for building believable social behaviors.

In Chapters 8 and 9, I will describe two behaviors in reasonable depth: negotiation and making friends. These behaviors allow me to describe many of the details of the methodology in more detail and to provide examples of how I use the methodology. These two behaviors also provide the basis for the user evaluations that I will describe in the section 7.2.3.

In addition to these two behaviors, however, I built a number of other behaviors using my methodology. Describing them each in detail does not provide anything that I cannot provide in the two deeper examples. To give a feel for the breadth of the methodology, however, I will briefly list them here.

- **Holding up a convenience store.** This behavior gives some evidence of the variety of behaviors to which the methodology applies. This behavior is not unusual for an agent designed for a story system but is not typically the kind of behavior studied by AI researchers building behavior architectures.

  In *Robbery World*, the gunman knows how to hold up a store and the cashier knows how to be held up. Traces of this system are provided in Appendix A.

- **Giving orders.** I have built a number of characters that can give orders. In *Robbery World*, the gunman can give simple orders to the cashier ("Gimme all your money.") and the officer ("Back off or I'll shoot the hostage."). In *Office Politics*, Mary and Sarah can order Gus to perform certain tasks, like fixing the copier or e-mail server. In *The Playground*, Sluggo can order the player to do various things, like get out of the tree house or give Sluggo a baseball card.

- **Following orders.** If agents are able to give orders, other agents need to be able to follow orders. In *Robbery World*, the cashier can follow the gunman's commands. Also, the gunman recognizes many of the commands of the officer, but typically chooses to ignore them. If he gives up, though, he needs to follow the officer's orders more closely. In *Office Politics*, Gus needs to know when to follow orders and when not to. He'll always obey his boss, Mary, he'll never obey Sarah, who he doesn't like, and he'll obey the player when the player has a deadline to meet.

- **Asking for help.** This behavior, like negotiation, has received some attention from the AI community [Cesta93]. The AI approach to this problem is typically concerned with creating agents that ask for help intelligently. My approach is to incorporate personality and make the behavior believable even if it isn't especially competent. In *Office Politics*, Sarah can ask other characters for help with her task of fixing the copier. She can also call in favors if she has previously agreed to help the player fix the e-mail server.

- **Insulting.** Both Sluggo and the gunman are able to insult other characters. this behavior is interesting in that it can either be a behavior unto itself or it can be incorporated into other behaviors. For instance, Sluggo can simply insult the player ("You're a dork.") or he can insult the player as part of another behavior, like negotiation ("What'll you give me for Mays, dork?").

- **Threatening.** Sluggo and the gunman are able to threaten other characters. For instance, Sluggo will threaten to beat up a character who is annoying him. The gunman will threaten to shoot the cashier if the officer doesn't let him escape.

- **Deceit.** In *Office Politics*, Sarah engages in a simple form of deceit. If the player asks her for help fixing the e-mail server, she will agree to help but will not ever fix the server. She need to be able to make the offer as well as remember that she has made the offer in order to handle future interactions with the player appropriately.

7.2.3 Evaluating the Methodology

The final goal for this work was to provide evidence that the methodology I had developed could be used to create social behaviors that users felt were believable. Chapter 10 will describe the user study that I ran to evaluate the social behaviors of the characters in *The Playground* simulation. In particular, I evaluate the negotiation behaviors of Melvin and Sluggo as well as Melvin's behavior of making friends with the user.

7.3 Methodology for Building Social Behaviors

The goal of the methodology is to provide artists with a set of heuristics that will help them create *specific* instances of social behaviors for *specific* characters in *specific* environments. For instance, in Chapter 8, I will demonstrate how I used the methodology to build negotiation behaviors for two specific characters in a particular world. I don't claim that this methodology is the only way to build believable social behaviors. I will try to show over the next three chapters, however, that it is an effective approach that has enabled me to create behaviors that users have found to be believable.

In this section, I will briefly introduce the main ideas in the methodology. Then, over the next two chapters, I will use a number of case studies to flesh out the methodology in much more depth and show how I used it to create a number of rather different kinds of social behaviors. The methodology has two parts. The first part is concerned with how to incorporate personality into social behaviors; the second part is concerned with how to model other characters.

7.3.1 Creating Personality-Rich Social Behaviors

In order to build believable social behaviors, the artist will need to do more than just build "working" behaviors. Very often, *the point of the behavior is not that it achieves a specific goal, but how the goal is pursued*. This is very different from the kind of thinking that goes on in traditional AI. AI has traditionally been aimed at building programs that solve problems, such as playing chess, diagnosing diseases, or scheduling machines in a factory. A believable social character may be able to do these things, but may not be able to do them well. For instance, a bumbling doctor that mixes up patients' charts and often misdiagnoses their ailments may be the artist's desired character.

One reason I built the playground simulation was to help make this point. The two characters, Melvin and Sluggo, are kids with distinct personalities. Melvin is a nerd who wants to make friends. Sluggo is a bully who is as happy to fight as to talk and who isn't particularly intelligent. One of the user's main activities on the playground is to negotiate with Melvin and Sluggo to try to get a Willie Mays baseball card. Both Melvin and Sluggo will negotiate and I have gone to great lengths to make sure they stay in character while they do so. Whether or not Melvin becomes your friend or Sluggo gets better cards isn't really important as long as they act believably. For some sample traces of Melvin and Sluggo negotiating, see Appendix A.

Even Sarah in *Office Politics*, who comes across as having a fairly bland personality, was specifically designed that way. Sarah's role in *Office Politics* is to trick the player so that the player fails to get the e-mail server fixed in time. As the villain in the story, I wanted her to seem fairly innocuous until the time that the player realized the darker side to her. In this world, I designed the character with the larger story experience in mind and chose to create a bland personality.

Characters, unlike most AI programs, have personalities. Artists need to take the character's personality into account when building social behaviors. Knowing how to do this, however, is not easy. Even in traditional, non-interactive artistic media, artists have no definite, provable method for accomplishing this. There is no reason to expect that it will be any easier, or less artistic, in this new medium.

The methodology I propose is to incorporate a specific set of aspects of personality into each behavior. This makes the problem of adding personality to social behaviors more concrete. I recommend the following personality issues be considered for each behavior: personality quirks, competence, emotions, relationships, attitudes, norms, roles, other goals of the agent, and robustness. I will briefly describe each of the items in the list. How each element of the list is incorporated into social behaviors will differ from behavior to behavior. I will discuss techniques for integrating these various aspects of personality into social behaviors in the next two chapters. I will also use the case studies to provide additional motivation for why I adopted this particular approach.

This list may not be complete, but, according to users, the behaviors created with these elements of personality incorporated have proven to be effective.

Personality quirks. Artists (e.g., [Jones89]) often talk about the importance of quirks for bringing a character to life. The fact that Bugs Bunny says, “What’s up doc?” in his distinctive Brooklyn accent is part of his personality and his success as a character. In *The Playground*, Melvin is a trekkie and if he is going to stay “in character” this element of his personality needs to come through even when he is engaging in social behaviors, like negotiation.

Competence. In traditional AI, this is the focus of most work—making the agent more competent, more intelligent, better able to perform tasks. In believable agents research, competence is just one issue to be considered. The goal is no longer strictly *greater* competence but creating the *right level* of competence for each agent. For instance, Melvin and Sluggo should have different levels of competence when negotiating, with Sluggo being less competent since that fits his personality.

Emotions. Agents have emotions based on all sorts of things and these emotions should be displayed through social behaviors. If a character is sad for reasons that don’t have anything to do with the social behavior in question, that behavior still needs to reflect that emotional state. In addition, agents must have appropriate emotions based on the outcome of their social interactions. For example, Sluggo should get angry if a trade doesn’t go his way.

Relationships and Attitudes. Agents that engage in social behaviors will often need to modify those behaviors based on their attitudes and on their relationships with the other characters with whom they are interacting. If a character is interacting socially with their friend (friendship is a relationship), they will often want to act differently than if the other character were their enemy. Similarly, interactions with trusted agents (trust is an attitude) will often need to be different from those with untrusted agents. Although relationships and attitudes are different, I will typically discuss them together as I have found that they are often similar in the ways they relate to social behaviors.

Social constraints on behavior (or social norms). Social behaviors need to incorporate what is and is not socially acceptable behavior. This is not to say that agents cannot act unacceptably, but that the agent builder should make such decisions based on the agent and behavior in question. Social norms are especially important in social interactions because they account for social phenomena like politeness.

Social roles. The role that an agent plays in society can affect any number of behaviors, but mainly affects those behaviors particular to that role. Roles are often occupations, like police officer or waitress. Dyer [Dyer83] shows that thinking of characters in terms of the roles they are playing is an effective way to understand stories. This implies that social role can be important to consider when generating stories as well.

Other Goals. An agent will often have other goals active simultaneously that will affect how a social behavior unfolds. For instance, if an agent is trying simultaneously to negotiate with an agent and to become that agent's friend, the negotiation will likely be modified based on the friendship goal.

Robustness. Robustness is important for a large class of behaviors, including non-social behaviors. Robustness for believable agents means never doing anything to break the user's suspension of disbelief. This means that even when social interactions don't go smoothly, the agent needs to be able to recover. This notion of robustness is different from that used in other areas of AI, such as robotics. In robotics, robustness is typically a measure of *competence* in a variety of situations as opposed to a measure of *believability* in a variety of situations.

7.3.2 How to Model Other Characters

One of the difficulties in creating social behaviors is the problem of representing other characters. For instance, how much does one character need to know about the goal, beliefs, relationships, internal processing, capabilities, etc. of other agents in order to interact with them?

The question of representation (typically related to modeling the physical world) is one that has perplexed the AI community for years [Hayes94]. Deliberative approaches (e.g., [Newell76, Fikes72]) have tended to rely on representation as an important part of being able to solve difficult problems. Behavior-based approaches (e.g. [Brooks91]) have pointed out that relying on good models of the world can be dangerous given how hard it is to create and maintain such models in complex worlds.

I believe that there is an important and useful middle ground between the two camps.

I sympathize with the behavior-based warnings of the difficulty of doing representation well (e.g., [Brooks91]); on the other hand, I don't see how to solve some complex social problems without some representation. Robots may be able to navigate without representations because they can sense the objects around them; many aspects of the social world, though cannot be sensed, like whether Tom likes me or whether Sue and Jeff are friends. In some cases there will be physical cues of such social phenomena (like facial expressions), but recognizing them may be hard and they will probably not be continuously presented (which means remembering them, which requires some sort of representation).

Some researchers have created social agents without using any representations, such as Mataric [Mataric92], Wavish [Wavish92], and Hickman and Shiels [Hickman91]. In all of these cases, the behaviors are (socially) much simpler than artists will often want and none of them incorporate believability. For instance, Mataric's robots can flock and gather food with other agents, but they are really just co-existing instead of directly socially interacting. They also have no sense of relationships to each other or any kind of personality. These issues were not important in her work, so she was able to avoid representation. I have found in my work that I was unable to avoid it entirely, but could still restrict its use.

The big question becomes, how can artists allow for representation without falling into the trap of being unable to control the representations they create? I suggest the following approach:

1. Start with no representation.
2. Begin building the character and the character's social behaviors.
3. If necessary, add the simplest representations that seem sufficient for the world, character, and behavior in question.

By following this technique, the behaviors tend to end up with small, manageable representations that are enough to enable the task at hand without committing to representations that are unnecessarily complex. *It turns out that often surprisingly small representations can produce complex, robust, social behaviors*. When more complex representations are called for, they are used—they are just not forced on the artist by default.

As I mentioned above, I chose not to use a general, default system for modeling other agents in the world because I wasn't sure such a system would be both fast enough and robust enough to handle complex, dynamic environments. An additional danger of such an approach is that if the default system is not extremely flexible, it could stifle the artist's freedom. For instance, a common assumption in AI work is that agents are rational; this assumption is unacceptable for believable agents. Similarly, the automated approach chosen would have to support many sorts of idiosyncratic reasoning to suit various characters. No AI system currently supports this level of flexibility and I believe that it would be difficult to create such a system. I will argue, however, that powerful, general systems are not necessary because many believable social behaviors can be built with very simple forms of representation and reasoning.

This and many of the other claims and arguments in this section will be expanded on in the case studies provided in Chapters 8 and 9. The case studies will also allow me to focus on some techniques for creating social behaviors that rely only on small amounts of representation.

7.4 Related Work

This work has drawn heavily from a number of other AI areas. In this section, I will discuss three main sources of inspiration and ideas: Distributed AI, behavior-based AI, and story-based AI.

7.4.1 Distributed AI

Distributed AI (DAI) is concerned with getting agents to either (1) work together with other agents to solve some common problem, or (2) solve an individual problem in an environment with other agents that are also solving problems [Lesser95]. In both cases the main goal of the agent is to solve a problem or attain a goal. The agents are engineered to be rational and optimal in solving their problems and when dealing with the other agents.

These are hard problems. They are not, however, the problems that are important to solve for creating interactive art and entertainment. Characters will typically not be optimal or rational; how they act is more important than whether or not they achieve any goals. *This emphasis on personality over task-achievement is critical to understanding this work.*

This difference in emphasis, however, doesn't mean that artists have nothing to learn from DAI. When designing a behavior (like negotiation) it is useful to know how a more competent agent would do it. For instance, I used Katia Sycara's PERSUADER [Sycara88] program as a source of inspiration when designing Melvin and Sluggo. Although I wasn't interested in emulating her system, I benefitted from the analysis she had done in order to create her system. For a simple example, the fact that the characters will respond to a rejected offer by trying to talk you into it was inspired directly from a similar feature in PERSUADER. For more complex characters, I expect her analysis of negotiation would have been of even greater use.

DAI might also suggest what kinds of representations artists should use when representations are called for. Although I have not borrowed representations from DAI systems for the agents I have built, there has been a lot of work on knowledge representation for multi-agent domains that some artists might find useful (e.g., [Huber95], [Rao95], [Vidal95], [Mor95]). I expect this will be true especially if the agent needs to be highly competent at some social task.

7.4.2 Behavior-Based AI

My work towards creating believable social behaviors has also been influenced by work in behavior-based AI.

I chose the behavior-based Hap language [Loyall96] as a substrate not only for physical behaviors, but for social behaviors as well. I found the combination of goal-directed behavior and reactivity to be well-suited to social domains.[1]

Brooks's behavior-based methodology of rejecting representation [Brooks91] also played a large role in helping define my approach to representation in social behaviors. I am sympathetic to the warnings about trying to use rich representations in complex and dynamic environments. I have found that, for speed and robustness, avoiding unnecessary representations is a useful approach in the behaviors I have built.

Maja Mataric has looked at building behavior-based social behaviors, such as flocking and group food gathering [Mataric92]. The success of her work with simple robots was encouraging, but she is working on a problem that is very different from mine in a number of ways. She is mostly interested in creating working robots that display simple, recognizable group behaviors like following and flocking. She is not working on the creation of agents with personality and emotion that engage each other in more complex behaviors like negotiation and making friends—the complexity in her work comes largely from the fact that she is working with real robots.

Hickman and Shiels [Hickman91] have also built behavior-based social agents that build television sets in a simulated world. They have taken a reasonably typical DAI task and shown that it can be accomplished by a set of agents without any representation. My goals are different in that I want more direct interaction between the agents (their agents have no communication; they are just working on a joint task) and more believability. To create believable agents, artists need to consider things like the relationships between the agents since this is important for telling stories—it is not important for building home appliances.

I have found that a nihilist approach to representation is not appropriate for creating believable agents. I have also found that a minimalist approach which allows some representation is better suited to the goal of building believable agents that can engage in a variety of social behaviors.

7.4.3 Story-Based AI

Meehan's TALESPIN, Lebowitz's UNIVERSE, and Dyer's BORIS are the three examples of story-based AI that are most relevant to my work.

1. Bryan Loyall is also using Hap to control real-time, believable text generation [Loyall96].

TALESPIN tells simple, fable-like stories about simple characters with different personalities and goals, where the characters interact with each other [Meehan76]. These stories are created by the simulation of a set of characters in a simulated world over time. These stories, however, are not interactive, so the behaviors do not need to be robust enough to endure interactions with real users, just other author-defined agents. The behaviors Meehan creates could, however, be used as starting points for creating interactive, believable behaviors.

TALESPIN is also different from my work in that the personalities of the agents are simpler than what I want to allow artists to create. The TALESPIN agents only have about six different types of goals and seven different attitudes/relationships with other agents. Because the agents also have a finite set of types of interactions they can engage in, Meehan is able to create universal social behaviors of the type I discussed in section 7.2.1. The methodology-based approach that I have taken is rather different in that, by keeping the artist deeply involved in the creative process, it allows for more distinctive personalities and a greater variety of social interactions. The TALESPIN framework could never tell a story about Yosemite Sam holding up a bank.

Lebowitz's work on UNIVERSE [Lebowitz84, Lebowitz85], which generates non-interactive soap-opera plots, is also relevant in some respects. The UNIVERSE characters have 4-dimensional relationships. Those dimensions are the following: positive/negative, intimate/distant, dominant/submissive, and attractedness. The first three are from Schank and Abelson [Schank77], based on the work of Wish [Wish76]. Lebowitz added the last because he found it necessary for his task.

Perhaps the most interesting part of the four-dimensional theory is that Lebowitz found that three isn't enough. As artists try to create new characters with different relationships, I don't want to stifle them by forcing them within some limited model of relationships or attitudes. This is why I allow artists to define the relationships of their characters on a case-by-case basis instead of relying on standardized relationship models. For instance, imagine Ralph Kramden and Norton from *The Honeymooners*. Try to fit their relationship into Lebowitz's four dimensional space. You might be able to do it, but not without losing much of the richness (and quirkiness) of their interactions that make the relationship so interesting.

The work of Meehan and Lebowitz is relevant because they have previously considered such problems as modeling other agents, creating agents that are suitable characters for stories, and creating agents that have relationships and attitudes about other agents. An important feature of the methodology that I am proposing is that it does not have a built-in methods for modeling other characters, so the specific schemes used by Meehan and Lebowitz cannot be directly adopted into my work. Artists, will, however, need to use representations for building specific characters and behaviors and I expect that the approaches of Meehan and Lebowitz to be useful in some of these cases.

Dyer's BORIS [Dyer83] is not a story-generation system, but a story-understanding system. Because of this, his goals are rather different from mine, but many of the aspects of stories and characters that his system needs to understand, my methodology should be able to create. In particular, I did not find any artists that suggested the importance of creating characters that have social roles; BORIS, however, has mechanisms for understanding stories about characters playing particular social roles. This suggested that my methodology needs to be able to create characters that can play particular social roles. I suspect that the fact that I did not find social roles stressed by artists is not because social roles aren't important in stories (cop shows are examples where characters play social roles) but because artists create roles for characters without even thinking about it. Artists can use implicit knowledge but a computer cannot, which is why it took an AI researcher to point out the importance of social roles in believable characters.

7.5 Summary

Here are some of the important issues that have come up in this chapter.

- I introduced the goal of this half of the thesis: enabling artists to create believable social behaviors beyond the limited forms of interaction currently available in computer and video games.

- I broke up the task into three parts: the development of a methodology that artists can follow to help them create believable social behaviors, the implementation of a set of believable social behaviors, and the evaluation of the behaviors built with the methodology.

- I provided an overview of other relevant work in Distributed AI, behavior-based AI, and story-based AI.

- I introduced a two-part methodology for building believable social behaviors. The first part of the methodology is a heuristic check-list to help artists create social behaviors that express the personality of the character. The second part of the methodology is a heuristic for using minimal amounts of representation of other agents. In Chapters 8 and 9, I will expand on the methodology through two case studies. In particular, I will discuss some techniques for integrating aspects of personality into social behaviors and creating behaviors that rely on minimal amounts of representation of other agents. I will also use various behaviors to further motivate my methodology and compare it to other possible approaches. In Chapter 10, I will describe user studies that have shown that behaviors built following this methodology can be believable.

CHAPTER 8

Understanding the Methodology I: Negotiation

The previous chapter introduced a methodology for creating believable social behaviors. In this chapter and the next, I use two case studies to more fully explain the methodology.

Recall that the methodology has two parts: a list of ways to incorporate personality into social behaviors and an approach to creating behaviors that use only minimal amounts of representation of other agents. In this chapter, I will examine both aspects of the methodology. I will describe the elements of personality in more depth, present some specific solutions for adding them to negotiation, and suggest some general lessons about how to use the methodology effectively. I will also provide examples of negotiation behaviors that use very small amounts of representation. In Chapter 10, I will provide evidence from users that the negotiation behaviors described in this chapter are believable.

Although the main focus of the next two chapters is to provide a deeper understanding of the methodology and how to apply it, I will also describe a number of interesting technical issues that arise when implementing believable social behaviors. For instance, in this chapter I describe the problem of creating transitions between behaviors; I had to address this problem so that an agent I was building could politely break off negotiations.

Before I describe how I created believable negotiating agents, I will start with a description of why I chose negotiation and then provide some traces of the agents that I will discuss for the remainder of the chapter.

8.1 Motivation for Building Negotiation

The decisions as to what behaviors to build were not made lightly. Negotiation was chosen, in large part, as a comparison to other AI work on negotiating agents. My hope was that by choosing a behavior that is already studied by the field, the differences in goals will become more obvious. Traditional approaches (e.g., [Sycara88]) aim at competent behaviors. My goal is to create believable behavior, so such issues of personality as incorporating personality quirks and deciding how competent to make the agent are new problems.

Also, negotiation is often considered to be a problem where modeling the other parties of the negotiation is considered important, if not necessary. For example, Sycara's system [Sycara88] uses structured models of the goals and the goal processing of other agents. To create a highly competent negotiation behavior, this modeling is, in fact, useful and important. Such modeling, I claim, is not always necessary or important for believable agents. By choosing a behavior that seems to need a lot of modeling, I hope to demonstrate this claim most clearly.

Finally, negotiation seems to be an extremely useful behavior. In its most simple forms it happens all the time. ("I'll make dinner if you do the dishes.") And even its more complex forms are relatively common. The goal of this chapter is to further explore the methodology for building social behaviors in general; because it focuses on negotiation, however, the examples that I provide may prove to be particularly useful to artists who want to create believable negotiation behaviors.

8.2 Traces of Negotiating Believable Agents

In this chapter, I will describe how I created negotiation behaviors for two specific characters, Melvin and Sluggo from *The Playground*. I will begin with traces of Melvin and Sluggo negotiating with the user. Following my methodology, I tried to give Melvin and Sluggo distinctive personalities that come across during negotiations. I will describe user studies in Chapter 10 that provide evidence that I was reasonably successful. Figure 8-1 provides a brief trace of the user negotiating with Melvin; Figure 8-2 provides a brief trace of the user negotiating with Sluggo. Both traces have been lightly edited for readability (not content). Additional (unedited) traces are available in Appendix A.

FIGURE 8-1 **A Trace of a Negotiation with Melvin**

```
You go into the sand box.
Melvin is now smiling.
Melvin says, ''Salutations! The Klingon high command has
sent   me in search of baseball cards.''.

PLAYER> Melvin: What cards do the Klingons want?

Melvin says, ''Are there any cards we (the Klingon high
command and I) can offer you in return for Babe Ruth?''.

PLAYER> Melvin: Give me everything you have for Ruth.

Melvin says, ''The Klingon Emperor wants to know if you
would be willing to part with Babe Ruth for Reggie Jack-
son?''.

PLAYER> Melvin: No way.

Melvin says, ''I wouldn't be hasty if I were you. Reggie
Jackson for Babe Ruth is a trade any Ferengi would be
proud of.''.
PLAYER> Melvin: How about Ruth for Jackson and Mantle?

Melvin is speaking mechanically to you.
Melvin says, ''We are borg. We will assimilate Babe
Ruth and Ted Williams. You may assimilate Reggie Jackson
and Mickey Mantle. We are borg.''.

PLAYER> Melvin: Oh, OK. It's a deal.
```

FIGURE 8-2 **A Trace of a Negotiation with Sluggo**

```
You go in the tree house
Sluggo says, ``Wanna trade Catfish Hunter for Babe
Ruth?''.

PLAYER> Sluggo: No way. Do you want to trade Mays for
Ruth?

Sluggo says, ``I'll swap you Jose Canseco for Babe
Ruth.  Deal?''.

PLAYER> Sluggo: No deal. I want Mays, get it?

Sluggo says, ``Sorry, bub, but Willie Mays is way bet-
ter than any of your cards.''.

PLAYER> Sluggo: How about Ted Williams and Hank Aaron
for Mays and Hunter?

Sluggo is now red.
Sluggo is now scowling.
Sluggo is now tense.
Sluggo says, ``You think you're cool with your fancy
trades?  No deal, dweeb!''.

PLAYER> Sluggo: Hey, lighten up.  I'll do Ruth for
Canseco, okay?

Sluggo says, ``Get out of here before I pound you, tw-
erp!''.

PLAYER> Sluggo: You don't scare me, you big jerk!

Sluggo says, ``Prepare to be pounded!''.
```

It is important to note that the natural language generation and understanding are both quite shallow in these agents, despite the appearances of the traces. The language understanding system matches on conjuncts and disjuncts of keywords. For instance, in the interaction with Melvin, the sentence, "What cards do the Klingons want?" is turned into a list of two symbols. The first symbol, `general-trade-request`, is created based solely on the existence of the words "want" and "cards." It represents that the speech seems to be a general request to trade cards. The second symbol, `trek`, is created based on the existence of the word "Klingons" and represents the fact that the speaker is simply using Star Trek terminology.

The language generation system is based on templates. For instance, Melvin has a set of templates for presenting counter-offers, like "We are Borg. We will assimilate <list-of-cards>. You may assimilate <list-of-cards>. We are Borg," "Captain Kirk would never settle for less than <list-of-cards> for <list-of-cards>," and "I can't accept less than <list-of-cards> for <list-of-cards>." All of these were written specifically to fit Melvin's artistically defined personality; this is just one of the ways that I suggest for incorporating personality into a character's behaviors. Creating a variety of templates for each response leads to less repeated text. Also, I will demonstrate how I built Melvin to express his personality be choosing between templates based on elements of his character, like his emotions and relationships.

Loyall [Loyall96] is currently developing a more sophisticated language generator, which I expect will make the characters more believable than I was able to achieve using templates. Templates, however, allowed me to study social behaviors without first having to solve the language generation problem.

8.3 Creating a Behavior with Personality

This is the first of two sections describing how to create a specific social behavior for two distinct characters using my proposed methodology. In this section, I will show how to make a social behavior reflect various aspects of the character's personality. In the next section, 8.4, I will discuss how to model other agents for the purpose of negotiating with them and show that small amounts of representation can go a long way.

If an artist wants to create a social behavior that expresses the personality of a character, my methodology suggests that the artist focus on the following aspects of the character: personality quirks, competence, emotion, attitudes and relationships, roles, norms, robustness, and other goals the agent has. Many of these aspects of personality (e.g., robustness and personality quirks) will be important in non-social behaviors as well, while some (e.g., relationships) are more specific to social interactions. In this section, I will show how I incorporated each of these aspects of Melvin and Sluggo into their negotiation behaviors. I hope to provide a better understanding of the problems that are involved and to suggest some ways to solve them.

It is important to remember as I describe the various ways that an agent's personality can affect their behavior that I am talking about very specific personalities. Each of the decisions to be made about how to incorporate personality must be made separately for each character that is created. For instance, I am not making general claims about how, say, emotions affect negotiation—I am describing techniques for creating social behaviors that express a specific character's emotions in a way that is consistent with that character's distinct personality.

8.3.1 Competence

Although it is no more important than the other issues of personality, I have found that when building social behaviors, starting with competence is generally useful (perhaps this is my AI background seeping in and most artists will not want to start with competence).

In creating negotiation for Melvin and Sluggo, I first performed a general, but rather shallow, analysis of negotiation and used that, along with my knowledge of the personalities of Melvin and Sluggo, to determine roughly what their negotiation behaviors should be like. I didn't try to analyze the behavior too deeply because I knew that I wasn't going to build agents that were supposed to take part in complex labor negotiations (like Sycara's agents [Sycara88]). I was more interested in simpler agents that used simple forms of everyday negotiation.

I began by trying to understand some of the dimensions of complexity negotiation can take on. I came up with three that I found to be useful: the number of agents involved, the objects of the negotiation, and the types of offers that can be made and understood.

- **Number of agents involved.** One dimension of complexity is the number of agents involved. The simplest is two and it can grow without bound. Some multi-agent negotiations can be created out of two-agent negotiations, but not all—a three-way circular trade requires the agent to negotiate with two other agents simultaneously. One step up in complexity from simple two-agent trades is to put together links of two agent negotiations. Another step is to be able to carry on true multi-agent negotiations with either set numbers of agents (e.g., 3) or arbitrarily many agents.

- **Objects of negotiation.** Another dimension of complexity is the type of thing being negotiated for. In my simulated worlds where sensing is easy and reliable, negotiating for visible objects is simpler than having to model the other agent's possessions or asking about unsensed items the other agent is willing to trade. Other types of negotiation behaviors might be for services, knowledge, relationships, or other abstractions.

- **Types of offers made and understood.** A third dimension of complexity is the types of offers that the agent can make and understand. For example, an agent might be able to respond to offers from other agents but not be able to make offers. Another source of variation is whether the offers offered/understood include partial offers (e.g., "What will you give me for X?" or "What do you want for Y?") or full offers (e.g., "Would you give me Y for X?"). The final variation is in the number of items in the offers, whether it is only single items (e.g., "I'll give you X for Y.") or multiple items (e.g., "I'll give you X for Y and Z.").

When designing negotiation for Melvin and Sluggo, my first step was to decide where along these dimensions the behavior should fall. Interactions with agents that have complicated behaviors can be more interesting, but there is no need to create more complexity than is necessary for the environment in question and the personality of the agent.

Along the dimension of whom to negotiate with, I decided that interactions with single other users was appropriate for both characters. More complex negotiations might make them seem too mature and might be more confusing than interesting for the user.

Along the dimension of objects to negotiate for, I decided the agents should be able to trade baseball cards. Also, the characters are able to see what cards the other agents are holding.

Finally, I decided that Melvin should be able to offer and respond to single and multi-card full offers (e.g., "I'll give you Ruth for Mays.") and partial offers (e.g., "What do you want for Aaron and Jackson?"), but that Sluggo's intelligence should only allow him to offer and respond to one-for-one full offers or single card partial offers. Note that Sluggo's behavior is somewhat less competent than Melvin's. The competence of the behavior is just a way of expressing Sluggo's personality, which I feel comes across better through a less competent behavior. *In traditional AI, more competence has been the goal; for artists, competence is a means to expressing personality and this may sometimes be achieved by low levels of competence.*

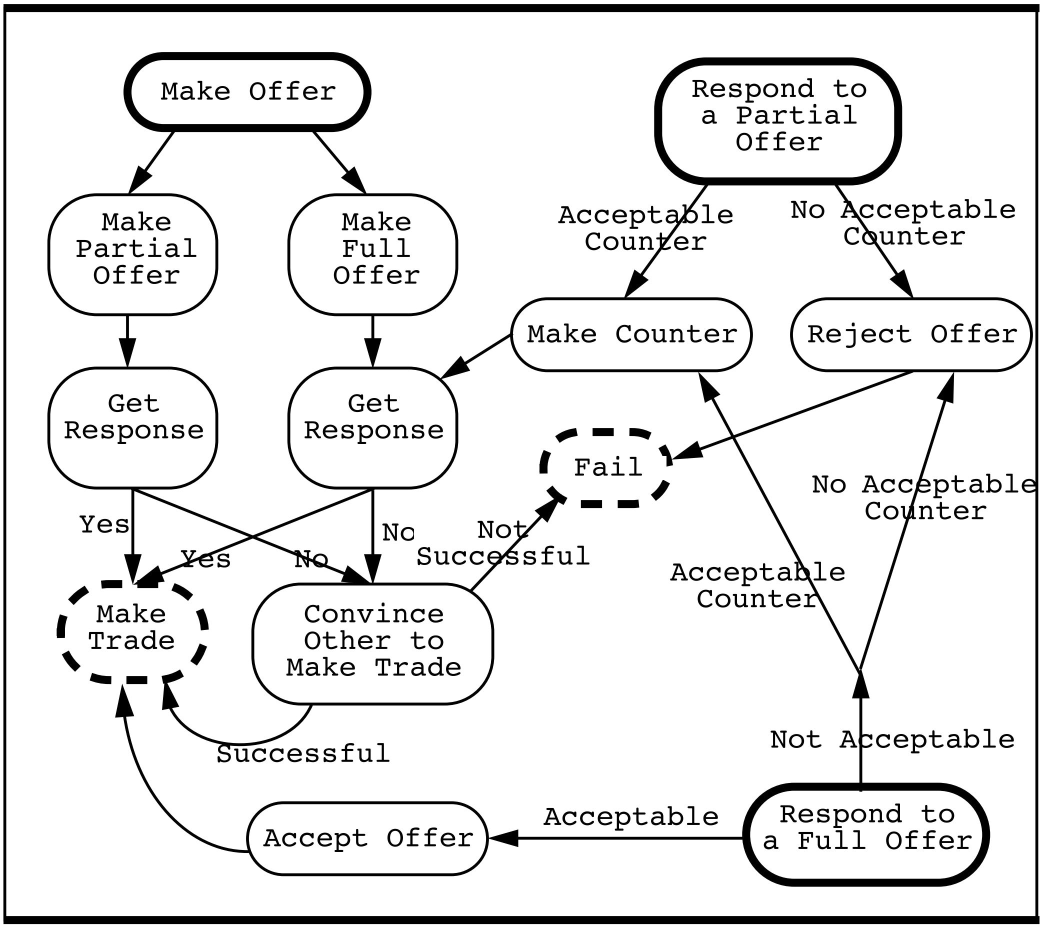
At this point, I began to code the two behaviors. Because the two characters have somewhat similar levels of competence (both are children capable of trading for baseball cards), I was able to share approximately 80% of the code between the two characters, with the other code accounting for Melvin's behaviors to respond to and offer multi-card trades. Once other aspects of personality are incorporated, the amount of shared code drops to roughly 50%. I expect this code transferral rate to be even lower when creating characters for different worlds because Melvin and Sluggo's behaviors are built with similar assumptions about their physical world and the types of objects (baseball cards) for which they negotiate. For instance, their negotiation behaviors assume that they can see all of the possessions that the other agent has to offer, which is not true in many domains.

Before coding behaviors, I find it useful to address competence and any other major constraints placed on the behavior by the character's personality. Once I address these issues and build a simple, working behavior, I add the other elements of personality in greater detail. This tends to take considerably more time than the initial coding as this is where the difficult problems arises—the behavior itself is usually quite simple, but making it personality-rich is difficult. Other artists may or may not find this to be a reasonable approach; like any complex programming task, there are many ways to proceed based on the style and preferences of the programmer.

The basic negotiation behavior I built is diagrammed in Figure 8-3. Despite the appearance of the diagram, the behavior is not very complex. The bold modules are starting points for the behavior: either the agent decides to offer a trade to another agent, another agent offers a partial trade (i.e., What do you want for X? or What will you give me for Y?), or another agent offers a full trade (i.e., I'll trade you X for Y.). The dashed modules are end-states; the behavior either fails or ends in the trading of cards, which was also built but not included in the diagram.

Here's an example. An agent offers Melvin a Catfish Hunter card for Melvin's Babe Ruth card, which starts in the bottom-right-hand corner of the diagram. Melvin decides that the offer is not acceptable, and then decides that there is a possible counter-offer he can make, Babe Ruth for Mickey Mantle, and he suggests it. From here, the behavior goes to the "Get Response" module which handles responses from the other character. If the other character says no, Melvin will try to talk the character into making the trade. If he is successful, the trade goes through.

FIGURE 8-3 **Outline of Negotiation Behavior**



Because it is not necessary to convey my main points, I do not include the full negotiation code (or pseudo-code) here, but an example will be useful. Figure 8-4 provides pseudo-code for responding to a full offer, which is part of the larger negotiation behavior. This is a somewhat simplified version of the original code since there is no consideration of making counter-offers to unacceptable trades. Throughout this section, I will show how to add various aspects of the characters' personalities to this behavior fragment.

FIGURE 8-4 **Pseudo-Code for Melvin and Sluggo Responding to Offers**

```
;; Top-level demon that fires when an agent offers
;; a full trade. A full trade includes cards offered
;; as well as cards requested. A partial trade only
;; includes one part of the offer and is handled by
;; a separate behavior.
Demon:
    Name: respond-to-trade-offer
    Input:  Actor (actor),
            offered-cards (o-cards),
            requested-cards (r-cards)
    Precondition: full trade has been offered by actor
    Code:
       react-to-offer(actor, o-cards, r-cards)

;; When respond-to-trade-offer demon calls react-to-
;; offer, there are a number of behaviors that might
;; be used. The first is an accept behavior. value is
;; an agent-specific function to evaluate a set of
;; baseball cards.
Behavior:
    Name: react-to-offer
    Input: actor, o-cards, r-cards
    Precondition: [Melvin]
       value(r-cards) <= value(o-cards)
    Precondition: [Sluggo]
       (and (value(r-cards) < value(o-cards))
            (length(r-cards) < 2)
            (length(o-cards) < 2))
    Specificity: 10
    Code:
       speak-to(actor, "I accept.")
       trade-cards(actor, o-cards, r-cards)

;; This is another possible react-to-offer behavior
;; the agent uses when the offer is unacceptable.
Behavior:
    Name: react-to-offer
    Input: actor, o-cards, r-cards
    Precondition: t
    Specificity: 0
    Code:
       speak-to(actor, "No thanks.")
```

This behavior fragment begins with a demon that waits for other agents to make full offers (e.g., "I'll give you Mays for Ruth."). When such an offer is made, a sub-behavior is initiated to respond to the offer. How the agent responds will eventually depend on many things, but for now the response is made based on the appraisals of the cards offered and, in Sluggo's case, the number of cards offered. The first react-to-offer behavior has a higher specificity value than the second, which means that it will be preferred if its precondition is true. If its precondition is not true, the second behavior is used. Even in this simple example, Melvin and Sluggo differ. Sluggo will automatically reject offers that are too complex and Melvin is generous enough to make trades where he comes out even.

8.3.2 Robustness

The idea of robustness is that agents need to stay in character even when events don't go as expected. (Recall from Chapter 7 that I have adopted a somewhat unusual definition for *robust*.) More specifically, Melvin and Sluggo need to be robust enough that they don't get out of character or do anything to break the user's suspension of disbelief, even when negotiations don't go smoothly. For instance, Melvin and Sluggo need to be robust in that they have to be ready for the user to wander away at any time or make offers unrelated to what had gone before.

The Hap language provides support for robustness in social behaviors in the same way it does for robustness in non-social behaviors, i.e. by means of demons and reactive annotations to goals and behaviors such as preconditions, success-tests, and context-conditions. All of these are described briefly in Chapter 1 and in [Loyall93]. Using these mechanisms, the artist needs to organize the behaviors to provide robustness in ways that fit the personalities of the characters. For instance, if the player makes a deal with a character, collects a card, and then wanders off without finishing the deal, the characters need to react—and they need to react in a personality-specific way. For instance, Melvin will shout, "Come back here you Romulan fiend" and then give chase.

Two Ways to Create More Robust Behaviors

I have found that two important ways to organize behaviors to be robust are: being prepared and being prepared for not being prepared. By being prepared, I mean to try to foresee as many contingencies as possible. Traditional AI agents that interact with other AI agents know ahead of time what kinds of things might be said to them since some protocol for interaction can be set beforehand. When interacting with a human, this cannot be done, so the agents should be prepared for a much larger range of possible statements and questions. By being prepared for not being prepared, I mean that it is typically impossible to foresee *all* possible contingencies, so behaviors have to be written to handle unexpected situations without appearing unreasonably confused.

Being Prepared. One way to be robust is to build the behavior with as many contingencies as possible. For example, after offering a trade, the behavior must be ready for as many kinds of responses as possible. The responses Melvin and Sluggo recognize are the following: yes, no, a counter-offer, handing over a card in the offer (assumes a ‘‘yes’’), a new full offer, both kinds of partial offers (e.g., “What do you want for X?” and “What’ll you give me for Y?”), and offers that cannot be made based on who currently owns what cards. By including as many responses as possible, I made the behavior more robust.

The `react-to-offer` behavior presented above (Figure 8-4) will produce unreasonable behavior when the offer made is not possible based on who has what cards. For example, if the player offers Willie Mays to Melvin for Melvin’s Babe Ruth, but Melvin doesn’t have a Babe Ruth card, Melvin might still accept the offer if he appraises the trade to be in his favor. This is fixed by adding a new respond-to-offer behavior with a higher specificity that checks this case. The higher specificity means that if another behavior also has a true precondition, this new behavior will still be the one to execute. Figure 8-5 provides a pseudo-code example of this fix.

Being Ready for the Unexpected. Another way to provide robustness is to provide general responses for situations that fall outside of the domain of the behavior. Believable agents cannot possibly be ready for every situation, but they shouldn’t seem confused at inappropriate times either.

When Melvin makes an offer and the user does something unexpected, Melvin will wait or maybe whistle. Maybe the user is looking over Melvin’s cards or just mistyped a command. In these cases, Melvin’s generic actions may be reasonable and allow the behavior to get back on track when the user is ready. If the user has ended the negotiation and begun a new behavior, Melvin will whistle briefly then begin a new behavior. This means Melvin doesn’t get stuck waiting on a response that never comes, but the behavior is also not so brittle that it falls apart when small interruptions occur.

FIGURE 8-5 **Pseudo-Code for a More Robust Negotiation Behavior**

```
;; When respond-to-trade-offer demon calls react-to-
;; offer, there are a number of behaviors that might
;; be used. The first is an accept behavior. value is
;; an agent-specific function to evaluate a set of
;; baseball cards. These behaviors are for Melvin.
Behavior:
    Name: react-to-offer
    Input: actor, o-cards, r-cards
    Precondition: value(r-cards) <= value(o-cards)
    Specificity: 0
    Code:
       speak-to(actor, “I accept.”)
       trade-cards(actor, o-cards, r-cards)

;; This is another possible react-to-offer behavior
;; the agent uses when the offer is unacceptable.
Behavior:
    Name: react-to-offer
    Input: actor, o-cards, r-cards
    Precondition: value(r-cards) > value(o-cards)
    Specificity: 0
    Code:
       speak-to(actor, “No thanks.”)
       trade-cards(actor, o-cards, r-cards)

;; ** New behavior **
;; Used when the agent has made an impossible offer
;; Takes precedence over other behaviors because of
;; higher specificity value
Behavior:
    Name: react-to-offer
    Input: actor, o-cards, r-cards
    Precondition: (or (actor doesn’t have o-cards)
                      (self doesn’t have r-cards))
    Specificity: 10
    Code:
       speak-to(actor, “That isn’t possible.”)
```

A similar kind of situation arises in language processing. Melvin and Sluggo are programmed to respond to a few statements and requests, but the user could potentially talk about anything. In these situations, the characters need to have a reasonable, but generic response. For example, here's one response Sluggo can give:

```
PLAYER> Say to Sluggo: How about them Steelers?
Sluggo says to you: ''Hey dork, shut up until I tell
you to talk.''.
```

In this case, Sluggo has no idea what the player has said, but instead of breaking the suspension of disbelief, I am able to turn Sluggo's response into an outlet for his personality. This is similar in spirit to Weizenbaum's Eliza program [Weizenbaum66] except that artists need to invent new, personality-specific "tricks." Eliza's therapist role made it possible to turn everything the user said into a question. Sluggo uses a different approach suitable to his personality.

Note that the language interactions are similar to Eliza when the characters do not understand what has been said, but that they are somewhat deeper in other cases. Essentially, Eliza never knew what was being said and always had to resort to "trick" answers. My characters are designed to handle certain situations and need to resort to these "tricks" when they get away from these areas and need to stay in character to be robust.

8.3.3 Social Norms

Incorporating social norms into behaviors means getting agents to act within normal social boundaries. This means both doing some things and not doing others. Both are hard problems.

An example of not doing something is that Sluggo should not go to sleep in the middle of a negotiation even if he's tired. In some cases social norms can be broken effectively; Sluggo getting angry and resorting to violence can be considered such an effective breach of etiquette.

An example of doing something to keep within social norms is Melvin apologizing to the player when he needs to break off a negotiation because Sluggo is coming his way.

Let's look at these two examples in a bit more depth.

Creating agents that don't do things when they aren't supposed to

Artists need to make sure that their agents don't do anything in an inappropriate context. An example is that agents can't just lie down and go to sleep when they get tired. They could be at a party, on the phone, or performing a death-defying high wire act at the time. Behaviors, like get-some-sleep, must only be used in certain situations, such as when in bed. If the agent isn't in bed when it gets tired, it must initiate other behaviors to get to its bed.

So far, I have found the basic Hap mechanisms of preconditions, goal priorities, plan specificities, context-conditions, and success-tests to be capable of solving these kinds of issues. The lesson here is simply to code the behavior to make sure the context is appropriate.

Creating agents that do extra things to meet social obligations

This problem turns out to be a bit harder. Here's an example: Melvin is in the middle of a trade with the player when Sluggo decides he wants to get in on the trading. Melvin is afraid of Sluggo, so his high priority goal to avoid people he is scared of kicks in. Now, the normal approach to such a problem is that the higher priority goal takes over and Melvin just walks away in the middle of the conversation. For some agents this might be acceptable, but I want Melvin to be polite and to say something like, "Sorry, but I've got to take off now. Why don't we finish this later?" The standard Hap language has no way of providing this extra pleasantry. This bit of dialogue is used solely to transition between behaviors; neither the negotiation nor the avoidance behavior needs Melvin to say this to succeed.

I have developed a somewhat ad hoc solution to this problem that I call *segue goals*. Segue goals will probably not scale up well as a solution, but they are capable of providing solutions at least some of the time. Sengers is exploring the use of "transition behaviors" to solve this problem in a more general way [Sengers96].

Segue goals are placed between behaviors and are responsible for creating a smooth transition. For instance, a segue goal might be placed between an on-going social behavior and an interrupting behavior. In the example above, Melvin's segue goal would be to speak to the player before walking away.

I have found the following to be important properties of segue goals:

1. Segue goals will sometimes need to reflect the goal/behavior being segued from. For instance, Melvin may want to say, "Let's finish this trade later" if it is a trade that is being interrupted, but this segue wouldn't work for other interrupted goals.
2. Segue goals will sometimes need to reflect the goal that is being segued to. For instance, Melvin may want to say, "I've got to go, Sluggo kind of scares

me and he's heading this way." He would want to say something different if the interrupting goal were something else.

3. Segue goals will sometimes need to have a priority, just like other goals. If Melvin is *very* scared, his `avoid-threat` goal might have an extremely high priority, which would override the segue goal as well as the negotiation goal. In this case Melvin would just walk away in the middle of the conversation—in some cases this is the proper thing to do.
4. Segue goals will sometimes need to be reactive just like other goals. If Melvin goes to say something to the user, but notices that the user has already rudely walked away, Melvin should not execute the segue. Hap annotations like success-tests and context-conditions are useful for situations like this.

I have implemented segue goals in the non-real-time version of Hap as follows: Any behavior can have a segue goal as its first step. The specific behavior spawned by that goal is determined by information about what other action was interrupted. This allows the segue to take into account both the goals being segued from and to.

The segue goal and the accompanying behaviors are normal Hap goals and productions with all of the typical annotations. This means that the goals can have priorities and success-tests and behaviors can have specificities and context-conditions (among others). This gives the artist a great deal of control in how to accomplish the segue.

The information about what goals have been interrupted is provided by keeping a list of all active goals and the time they were last worked on. By noting what active goal was last worked on, the segue behavior can infer what branch of the behavior tree was interrupted.

This approach works better than some other options, like providing each goal information about what just preceded it, because it handles interactions with non-physical goals better. In the example above, Melvin's negotiation behavior is first interrupted by an emotion goal that updates his fear, then that goal finishes and the goal to avoid Sluggo kicks in before ever returning to the negotiation. This means that the goal that proceeds the avoid goal is *not* the negotiation, but the emotion update goal. However, if Melvin has a negotiation goal in his APT and it was last worked on the previous turn, it was interrupted by the avoidance goal.

Figure 8-6 shows pseudo-code for handling the negotiation to avoid-threat segue. There are three separate behaviors. The first is the main avoid-threat demon that is keyed off of the fear of another agent, such as Sluggo. This behavior has two steps: the first performs the segue from the previous goal, the second moves Melvin away from the threat.

The second and third behaviors are two possible ways to handle the segue. The first `avoid-threat-segue` behavior is used when the previous behavior was negotiation. This segue results in saying to the negotiation partner, "Why don't we finish this later..." The other `avoid-threat-segue` behavior will be used if the previous behavior was not a negotiation and causes no behavior. So, if the previous behavior was not a negotiation, Melvin will start right in with step #2 of the avoid-threat behavior and move away from the threatening agent. The null behavior is necessary because the `avoid-threat` behavior will fail if there is no suitable behavior for the `avoid-threat-segue` step in the `avoid-threat` behavior; the existence of a null behavior is sufficient to keep that from happening. I make sure that the first `avoid-threat-segue` behavior is chosen when possible by setting the specificity values of the segue behaviors to prefer that behavior if it is applicable.

Figure 8-7 shows this segue in action. It bears repeating that the natural language capabilities of these agents are much shallower than they may appear; Melvin uses templates, key-word matching, and Hap behaviors for simple discourse management. Although the results are impressive, there are a number of drawbacks to this approach that I will discuss in Chapter 11.

This implementation of segue goals has worked fairly well and the potential for combinatorial explosion of having to do pairwise segues between all goals has not been a problem to date. Since most behaviors are not dynamically changing their priorities or being created on the fly, they never interrupt each other and the artist doesn't have to worry about segues. The artist may want to write a number of cases for each possible interrupted goal but some general segues can ease this burden if the artist doesn't feel that the more specific segues are necessary. I used such a generic segue (the same one for three variations of the negotiation behavior[1]) in the example above.

1. The three variations, for those curious, are: Melvin initiates a trade, the other agent initiates a trade with a partial offer, and the other agent initiates a trade with a full offer.

FIGURE 8-6 **Pseudo-Code for Handling the Negotiation to Avoid-Threat Segue**

```
;; A demon to avoid feared agents that are nearby
;; First action is to segue from previous behavior
;; Then move away from threatening agent
Demon:
    Name: AVOID-THREAT
    Input: threat
    Precondition:
       (and (behavioral-feature(threat)='defensive)
            (threat is nearby))
    Code:
       avoid-threat-segue(threat)
       move(away from threat)

;; Segue from negotiation to avoid-threat behavior
;; Checks that previous behavior was negotiation and
;; that the threat isn't the same as the agent being
;; negotiated with
Behavior:
    Name: AVOID-THREAT-SEGUE
    Input: threat
    Precondition:
       (and (or
              (previous-goal=react-to-offer)
              (previous-goal=respond-to-partial-offer)
              (previous-goal=trade-for-stuff))
            (threat != interactor in previous-goal))
    Specificity: 10
    Code:
       speak-to(interactor,
            "Why don't we finish this later...")

;; Low-specificity segue that is used if no other segue
;; is appropriate. This behavior always succeeds, so
;; avoid-threat behaviors works
Behavior:
    Name: AVOID-THREAT-SEGUE
    Input: threat
    Precondition: t
    Specificity: 0
    Code: NIL
```

FIGURE 8-7 **Trace of Melvin in a Segue Between Negotiation and Avoid-Threat**

```
You go into the sand box.
Sluggo spits.
Melvin is now smiling.
Melvin is speaking to you.
Melvin's voice says ''Salutations, Vulcan ambassador!
The Klingon high command has sent me in search of base-
ball cards.''.
PLAYER> Melvin: What do you want for Mantle?

You are speaking to Melvin.
Player's voice says ''What do you want for Mantle?''.
Sluggo smokes a cigarette.
Melvin is speaking to you.
Melvin's voice says ''The aliens told me to offer you
Mickey Mantle in return for Babe Ruth.''.
PLAYER> Sluggo: Hey dork, get a life!

You are speaking to Sluggo.
Player's voice says ''Hey dork, get a life!''.
Sluggo is now red.
Sluggo is now scowling.
Sluggo is now tense.
Sluggo goes into the sand box.
Melvin is now pale.
Melvin is now bug-eyed.
Melvin is now trembling.
Melvin is speaking to you.
Melvin's voice says ''Why don't we finish this lat-
er...''.
PLAYER> Sluggo: What do you want, butthead?

You are speaking to Sluggo.
Player's voice says ''What do you want, butthead?''.
Sluggo punches you.
You are now wounded.
Melvin gets on the jungle gym.
```

8.3.4 Social Roles

The nature of the playground world means that social roles don't play a very important part. In other forms of negotiation, this may not be true. In the holdup simulation, the gunman and police officer (the user) can negotiate a little bit, though the behavior is not nearly as developed as the behaviors in Melvin and Sluggo. In this case, the gunman is only willing to negotiate with police officers (as opposed to cashiers or other types of characters). Therefore, his negotiation behavior encodes this knowledge about social roles in the precondition for responding to offers.

8.3.5 Emotions

Building emotional versions of social behaviors is simplified by the fact that Tok provides an action architecture that is already coupled with an emotion architecture (Em). If things are set up properly, Em automatically generates emotions and helps express them, which takes some of that burden off of the artist. Nonetheless, there is still work to do.

The coordination of emotion and social behaviors has two components. First, the behavior may be the source of emotions. Artists need to put information in the behavior to let Em generate such emotions, just like they need to do for non-social behaviors. Second, emotions may affect behavior. This is the more difficult of the two for the behavior builder. I will discuss both of these.

Generating Emotions from Negotiation

The Em system, as described in Chapter 3, uses a set of default emotion generation rules that generate emotions based on (among other things) simple annotations added to important goals that include the following: the importance of success, the importance of not failing, a likelihood of success function, a likelihood of failure function, a function to determine who is responsible if the goal succeeds, and a function to determine who is responsible if the goal fails.

Here are some of the ways that Melvin and Sluggo generate emotions while negotiating. This list might be useful to artists creating negotiating agents who want ideas about where emotions might need to be generated. Artists will need to fit the items in this list to their particular characters; they may also use elements that are not on this list and not use elements that are on this list. This is an artistic process, so there are not going to be right and wrong answers—although, as with traditional arts, some may be more aesthetic than others.

- **Expecting cards from a trade.** If the user stalls while Sluggo or Melvin is expecting his cards, the character finds the important goal to get the cards offered in a trade is more likely to fail. This leads to *fear*, *distress*, and *anger* towards the player. Because the character also believes that this goal may still succeed, Em also generates some *hope*. Which emotion most affects behavior is based on their relative intensities and the personality of the agent. Em also generates *joy* and *gratitude* when the trade is successful or *distress* and *anger* when the other agent fails to fulfill the bargain. Melvin differs from Sluggo in that his negative reactions to stalls are often less strong.

Figure 8-8 provides pseudo-code of the `trade-cards` goal with emotional annotations. In this example, the `compute-lof` annotation of the `trade-cards` goal causes Melvin's belief about the current likelihood that the exchange of cards will fail to be stored in the variable, `$trade-lof-var`. This variable is updated in the trading behavior to increase when the other character stalls, which leads to the *fear*, *anger*, and *distress* reactions I just described. The `compute-los` function, which for Melvin always returns 0.75, causes him to always have some *hope* that the goal will be successful. Finally, when the trade ends in success or failure, the `responsible-for-failure` and `responsible-for-success` annotations of Melvin's `trade-cards` goal cause Melvin to assign credit or blame to the other agent in the trade, which leads to either *anger* or *gratitude*.

- **Dealing with rejection.** Sluggo generates mild *anger* when another agent rejects one of his offers, the other agent ignores his offer, or the other agent walks off in the middle of a negotiation. These are all based on the failure of Sluggo's goal to make deals.

- **Getting new cards.** One of Sluggo's important goals is to get good cards, so he generates *joy* when he makes a good trade, he is given a card, or he is able to get new cards off of someone that he has beaten up. Melvin also likes to get new cards and will generate *joy* when he makes a good trade or is given a card as a gift.

- **Social interaction.** Melvin generates *joy* when interacting socially with agents he likes. He gets happy from negotiation even when it doesn't result in a trade and he generates *joy* when another agent greets him, talks to him, or asks him questions. These are all important goals that succeed in the proper contexts. He gets especially happy when the other agent humors him by using Star Trek terminology, which is caused by a goal that succeeds when it notices speech that contains any of a set of Star Trek keywords.

- **Making a deal.** Melvin generates *joy* when a trade has been negotiated. This is different from Sluggo, who only generates *joy* once the card he wants is in his hand. Melvin becomes even happier once the cards have actually changed hands successfully.

FIGURE 8-8 **Adding Emotional Annotations to Negotiation for Melvin and Sluggo**

```
;; Melvin accepting offer behavior:
;;  Melvin accepts a trade and then begins swapping
;;  cards. The swapping behavior can be a source of
;;  emotions because of the emotional annotations that
;;  have been added to it.
Behavior:
    Name: react-to-offer
    Input: actor, o-cards, r-cards
    Precondition: value(r-cards)<=value(o-cards)
    Specificity: 0
    Code:
       speak-to(actor, "I accept.")
       ;; The "with" statement contains a number
       ;; of emotion annotations for the trade-cards
       ;; goal. The importances represent how intense
       ;; the emotional reactions to this goal should
       ;; be. Ex: if successful, this goal leads to
       ;; a joy structure of intensity 5 because of
       ;; the importance-of-success. The compute-*
       ;; functions return a list of how likely the
       ;; goal currently is to succeed/fail and who
       ;; to assign credit/blame to. Melvin always
       ;; thinks the goal is 75% likely to succeed and
       ;; has a variable representing how likely it is
       ;; to fail. The var is updated in the trade-cards
       ;; behavior. The responsible-for-* functions
       ;; assign credit/blame to the other agent when
       ;; the trade ends either in success or failure.
       with {
             importance-of-success=5;
             importance-of-failure=5;
             compute-los=(0.75, actor);
             compute-lof=($trade-lof-var, actor);
             responsible-for-failure=actor;
             responsible-for-success=actor;
             }
          do {
             trade-cards(actor, o-cards, r-cards)
             }
```

- **Understanding the offer.** If the user offers Sluggo a trade that's too complex for Sluggo to understand, Sluggo will get *angry*, *distressed*, and *reproachful* towards the player for making him feel stupid.

The Effect of Emotions on Negotiation

The negotiation behavior needs to reflect the agent's emotional state, whether that state is a result of negotiation-related events or not. There are both general and negotiation-specific ways that emotions influence negotiation.

Recall, from Chapter 5, that emotion structures do not directly affect behavior; they do so indirectly through the behavioral feature system. I will describe these effects as being the result of emotions, but the examples will use behavioral features. Melvin's behavioral feature map can be found in section 5.1.

Four general effects are the following: changes to appraisals, changes to goal priorities, changes in willingness to interact, and changes in particular action choices. These types of changes are general in the sense that they can be seen in other social behaviors as well. Many of the previously described effects that emotions can have, including influencing action, inferencing, and language (all of which are described in Chapter 5), can be used in the context of social behaviors. I will present the four that I have found most useful, but the more complete list of effects can also be a source of inspiration for artists.

- **Changing appraisals.** Emotions can change how an agent appraises objects. Melvin is more likely to make trades with agents when he is *cheerful* because he appraises the other agent's offers more favorably.

  More generally, appraisals can take many forms that are important for social interactions (e.g., appraising how much I want to be your friend, appraising how trustworthy I think you are). Emotions can be incorporated into many of these appraisals.

- **Changing goal priorities.** Emotions can affect the priority of negotiation goals. If Sluggo feels *fear* that his goal to get the cards from a trade is in jeopardy, this normally medium-high priority goal will take on a higher priority. This makes Sluggo very persistent about achieving this goal before doing anything else. I extended the Hap language to allow dynamic priorities for goals so a goal's priority can be a function that takes emotion information into account (see Chapter 5).

  More generally, many behaviors (social and non-social) should reflect threats to the agent's goals—variable priorities provide one technique for doing this that I have found useful.

- **Changing the willingness to interact.** Emotions influence how likely an agent is to interact. Melvin's demon to respond to offers has a precondition that makes sure that Melvin is not acting *unfriendly* or *withdrawn* before firing. A pseudo-code example is provided in Figure 8-9.

  More generally, many social behaviors will take emotions into account when deciding whether or not to interact with various agents.

- **Changing the style of actions.** Agents will often change details of their actions based on their emotions. Melvin might agree to a trade by saying, "Okay. I accept" or "Sure! I'd be happy to make that trade my Vulcan friend." The choice of which template to use is based, in part, on Melvin's current emotional state. An example is provided in Figure 8-9.

  More generally, I have found the style of actions (speech in particular) to be a useful way of expressing emotions. This can be done in speech through word choice and inflection. In other actions it takes other forms, such as stomping or skipping instead of walking. This is not limited to actions that are part of social behaviors, but is certainly true for social actions.

When designing a social behavior artists may start with the general kinds of influences of emotions above, but many behaviors will have their own unique ways of expressing emotions as well. Here are some negotiation-specific ways of expressing emotions that I used in Melvin and Sluggo.

- **What to trade/not trade.** Agents might have specific preferences or restrictions on what they are or are not willing to negotiate for based on their emotional state. For instance, Sluggo might completely rule out all trades of his Willie Mays card except when particularly *cheerful* (he does not currently do this). This can be coded into the appraisal function or into the preconditions of the behaviors that make and respond to offers.

FIGURE 8-9 **Expressing Emotions through Negotiation (Pseudo-Code)**

```
;; Extension to accepting react-to-offer behavior:
;; * be more generous when friendly, generous, or
;;     good-mood features are present
;; * do not accept when unfriendly or withdrawn
;;     features are present
;; * respond enthusiastically when friendly or
;;     good-mood features are present
;; (Melvin's full set of features can be found in
;; section 5.1.5. Note that not all of Melvin's
;; features need to be explicitly expressed in every
;; decision he makes.)
Behavior:
    Name: react-to-offer
    Input: actor, o-cards, r-cards
    Precondition:
     (and
       (or (value(r-cards)<=value(o-cards))
           (and
             (or (behavioral-feature(actor)=friendly)
                 (behavioral-feature(actor)=generous)
                 (behavioral-feature()=good-mood))
             (value(r-cards)-2 <= value(o-cards))))
       (behavioral-feature(actor)!=unfriendly)
       (behavioral-feature()!=withdrawn))
    Specificity: 0
    Code:
       if (or (behavioral-feature(actor)='friendly)
              (behavioral-feature()=good-mood))
          speak-to(actor, "Sure! I'd be happy to.")
       else
          speak-to(actor, "I accept.")
       trade-cards(actor, o-cards, r-cards)
```

- **Generosity of offers.** Agents might act more or less generously in their offers and willingness to accept offers based on emotions. Agents that are feeling gratitude towards another agent will likely be more generous toward that agent; agents that are angry at an agent will be less generous. This is true of Melvin—Sluggo is never particularly generous. This is accomplished by modifying the preconditions of various behaviors to allow more or less generous trades to go through. An example is provided in Figure 8-9.

- **Making counter-offers.** Agents can make decisions about what counter-offer to make based on emotions. For instance, if Sluggo is in a reasonably good mood, he will usually follow up an unacceptable offer with a counter-offer. If he's in a bad mood, he'll reject the offer completely and end the negotiation.

8.3.6 Relationships and Attitudes

The relationship between two agents should affect the way the negotiation unfolds and, when things go particularly well or poorly, the relationship itself should change based on the negotiation. Also, agents' attitudes about other agents can influence and be influenced by negotiation. To clear up any confusion about terminology, relationships are things like "friends" and "lovers," while attitudes are things like "trust" and "liking." While these are different concepts, I have found that they tend to relate to social behaviors in similar ways and I will typically discuss them as a pair.

The Influence of Relationships and Attitudes on Negotiation

Some of the general ways that emotions can affect social behaviors also apply to the ways that relationships and attitudes can affect social behaviors. Some of the general effects are on the following: willingness to interact, appraisals, and style of action. Goal priority is one area that I have not used as a way to express attitudes and relationships as I did with emotions. That does not mean that it is not important, just that I have not used it—other artists might find it a useful form of expression. This is not meant to be an exhaustive list, just a useful starting point. In particular, specific characters and behaviors will have additional ways of incorporating attitudes and relationships.

The list below represents how Melvin incorporates attitudes and relationships into some important negotiation decisions. All of these examples happen to fall into the general classes of effects just laid out. Other negotiating agents will probably want to incorporate attitudes and relationships into these decisions, but will want to handle them in a way that corresponds to the character's personality. For the items in the list below, pseudo-code for the examples related to responding to offers can be found in Figure 8-10.

- **Willingness to Interact.** Melvin prefers initiating negotiations with friends and liked agents. His offer-initiation behaviors have preconditions that check his attitudes and relationships with potential trading partners. Also, when deciding how to respond to an offer, Melvin rejects offers from enemies and *feared* or *disliked* agents.

- **Appraisals.** Melvin will appraise offers differently from friends than from other characters, giving offers from friends and *liked* agents more weight.

- **Style of Action.** When speaking to another agent, the tone can change while the message remains the same. When Melvin accepts an offer, he can say, "Okay, I accept." or "Sure! I'd be happy to make that trade my Vulcan friend!" Each of these has the same basic content, but which is chosen is still very important from a believability standpoint. This choice is influenced by the relationship between Melvin and the other agent, Melvin's attitudes about the other agent, and Melvin's current set of behavioral features. Figure 8-10 shows how all of these influences come to bear on a single choice.

The Influence of Negotiation on Relationships and Attitudes

If an agent cheats Melvin, Melvin will like that agent less because of a negative emotional reaction to being cheated. As described in Chapter 5, Em provides default behavior that causes *like* and *dislike* attitudes towards agents to change based on *anger* and *gratitude* emotion structures of sufficient intensity. In Melvin's case, if the player cheats him, he will have an *anger* reaction towards the player that will result in him *liking* the player less. In this case, the artist needs to make sure that Melvin's goal to get his cards in a trade is set to be an important goal, which will lead to an emotional reaction when it fails. Otherwise, the artist does not have to do anything to cause this change in attitude.

This change in attitude will cause Melvin to act differently in subsequent negotiations because I have designed his negotiation behavior to take attitudes into account as I just described. For example, he will be less likely to accept offers and will be less enthusiastic when he does accept an offer.

FIGURE 8-10 **Adding Attitudes and Relationships to Negotiation**

```
;; Extension to accepting react-to-offer behavior:
;; * reject trades from feared and disliked agents
;; * be more generous to friends and liked agents
;; * respond enthusiastically to friend/liked agent
;;     if not in a bad mood
;; * emotion extensions are also present (see Fig. 8-9)
Behavior:
     Name: react-to-offer
     Input: actor, o-cards, r-cards
     Precondition:
      (and
        (or (value(r-cards)<=value(o-cards))
             (and
               (or (attitude(actor)=like)
                    (relation(actor)=friends)
                    (behavioral-feature(actor)=friendly)
                    (behavioral-feature(actor)=generous)
                    (behavioral-feature()=good-mood))
               (value(r-cards)-2 <= value(o-cards))))
        (behavioral-feature(actor)!=unfriendly)
        (behavioral-feature()!=withdrawn)
        (attitude(actor)!=dislike)
        (attitude(actor)!=fear))
     Specificity: 0
     Code:
        if (or (behavioral-feature(actor)=friendly)
               (behavioral-feature()=good-mood)
               (and
                  (not (behavioral-feauture()=bad-mood))
                  (or (attitude(actor)=like)
                       (relation(actor)=friend))))
           speak-to(actor, “Sure! I’d be happy to.”)
        else
           speak-to(actor, “I accept.”)
        trade-cards(actor, o-cards, r-cards)
```

8.3.7 Interactions with Other Goals

Agents' behaviors are not happening in a vacuum. A system where a single "current best goal" exclusively determines the agent's behavior can generate potentially unbelievable behaviors. For instance, if an agent has a goal to buy groceries and a goal to buy a hammer, the agent might want to go to a shopping center that has a grocery store *and* a hardware store. If one goal is chosen to act on, say the groceries, the agent might go to a grocery store that isn't near the hardware store. This behavior doesn't necessarily mean the agent is unbelievable; it could be a disorganized character. However, for some (more organized) characters, this behavior might break believability. Note that although the example I have given is one where the agent can be unbelievable because of a lack of competence, it is possible to create other examples that produce behaviors that are inconsistent other aspects of the agent's personality. Such an example is given below.

One of the problems with interacting goals is that a behavior that has to account for every other possible ongoing behavior has a lot to worry about. The number of interactions is at least $O(n^2)$ in the number of goals and possibly worse if there are cases when the current combination of other goals present is important. Anything that can be done to allow for goal interactions without this blow-up is a step in the right direction.

This problem is not particular to social interactions. Nonetheless, it is a problem that must be addressed to handle social interactions properly. Believable agents engaging in social interactions will often need to express other goals through their social behaviors—and they should do this in a way that expresses the personality of the agent most clearly.

One solution is to use the behavioral feature mechanism of the Em system (see section 5.2) to provide a simple form of communication between behaviors. Behavioral features help determine the following: what agent to act in a particular way towards, how to act towards them, and why. The *how* is encoded in the type and intensity fields. The *why* is encoded in the cause field. The *who* is encoded in the direction field.

It is possible for behavioral features to be created by goals independently of emotion processing. For instance, Melvin's goal to become the user's friend creates a behavioral feature for acting friendly towards the player. Other behaviors, such as negotiation, are (if my methodology is followed) built to take behavioral features into account as I described in section 8.3.5. This allows the code for negotiation to implicitly pursue two goals (negotiation and making friends) simultaneously.

This mechanism works well for goals that can be expressed as features, but is not a general approach to this problem. A few more examples of goals that could be expressed this way include the following: console another agent, avoid another agent, and act gratefully towards another agent.

8.3.8 Personality Quirks

Artists tell us that it is very important for characters to have very distinctive, quirky personalities. Personality quirks can be just about anything, from a character's accent to the way the character looks at life. Each quirk needs to be fit into the agent's behaviors in a manner suitable to the particular quirk. Quirks, by their very nature, are quirky, so I have not been able to draw general lessons about how to incorporate quirks into behaviors. Here are a few ways I expressed Melvin and Sluggo's personality quirks in their negotiation behaviors:

- **Handling uncooperative trading.** Melvin and Sluggo approach the problem of an uncooperative user differently, as can be seen in Figures 8-11 and 8-12. Melvin is persistent but polite. If pushed too far too often, he will eventually tell the teacher and get the user in trouble[1]. Sluggo goes into the trade with an impatient and aggressive attitude and quickly moves towards more aggression and threats which finally end in violence. I achieved this by writing two distinct behaviors for handling uncooperative agents in Melvin and Sluggo—in other words, these are not aspects of a single behavior differentiated only by some personality "knobs," they are significantly different behaviors designed for the same purpose but for different personalities.

- **Melvin as Trekkie.** Early versions of Melvin were similar to the final version in terms of most aspects of his personality. However, at one point I decided that his personality was too generic, so I made him a Trekkie. I was able to get this personality across simply by changing Melvin's speech templates. For example, in Figure 8-4, I changed "No thanks" to "I'd like to trade with you, but the Klingon High Command finds that offer unacceptable." The response to his more distinctive personality has been positive, except for one user who thought he was too stereotyped. Despite this reaction, I feel that the effort to avoid a generic "nerd" personality resulted in a more interesting character.

1. There is no teacher agent in the simulation, but the simulation can end with a message to the user that this is what has happened.

FIGURE 8-11 **How Melvin Swaps Cards**

```
Melvin says to you `‘Sure!  I’d be happy to make that
trade, my Vulcan friend.’’.

PLAYER> Say to Melvin: Okay, give me Mickey Mantle.
Melvin offers a Mickey Mantle trading card to you.

PLAYER> Take the card
Melvin waits.

PLAYER> Say to Melvin: Thanks.
Melvin says to you ``Can I have Babe Ruth now?’’.

PLAYER> Say to Melvin: No.
Melvin says to you ``Please, won’t you give me Babe
Ruth?’’.

PLAYER> Melvin: Take a hike.
Melvin is now frowning.
Melvin gets on the jungle gym.

PLAYER> Wait
Melvin sulks.
```

- **Generic actions.** There are a number of situations where Melvin or Sluggo need to kill a turn or two when nothing else is going on or when they are waiting on another agent. I call these stall actions, but choosing which action to perform during the stall is important. Melvin will whistle, adjust his eye glasses, or look intently at some cards. Sluggo uses this time to spit, swear, and look intently at the other agent. I have found that even these little, seemingly unimportant actions have a good deal of impact on the way the user views an agent.

- **Sluggo as hard-bargainer; Melvin as push-over.** Sluggo is very strict about what trades he will and will not make. Only trades that strictly benefit him are considered at all. Melvin is willing to make trades where he comes out even and, as I described, other aspects of his personality (e.g., his emotions) can lead to him accepting trades where he comes out worse-off.

FIGURE 8-12 How Sluggo Swaps Cards

```
PLAYER> Say to Sluggo: Um, OK.  I guess I'll make that
trade after all.
Sluggo offers a Jose Canseco trading card to you.

PLAYER> Take card
Sluggo says to you: ``So, hand over the card, twerp.''.

PLAYER> Say to Sluggo: Hey, lighten up!
Sluggo looks angrily at you.

PLAYER> Look angrily at Sluggo
Sluggo says to you: ``You don't know who you're messin'
with, dork!''.

PLAYER> Say to Sluggo: You don't scare me.
Sluggo is now frowning.
Sluggo says to you: ``Prepare to be pounded!''.
```

In this section, I stepped through the various aspects of personality my methodology suggests to incorporate into a social behavior. I also provided examples of some of the techniques that I used to build these aspects of personality into two negotiation behaviors. In the next section, I will turn to the second part of the methodology, which is concerned with modeling other agents in the environment.

8.4 The Role of Representation in Negotiation

The second part of the methodology suggests creating believable social behaviors with minimal amounts of representations of other agents. This does not mean no representation or even small representations; if a lot of representation is needed, then use it. However, I have found that the amount of representation is often much less than might be assumed. For example, negotiation is a behavior where it might seem necessary to use reasonably deep modeling of the other agents in the environment. For some characters and tasks this may be true, but it is not inherent in negotiation, as I will show for Melvin and Sluggo.

For Melvin and Sluggo, the only representation of other agents that I needed to add was a list of the offers other agents rejected in the past so as not to repeat them. This representation implicitly models the appraisals of other agents that are relevant to this particular task; in particular, modeling what trades an agent doesn't want to make implicitly models the agent's relative appraisals of those sets of cards. With this minimal amount of representation, users found the behaviors robust, personality-rich, and believable as I will describe in Chapter 10.

In some cases, I was able to give the appearance of having deeper representations than were actually used. For instance, I used the integration of the emotion/attitude system to make Melvin appear to have a representation of the trustworthiness of the other characters. If Melvin and the player are trading and the player cheats Melvin, Melvin will act more wary of the player in subsequent interactions (as described in section 8.3.5). It would seem that Melvin would have to have some model of the trustworthiness of the player. Although I could use such a representation, I accomplish this by, first, having Melvin's negotiation behavior change his liking attitude[1] about the player when the player cheats him and, second, by building a negotiation behavior that takes Melvin's attitude-based emotions into account.

In other words, because of the tight integration of the emotion/attitude system with the social behavior, it is possible to create behaviors, like negotiation, that might be thought to require additional modeling of other agents, without actually adding additional representation. In this case, Melvin's Em system is already implicitly modeling the relevant aspects of the other agent by way of Melvin's attitudes. *By using a broad agent architecture, artists may find that certain behaviors that might otherwise require additional representation, don't.*

Another example of how I created the appearance of deep models is typified by the following exchange:

Player: Melvin, will you give me Mickey Mantle for Ted Williams?
Melvin: We are borg. We will assimilate Ted Williams and Babe Ruth. You may assimilate Mickey Mantle. We are borg.

In this example, Melvin seems to know what the Player is interested in trading with and trading for. In other words, it seems like Melvin might represent the player's goals and possibly appraisals of various cards. In fact, this is a much simpler behavior that simply uses the previous offer to generate the counter-offer. Melvin needs no representation at all to accomplish this. This technique is somewhat similar to techniques used in Eliza [Weizenbaum66] where pattern matching and templates are used to replace knowledge and representation. What

1. I could also have used a trust attitude. In either case, Melvin knows how he feels about the agent but he doesn't rely on any independent model of the agent to know this.

I've shown is that versions of these techniques can sometimes be used effectively in more complex characters engaging in more responsive kinds of interactions. Although this approach might be unsuitable for more complex behaviors that require additional representation to achieve greater competence, it was quite robust in the face of users interacting with Melvin and Sluggo.

I am not claiming that other representations won't ever be necessary to create competent, believable negotiation. It may be that more complex representations will be needed for some believable behaviors. By building representations that are specific to the character, behaviors, and world, however, I have found that simple, specific representations can often take the place of powerful, general representation schemes. Furthermore, since the artist's goal is to create believable agents, which are rarely going to be called on to be deeply competent, the behaviors will not typically require the deep models that have been used by traditional AI to achieve deep competence.

8.5 Summary

In this chapter, I described how to apply the methodology for building believable social behaviors (described in the previous chapter) to building believable negotiation behaviors for Melvin and Sluggo in *The Playground*. During this process, the following important issues came up.

- I described various types of competence in negotiation and how competent I chose to make Melvin and Sluggo. This type of discussion is very different than goes on in most AI tasks, where high competence is assumed to be the goal.
- I defined robustness as the property of "staying in character" even in unexpected situations. I described two approaches to problems of robustness: being prepared for many contingencies and being prepared for not being prepared.
- I described the two-part problem of integrating emotions into social behaviors: having emotions affect the behavior and having the behavior lead to appropriate emotions. I described some general ways that emotions will tend to affect social behaviors and some specific ways that emotions affect negotiation. I also showed how artists enable Em to generate emotions from social behaviors. This is done by annotating important goals in the behavior with information that is used by the emotion generation rules, such as how important it is that the goal succeed.
- I presented some ideas about incorporating relationships and attitudes into social behaviors. The general list of ways to affect social behaviors that was in-

troduced in the emotion discussion was reused in this context. It is also possible for attitudes and relationships to change based on social interactions; I showed how the emotion system can be used to help with this task.

- While building social norms of behavior into Melvin, I found that in order for him to disengage from negotiation politely, he needed to be able to transition between behaviors smoothly. To handle this problem, I introduced segue goals that can be used to produce transitional behaviors between goals.

- Agents will typically have many goals active at one time and agents may want to negotiate differently based on their other goals. One way of allowing negotiation to change based on the presence of certain types of other goals is to use the behavioral feature system described in Chapter 5.

- I was able to create somewhat competent, believable negotiation behaviors with the only representation of other agents being a list of trades that other agents had previously rejected, which is considerably less than might be expected. I described some of the techniques I used to minimize representation while giving the appearance of having deeper models of others.

CHAPTER 9

Understanding the Methodology II: Making Friends

The main goal of this chapter is to further explore the methodology that I have proposed for building social behaviors. In this case, I use the making friends behavior of Melvin from *The Playground*.

I begin by motivating the choice of making friends as a behavior to build and study. I then break down making friends into a number of behaviors that can be used to achieve the larger goal, including helping others achieve their goals and modifying other behaviors to be more friendly. I use each of these behaviors to explore aspects of the methodology for building social behaviors.

Recall that the methodology has two main components. One is a set of heuristics for incorporating personality into social behaviors. The other is an approach to modeling the other agents in the world.

I describe how I added a number of different aspects of Melvin's personality to his behavior for helping other agents achieve their goals. This demonstrates how some of the techniques I discussed in the previous chapter apply to a different behavior. I hope that this will make this aspect of the methodology more clear.

I also describe the simple representation scheme Melvin uses when modeling other agents in order to help others. He needs to model the goals and appraisals of other agents; however, by using my knowledge of Melvin, Melvin is able to use very small and specific representations that are sufficient for the task and his personality. Again, this discussion will be similar to the one about the representations used in negotiation. I will use this behavior, however, to explore a few new ways to limit the representations that agents need to have of other agents.

I also revisit the idea, first presented in the previous chapter, that behaviors can communicate with each other using the behavioral feature system provided with Em, as long as the behaviors are built following the proposed methodology. Melvin's making friends behavior is able to modify other behaviors, like negotiation, using this technique.

9.1 Motivation for Building a Making-Friends Behavior

My approach to showing that my methodology is generally useful was to show how it could be used to build a varied set of social behaviors. In total, I built nine different social behaviors: negotiation, making friends (in a number of varieties), and seven more that I discussed briefly in Chapter 7. To contrast with negotiation, I chose making friends because it involves the social goal of being someone's friend, while negotiation is used to achieve the asocial goal of possessing some object—making friends is about relationships, negotiation is (at least usually) about physical objects.

Another reason for implementing a making friends behavior is that it is not an area where much research has been done in AI. Negotiation has already been studied within AI and is considered a "useful" skill. Making friends is a new kind of problem that assumes agents will have goals about having relationships, which is not a typical AI problem. I chose making friends to demonstrate some of the new kinds of problems that are encountered as we move from problem solving agents to believable agents. I implemented negotiation to show how believable negotiation contrasts with a traditional-AI behavior like competent negotiation; I implemented making friends to show that many new kinds of behaviors artists may need to build do not have counterparts in traditional AI at all.

9.2 Ways to Make Friends

Making friends is not really a behavior; it is a set of behaviors with a common goal. I will start, therefore, by briefly laying out some of the behaviors that can be used to make friends. I will spend the rest of the chapter looking at how to apply the methodology to a few of these individual behaviors.

1. **Complimenting.** An agent can compliment the other agent in a variety of ways. For example, Melvin will occasionally tell the player how smart he thinks he/she is or how much he enjoys hanging out with him/her.

2. **Helping others achieve goals.** An agent can offer assistance to the other agent in achieving various goals. If Melvin is aware of the player trying to get a certain baseball card (e.g., by being asked by the player for help or by overhearing the player offer a trade to Sluggo), Melvin may offer to help him/her achieve that goal.
3. **Modifying other behaviors to be more friendly.** Other behaviors that aren't specifically designed to make friends can also be modified to make them more friendly. Melvin will negotiate in a more friendly manner, using his Star Trek banter and offering better deals, when he is trying to become the player's friend.
4. **General interaction.** An agent can simply spend time with the other agent or engage the other agent in social behaviors for the sake of interacting (as opposed to pursuing some other goal). Melvin will use behaviors like negotiation just to spend time with the player. This doesn't always lead to a more amicable relationship, but it might as the player becomes better acquainted with Melvin.
5. **Deceptive friendliness.** The other four behaviors can also be used in a deceptive manner. They will look similar to behaviors for actually making friends in many cases, but the real goal is something else. Sluggo does not actually use this kind of behavior, but this could have been used by him to get the player to offer him better deals.

The next two sections will focus on helping others and modifying other behaviors to be more friendly. In the helping others section, I will step through how to apply the methodology to the behavior, as I did with negotiation. In the modifying behaviors section, I will show how using the methodology to build social behaviors makes it possible to make friends even when engaging in other behaviors.

I will not discuss complimenting, general interactions or deceptive friendliness in this chapter. I built versions of the complimenting and general interaction behaviors, but they do not offer any opportunities to describe aspects of the methodology that are not better described elsewhere. Since deceptive friendliness was not implemented, I will not discuss it here. I did implement a simple form of deceptive helping others in *Office Politics* that was described briefly in Chapter 7.

9.3 Helping Others Achieve Their Goals

One way to become friends with another agent is to help them achieve their goals. In *The Playground*, one of the ways that Melvin tries to befriend the player is by helping the player get baseball cards that Melvin believes the player wants.

Figure 9-1 provides two traces of Melvin offering to help the player. In the first trace, Melvin hears the player ask Sluggo for a trade. When he hears this, he realizes that he has an opportunity to help the player. This makes him excited, which makes him smile. Unfortunately, he is also too scared of Sluggo to approach the player, so he also looks nervously at Sluggo. When the player comes over to see what Melvin is excited about, Melvin excitedly suggests a trade that will help the player achieve the goal of getting the Willie Mays card from Sluggo.

In the second trace, the player initiates the interaction by asking Melvin for help. Melvin offers assistance, as he does in the first trace, by offering to trade with the player and suggesting a good offer to propose to Sluggo.

Both of these traces have been edited for readability, but the content has not been changed. Also, recall that in these traces, the natural language skills of the characters are limited to keyword matching and template-based speech.

I will analyze how I used my methodology to create this offering help behavior. I will begin by analyzing how to incorporate various aspects of personality into the behavior. Then I will discuss how such a behavior can be made with minimal amounts of modeling of other agents. I will not discuss all of the ways that personality can be incorporated into this behavior as they do not help shed new light on the methodology. I will focus on competence, robustness, emotion, relationships and attitudes, and personality quirks. I will not discuss norms, roles, and interactions with other goals.

FIGURE 9-1 **Two Traces of Melvin Offering Help**

```
You go into the tree house.
Sluggo says to you, ``Wanna trade Catfish Hunter for Babe
Ruth?''.

PLAYER> Sluggo: How about Ruth for Mays?

Sluggo says to you, ``I'll swap you Jose Canseco for Babe
Ruth.  Deal?''.
Melvin is now smiling.
Melvin looks nervously at Sluggo.

PLAYER> get in the sandbox

Sluggo says to you, ``Hey, jerk, I was talking to you.''.
Melvin is speaking excitedly to you.
Melvin says ``Hey Vulcan, I'll trade you Mickey Mantle for
Babe Ruth, which might help if you're trying to get Willie
Mays from Sluggo!''.
```

```
You go into the sand box.
Melvin is now smiling.
Melvin says to you, ``Salutations, Vulcan ambassador!  The
Klingon high command has sent me in search of baseball
cards.''.

PLAYER> Melvin: Hi! Can you help me get a Willie Mays card?

Melvin says to you, ``So the Vulcan ambassador wishes to get
Willie Mays from Darth Sluggo.  He really likes Mickey Mantle
you know and I'd trade you.''.

PLAYER> Melvin: Thanks.  What do you want for Mantle?

Melvin says to you, ``The aliens told me to offer you Mickey
Mantle in return for Babe Ruth.''.
```

9.3.1 Competence

As with negotiation, I began by determining how competent to make the behavior to suit the personality of the particular character—in this case, Melvin. I decided what goals Melvin should be able to help the player achieve and how complicated his methods of assistance would be.

It would be possible to build a general behavior for Melvin to help many agents achieve many types of goals, but that isn't necessary for this world and characters. All I wanted Melvin to be able to do was to help other agents get baseball cards; building a more general behavior than that was unnecessary.

Here is a general description of Melvin's behavior that demonstrates the level of competence, but none of the other aspects of personality:

```
1. Learn that other agent wants a card
2. If self has card
     then if know of good full offer
        then make full offer
        else make single-card offer
     else if third-party has card
        then if know of full trade other can offer third-party
           then suggest trade to other
           else if know of two-step trade
              then suggest two-step trade
              else if initially asked for help
                 then apologize
```

In English: Melvin keeps track of cards that the other character (in this case, it is always the player) wants. For instance, if Melvin hears the player say to Sluggo something like, "What do you want for Mays?" or "Would you trade me Mays for Ruth?" Melvin infers that the player wants to get Sluggo's Willie Mays trading card. It can also be that the player asks Melvin directly for help, such as, "Melvin, how can I get Sluggo to trade me his Willie Mays card?"

If Melvin has a Willie Mays card, he may offer to trade it to the player, either by suggesting a full trade ("I'll give you Mays for Aaron.") or by letting the player lead the negotiation ("What'll you give me for Mays?"). If Melvin doesn't have a Mays card he can try to suggest a reasonable offer for the player to make to Sluggo ("Why don't you offer Sluggo Reggie Jackson for Mays?"). A final suggestion would be to offer to trade the player a card that the player could then use to get Mays from Sluggo ("I'll trade you Mantle for Aaron, which you could then offer Sluggo for Mays." or "Do you want to trade me for Mantle? You might be able to offer Mantle to Sluggo for Willie Mays.").

If none of these approaches are reasonable and Melvin was initially asked for help, then he will apologize for being unable to help.

9.3.2 Robustness

Robustness, again, is making sure the agent never (or rarely) does anything to break the user's suspension of disbelief. As before, I propose two approaches to this problem. They are being prepared and being ready for when the character isn't prepared.

Here are a few illustrative examples of how I introduced robustness into this behavior.

Being prepared

- If the user somehow gets the card that Melvin is trying to help him get, Melvin will know to stop the behavior or at least choose something else to help the user with. This is achieved by adding a context-condition to the helping behavior that notices that the behavior no longer makes sense in contexts where the other agent has already achieved its goal.
- Melvin is able to follow the user around when the user moves during Melvin's efforts to offer help. This is the result of a demon created by the helping behavior that fires whenever Melvin is in a different location than the agent that he is trying to help.

Being ready when not prepared

- When the player asks for some kind of help that Melvin is not programmed to understand (e.g., "Melvin, will you help me do my taxes?"), Melvin will try to avoid the question without being unbelievable. One way he does this is to yell, "Red alert! Red alert! The Romulans are in the neutral zone." and run off to the jungle gym. Hopefully, the player will be distracted by this new game or at least temporarily leave Melvin alone to his delusions.

  While this reaction would be inappropriate for most traditional AI agents, Melvin is not a traditional AI agent. Melvin is a believable agent and reactions like this are critical to maintaining believability.

9.3.3 Emotion

Emotions affect Melvin's helping others behavior and the behavior affects his emotions. I will look at each of these effects.

The Effect of Emotions on Helping Others

As I described in the previous chapter, there are some general ways emotions often affect social behaviors that can provide a useful starting point. There are effects on the agent's priorities, appraisals, willingness to interact, and style of action. Not all of these are useful for every behavior; in Melvin's helping others behavior I only used the latter two.

- **Willingness to interact.** The choice to help another agent is influenced by Melvin's emotions. If he is acting friendly towards an agent, he will offer to help that agent. As long as he does not have an active *bad-mood* behavioral feature, he will give help when asked. If he has a *bad-mood* feature present, he will not help anyone, even if asked.
- **Style of action.** Melvin's tone of voice will change based on his emotions. For instance, when the helping others goal gives rise to an *anticipation* behavioral feature, Melvin's offers to help will end in exclamation marks and be described as excited speech.

I have also claimed that besides these general classes of effects, most behaviors will have some specific ways to express emotions. I will discuss a few of these as well.

- **Other effects of emotion.**
  - Emotions can affect how much help Melvin will give to other agents. For instance, Melvin might be generally willing to help (this is the "willingness to interact" from above), but not so willing to help as to give up anything he values. If, however, Melvin has a *generous* behavioral feature directed towards the other agent, he is more willing to trade away a valued card to help the other agent.
  - One of Melvin's options for helping is to give the other agent a card they want or need in exchange for something that Melvin wants. This is a negotiation as described in the previous chapter; that behavior can be invoked by Melvin's making friends behavior. Negotiation has a number of ways of expressing emotions that were described in the discussion of negotiation. In this case, the helping others behavior is emotionally expressive because it invokes another behavior that is already emotionally expressive.

The Effect of Helping Others on Emotions

Here are a few of the ways that Melvin's emotional state is influenced by this behavior:

- The `meet-goals-for` goal, which leads to Melvin trying to help another agent achieve their goals, has an importance of success of 7. This means that when it is successful, Melvin's emotion generators will generate a reasonably intense *joy* structure.

- There is a function that computes the perceived likelihood of the `meet-goals-for` goal succeeding. The function is updated as progress is made during the interaction. As it goes up, appropriate *joy* and *hope* structures are generated.
- When Melvin successfully helps another agent achieve a goal, Melvin gives credit to himself. This leads to *pride* and *gratification*.
- Helping another agent can result in a full-fledged negotiation and there are a number of emotions that might arise during that interaction. These are discussed in the previous chapter.

9.3.4 Relationships and Attitudes

Although the reason I built this helping others behavior was to help Melvin form a new relationship, it is possible for other pairs of agents to use a helping behavior when they are already in a relationship (e.g., friends, parent-child). For instance, if Melvin were to become friends with the user, Melvin should still want to help the user. Because I did not actually integrate relationships into the offering help behavior, I will not discuss how to do so here.

Melvin does, however, have attitudes about the player that affect this behavior. There are a couple of ways that attitudes interact with the helping others behavior in Melvin. I suggested three general classes of effects attitudes and relationships often have social behaviors in Chapter 8; they are affects on the willingness to interact, the style of action, and the agent's appraisals. Only willingness to interact was used in Melvin. Also, I have incorporated one other effect that is more specific to this particular behavior.

- **Willingness to interact.** Melvin begins with a like attitude towards the player, which leads to a goal to make friends, which in turn leads to the behavior to help the player. If the liking attitude goes away, then Melvin will no longer initiate helping the player. If Melvin comes to dislike the player, he will not even help the user when asked directly.
- **Other ways to affect the behavior.** When deciding how much help to give, Melvin will base his decision partly on how much he likes the player.

9.3.5 Personality Quirks

I needed to find ways to make Melvin's quirky personality come through in this behavior. I used a few methods that I previously discussed in the negotiation section.

- **Choice of phrasing.** Melvin's trekkie side comes through in how he offers help to the other character. He says things like, "If you want to get Willie Mays from that Ferengi Sluggo, you should try offering him Mickey Mantle."
- **Willingness to help.** Melvin's willingness to help others and to make friends is part of his distinctive personality. In other words, the existence of this behavior in the first place is a matter of Melvin's personality. Sluggo does not have this behavior because it does not fit his personality.

This is a particularly good example of what I mean when I say that personality permeates behavior. Melvin's personality can affect every choice from the most general choice of whether the behavior should even exist in this agent to how individual speech-acts should be phrased. Any particular aspect of personality, like personality quirks or emotions, may not be expressed everywhere in the behavior, but my experience has been that almost every part of every quality believable behavior is influenced by at least some aspect of the character's personality.

This is one of the main reasons that I feel building behaviors with knobs to express personalities would be an ineffective approach to this problem. Each behavior has a large number of places that personality can be incorporated and the variety of ways of expressing personality at many of those places is very large. This makes the problem of pre-creating behaviors that can express a wide enough range of personalities to be suitable for artistic domains a very difficult one.

9.3.6 The Role of Representation

My methodology for building social behaviors calls for using minimal amounts of modeling of other agents. Helping others provides another example of how a little bit of representation can go a long way. This behavior requires some of the same representation schemes I used in negotiation as well as some different ones.

First, I reused the representations that were used in the general negotiation behavior to keep Melvin from repeating trades the user had previously rejected. This representation is a list of offers that each character has rejected. In helping others, this is used when Melvin helps by offering a trade that would help the user achieve his/her goal; the representation ensures that Melvin doesn't make offers to help that require re-offering a previously rejected trade. The helping behavior has its own behavior to offer a trade, since I wanted Melvin to phrase the offer differently if he was making the offer to be helpful, but the actual negotiation is done by the previously described negotiation behavior.

Second, I used a simple model of other characters' goals to get specific cards. This allows Melvin to store such information until he can act on it. Because Melvin only stores one kind of goal, he can store goals as a list of tokens (e.g., `(babe-ruth willie-mays)` would represent the desire to get a Babe Ruth and a Willie Mays trading card).[1] This model needs to be updated when other characters satisfy their goals. The update rule is a demon that watches for an agent to be holding a card that Melvin's model of the agent indicates that agent wants. The demon removes this goal from the model. This update rule is so straightforward because of the simple nature of these goals and the ability of the characters in this environment to see what the other characters have in their possession.

Finally, if I want Melvin to suggest other possible trades, he need some way of suggesting plausible trades to make for the desired card. For instance, if Melvin wants to suggest an offer that the player should make to Sluggo, Melvin needs to have a model of Sluggo's preferences in order to make useful suggestions. Melvin's model of Sluggo's card preferences is a list ordered by how much he believes Sluggo likes each card.

I have avoided the difficulties involved in keeping this model of Sluggo's appraisals up-to-date by giving Melvin an accurate model of Sluggo's appraisals so he will always give the player good advice. As my goal was for Melvin to give the player good advice, there was no need to add representations that were incorrect, especially as that would require adding code to correct those errors in certain circumstances. Melvin doesn't need to be perfectly general; he just needs to fit the artistic goals of a specific world. By taking the artistic goals into account, it is possible to limit the need for modeling other agents.

Melvin also doesn't need to model the player because he doesn't like Sluggo and, therefore, will never help Sluggo achieve a goal by suggesting trades with the player. If Melvin could help Sluggo by suggesting a trade with the player, extra behaviors would have to be written specifically to keep Melvin's model of the player's preferences up-to-date.

If the artist knows his/her characters and artistic goals well, I have found that there are often ways to limit the amounts of representation those characters need. There is no need to create general representations when simpler, more spe-

1. If more goal types were stored, I could add a bit more structure (e.g., `((make-friends sluggo)(get willie-mays))` could represent the desire to make friends with Sluggo as well as the goal to get a Willie Mays card).

cific representations will do, and there is also no need to create representations to support behaviors that the agent will never engage in.

9.4 Making Friends by Modifying Other Behaviors

A second way to make friends with another agent is to modify interactions with that agent to be more friendly. Following my methodology in building other behaviors can simplify this kind of interaction. This is because Em's behavioral feature system can be used to modify other behaviors in certain circumstances.

Behavioral features encode how an agent is supposed to be acting at any given moment. Recall, from Chapter 5, that behavioral features have a type (e.g., act aggressively), a direction (e.g., towards Sluggo), an intensity (e.g., 7 out of 10), and a cause (e.g., because he insulted me). The direction and cause are not present in all features. Behavioral features are often generated by the emotion system, but this is not always the case—it is also possible for other behaviors to create behavioral features for any number of reasons. Being able to communicate certain kinds of information between goals is one such reason.

Melvin's make-friends behavior creates a new feature to act friendly towards the player (i.e., type=friendly, direction=player, intensity=[importance of becoming friends], cause=make-friends). Since Melvin's other behaviors have been written to take features into account, this friendly feature will be expressed in all sorts of other behaviors. Because the intensity of the feature is set to the importance of the goal, the more Melvin wants to be friends, the friendlier he will act.

Melvin's other behaviors take features into account because they were built following my methodology, which tells artists to integrate emotions into their agent's social behaviors and, in Tok, this is done via the behavioral features. So, if the agent builder follows my methodology when building social behaviors, those behaviors can interact using this mechanism. For instance, when Melvin and the user are negotiating, Melvin will act friendly towards the player. (I discussed some ways to act friendly while negotiating, in section 8.3.6.)

As discussed in the previous chapter, this is a rather limited form of communication. Where it works, however, the artist is able to leverage off effort towards creating emotional behaviors to achieve inter-behavior communication.

9.5 Summary

In this chapter, I used two variations of Melvin's making friends behavior to shed additional light on how to apply my methodology to create believable social behaviors. Here are some of the important issues that have come up in this chapter.

- I looked at making friends, a behavior that is not traditionally studied in AI, with a number of possible instantiations: helping others achieve their goals, general interactions to make friends, complimenting other agents, and modifying other behaviors to be more friendly.
- I described how to apply my methodology for building social behaviors to Melvin's behavior for helping other agents achieve goals. I showed how to incorporate Melvin's personality into the behavior and how I was able to use my knowledge of Melvin and the playground world to constrain the amount of representation Melvin needs of other agents.
- I reviewed how the Behavioral Features mechanism I described in Chapter 5 can be used as an effective way to provide a simple form of communication between certain behaviors built following my methodology.

CHAPTER 10 Validation of the Methodology

The main contribution of this half of the thesis is a methodology for building believable social behaviors for individual characters. In this chapter I present a user study that suggests that social behaviors built using this methodology can be believable. The study I describe focuses on the negotiation and making friends behaviors described in Chapters 8 and 9 as I built them into the characters in *The Playground*.

10.1 Experimental Methodology

The experiment is the same one described in Chapter 6, with 17 users interacting with Melvin, Sluggo, and a variation of Melvin called Chuckie. The questionnaire asks not only about the emotions of the characters, as described in Chapter 6, but also about their social behaviors. In this case, the answers provided about Chuckie are ignored since he was included specifically to test the emotional aspects of the characters. The questionnaire given to the users is included in Appendix B. More details about the experimental methodology can be found in section 6.2.1.

10.2 Experimental Results

The goal of this experiment was to determine if the behaviors and characters that I built using my methodology for building social behavior are believable. To answer this question, I asked users about a number of facets of believability. The evaluation of the behaviors focuses on four questions: (1) Did the behaviors basically work? (2) Did the characters have distinctive personalities even when they were engaging in social behaviors? (3) How often did the characters break the users' suspension of disbelief? (4) Were the characters good characters?

By evaluating the answers to these questions, I show that it is possible to create behaviors with important elements of believability using my proposed methodology. In each of the following four subsections, I will analyze these questions in turn and the resulting answers I received from users.

10.2.1 Competence of Behavior

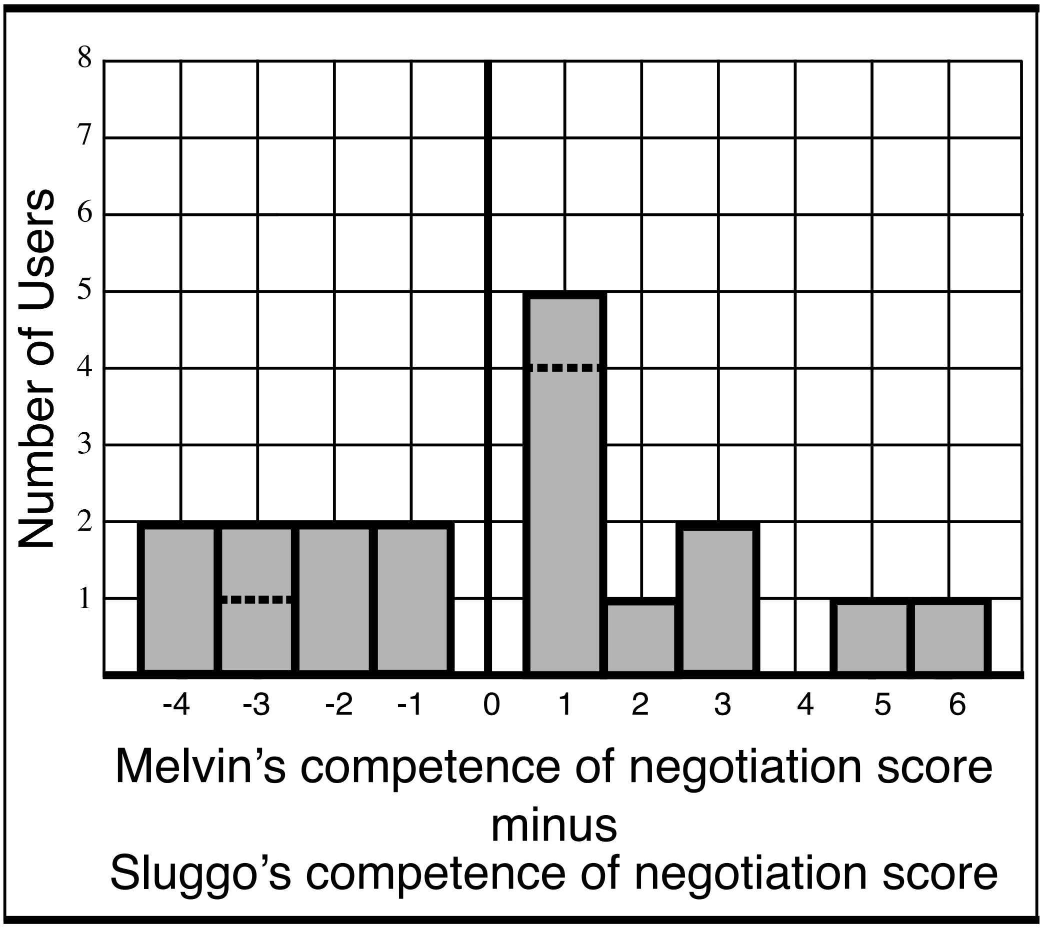
The initial question is whether the negotiation and making friends behaviors that I have described worked. In other words, if it turned out that Melvin and Sluggo were completely inept at negotiation or users never understood that Melvin wanted to make friends, then the behaviors would have been shown to be ineffective. Remember that the goal of this work, unlike work in traditional AI, is not to make highly competent behaviors, but to make behaviors that are appropriately competent and recognizable.

I asked the users to rate Melvin and Sluggo on a scale of 1 (poor negotiator) to 7 (great negotiator). Melvin's mean score was 3.82; Sluggo's was 3.41(3.65[1]). The first thing to notice about these numbers is that they are somewhere in the middle of the scale, which is what I wanted. Neither was considered a completely incompetent negotiator, but neither were they overly good.

When the differences in scores received by Melvin and Sluggo is computed, the mean difference is 0.41(0.18) and the standard deviation is 2.92(3.03). (Figure 10-1 shows the distribution of values in histogram form.) Although Melvin has a higher mean score in the population of users that I asked, the more important question is whether there is enough evidence to make a claim about the general population. That is, how likely is it that Melvin's score for negotiation competence is greater than Sluggo's score in the general population, given the data I gathered?

1. I allowed users to give a high and low value for answers if they felt that a character changed over the course of the interaction. This is the only case where a user gave a range answer to any of the questions of interest in the study. In this case, the user gave Sluggo a low value of 1 and a high value of 5, resulting in the two values given here. The first value of each pair mentioned in this section uses the 1 response. The second value uses the 5 response.

FIGURE 10-1 **Melvin vs. Sluggo: Competent negotiators?** This histogram shows the differences in how Melvin and Sluggo were scored in terms of negotiation competence. They were rated on a scale of 1 (poor negotiator) to 7 (great negotiator). For example, this chart shows that 2 of 17 users scored Melvin 3 points higher than Sluggo on the 1-7 scale. One user provided a high value for Sluggo of 5 and a low value of 1. Compared to the constant value of 2 given to Melvin, the difference shows up as either 1 or -3. The dashed lines at 1 and -3 represent these two options.



If I use a statistical t-test[1], I can only claim Melvin to be a better negotiator with confidence of <75%. This means that there is not enough data to support the hypothesis that the general population of users would think that Melvin is a better negotiator. This is somewhat disappointing since I had tried to make Melvin the better negotiator and it didn't come across very well. After completing the ques-

1. A t-test is a standard statistical tool for analyzing this sort of data. For a slightly more detailed description of what a t-test is and how it is used, see section 6.2.2.

tionnaire, I asked some users about their scores. Some users felt that Sluggo's direct, aggressive approach was very effective, while some felt that Melvin was a less effective negotiator because he was also trying to make friends with the user.

To test the competence of the making friends behavior, I asked, "Do you think X wanted to be your friend?" 16 users answered on a scale of 1 (not at all) to 7 (yes, definitely). Melvin mean score was 5.18 (standard deviation of 1.94); Sluggo's mean score was 1.88[1] (standard deviation of 1.45). The average user scored Melvin 3.30 point higher than Sluggo. Users clearly picked up on Melvin's desire to be friends more than Sluggo—using a t-test I can claim that Melvin's actual mean score is at least 2.15 greater than Sluggo's with 95% confidence. Figure 10-2 shows a histogram of the user's answers to this question.

In addition, I asked users to answer the following question only if they thought that the character had tried to make friends with them: "How good a job did X do at winning you over to be his friend?" 15 of the 17 users (88.2%) answered the question for Melvin, indicating that they thought that Melvin was trying to make friends. Only 5 of 16 (31.3%) users felt that Sluggo was trying to make friends. I also have reason to believe that some of those responding didn't notice that they were supposed to answer only if they felt that Sluggo was trying to become friends, as 2 of the 5 gave him the lowest possible score on the 1-7 scale. (1 of the 15 users gave Melvin a score of 1).

Of those answering this question, the mean score Melvin achieved was 4.20 (standard deviation of 1.66) to Sluggo's mean of 2.20 (standard deviation of 1.64). This indicates that those who thought that Melvin and Sluggo were trying to make friends with them also seem to have thought that Melvin did a better job at it. Because of the small number of answers for Sluggo, statistical comparisons are inconclusive. Figure 10-3 provides a histogram of the scores users gave the two characters on this question.

An important aspect of believability is being able to create characters with the appropriate level of competence at various behaviors. The experimental data suggests that this is possible to do following the proposed methodology.

1. Some of the data for Sluggo contains only 16 data points as one of the users skipped one sheet of questions related to Sluggo. The questions about making friends, overall clarity of personality, overall quality of character, and disrupting disbelief were all skipped by this user. When comparing Sluggo to Melvin in these cases, I eliminated the scores this user gave for Melvin.

FIGURE 10-2 **Melvin and Sluggo: Did they want to make friends?** When I asked users to rate Melvin and Sluggo in terms of how much they thought the characters wanted to make friends with them on a scale of 1 (not at all) to 7 (yes, definitely), I received the following distribution of responses. 17 users responded for Melvin; 16 responded for Sluggo.

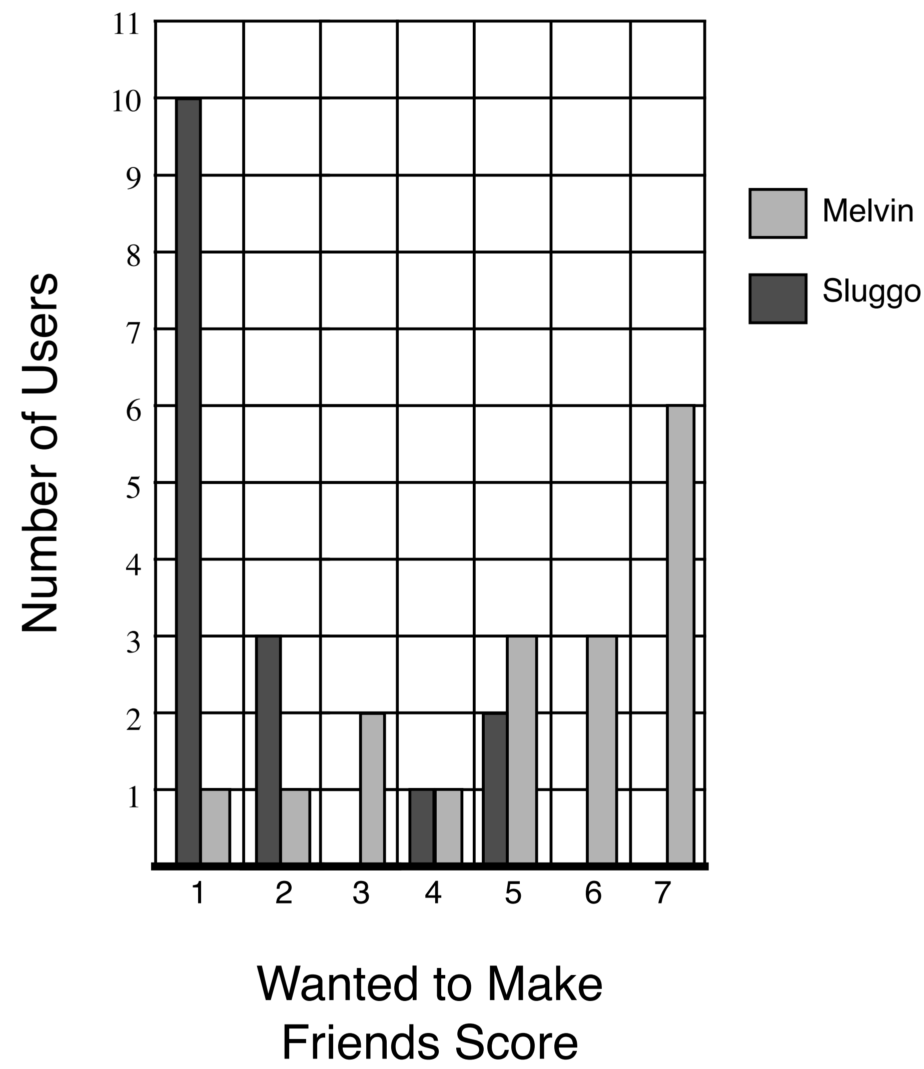
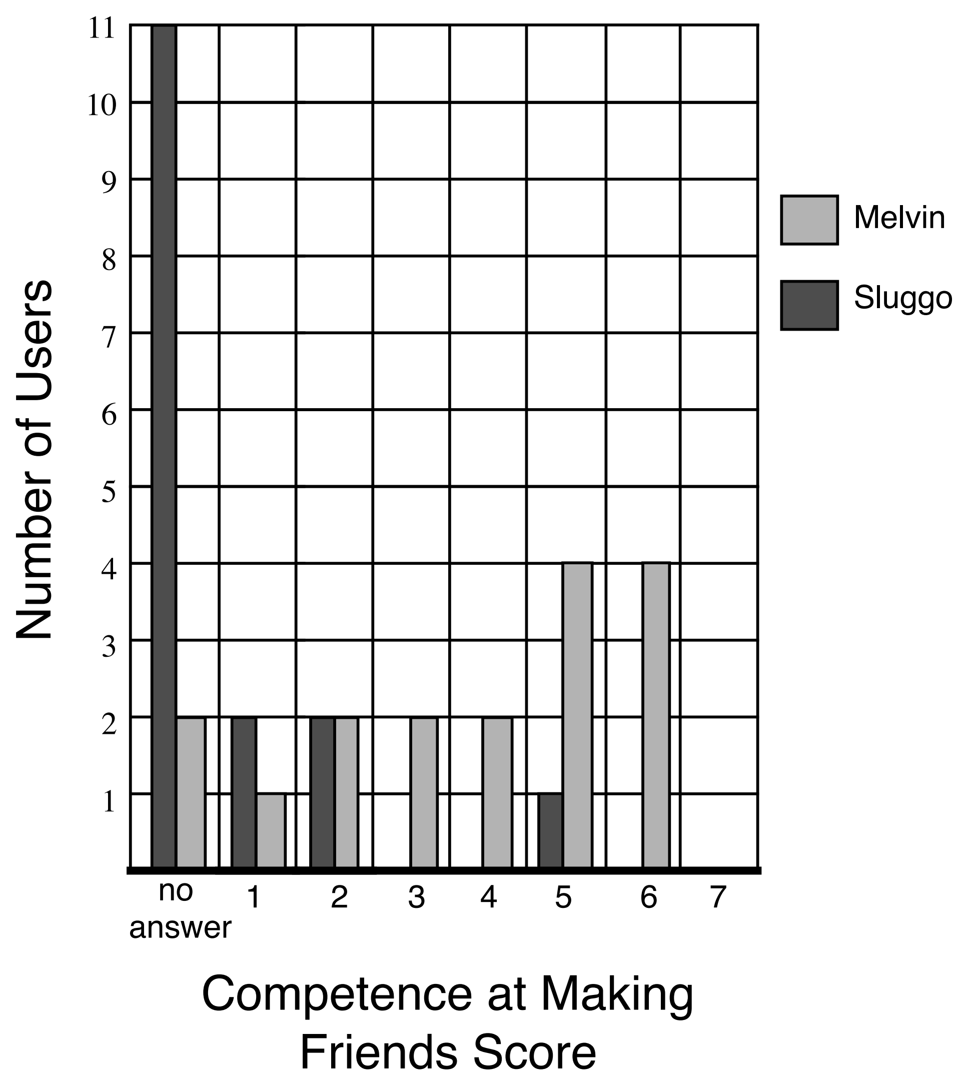


FIGURE 10-3 **Melvin vs. Sluggo: How good at making friends?** This histogram shows how Melvin and Sluggo were scored in terms of how good a job they did at making friends with the user. They were rated on a scale of 1 (terrible job) to 7 (great job). The total number of users was 17 for Melvin and 16 for Sluggo. Users were told not to answer this question if they thought the character did not try to become friends with them at all. There were 2 no-answers for Melvin and 11 no-answers for Sluggo. For example, this chart shows that 1 user gave Sluggo a score of 5 compared to 4 users for Melvin.



10.2.2 Personality

The second question is whether the characters have distinctive personalities and whether these personalities come across during their social interactions. The focus of the first part of the methodology is to create social behaviors that are rich in personality. This question addresses success in this area.

I asked users to answer the question, “Did X have a clearly defined personality?” on a scale of 1 (not at all) to 7 (very much). Melvin scored a 5.59 (standard deviation of 1.18) and Sluggo a 5.81 (standard deviation of 1.17). Using a t-test, I can claim that the true mean is greater than 5.00 with confidence >95% (Melvin) and >99% (Sluggo). These numbers suggest that users feel that Melvin and Sluggo have clearly defined personalities; the main question, though, is if the personalities come across through their social behaviors. To determine is this is the case, I compared this general personality data with data I gathered about the personalities of the characters while they were engaging in particular social behaviors.

First, I asked the users if they though that the characters’ personalities came across during social interactions. First, I asked, “Did X’s personality come across even when he was negotiating with you?” On a scale of 1 (not at all) to 7 (very much), Melvin scored a 5.82 (standard deviation of 0.95) and Sluggo scored a 5.81 (standard deviation of 1.42).

I also asked users to rate how well Melvin’s and Sluggo’s personalities came across when they were trying to make friends. Specifically, I asked, “Did X’s personality come across even when he was trying to become friends with you?” If the user felt that the character didn’t try to make friends, they were told not to answer the question. 15 of 17 users answered for Melvin, who received a mean score of 6.0 (standard deviation of 1.0). Only 5 of the 16 users answered for Sluggo, which is not a statistically significant sample. This low response rate is expected since Sluggo did not try to make friends with the user.

The data for each of these questions is presented in histogram form in Figures 10-4 through 10-6.

Informally, this data suggests that both characters have reasonably clearly defined personalities. Also, the personality scores for the specific social behaviors are all at least as high as the general clarity of personality scores, which indicates that the characters were able to express their personalities when they were engaging in social behaviors.

More formally, I computed the differences in scores between the specific behaviors and the general personality. When I compare Sluggo’s personality score while negotiating to his general personality score, I find that users score his personality while negotiating 0.31 higher than his overall score (mean=0.31, standard deviation=1.34). Using a t-test, I can show that there is a less than 20% chance of Sluggo's negotiation personality score being lower than his general personality score, but the probability of it being more than 0.5 points lower is less than 2%.

Doing a similar comparison, I find that Melvin's mean personality score when negotiating is 0.24 higher than his overall personality score (standard deviation=0.81). Using a t-test, I can show that there is a less than 15% chance of his behavior-specific score being below the general personality score but a less than 1% probability of being more than 0.5 points below general personality score.

Finally, I find that Melvin's mean personality score when making friends is 0.33 higher than his overall personality score (standard deviation=1.14). Using a t-test, I can show that there is a less than 15% chance of his behavior-specific score being below the general personality score but a less than 1% probability of being more than 0.5 points below the general personality score.

Artists claim that having clear, distinctive personalities is crucial for creating believable characters. The first part of the methodology that I have proposed is concerned with how to create personality-rich social behaviors. The experimental evidence suggests that I have been able to use the methodology to create characters that are able to maintain distinctive personalities even when they are engaging in social behaviors.

FIGURE 10-4 **Melvin and Sluggo: Clear Personalities (overall)?** When I asked users to rate Melvin and Sluggo in terms of how clear their personalities were overall on a scale of 1 (not at all) to 7 (very much), I received the following distribution of responses. 17 users responded for Melvin; 16 responded for Sluggo.

FIGURE 10-5 **Melvin and Sluggo: Clear Personalities (when negotiating)?** When I asked users to rate Melvin and Sluggo in terms of how clear their personalities were when they were negotiating on a scale of 1 (not at all) to 7 (very much), I received the following distribution of responses. 17 users responded for Melvin; 16 responded for Sluggo.

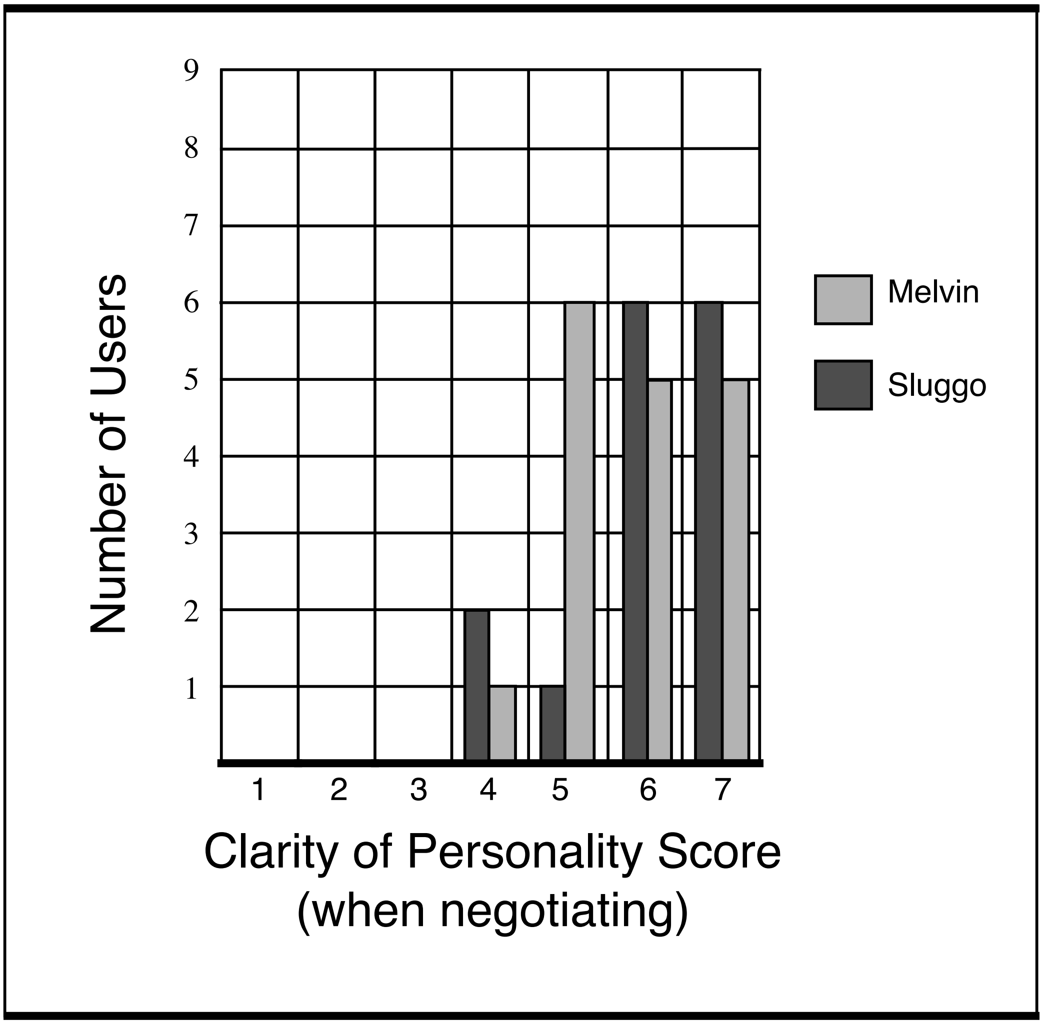
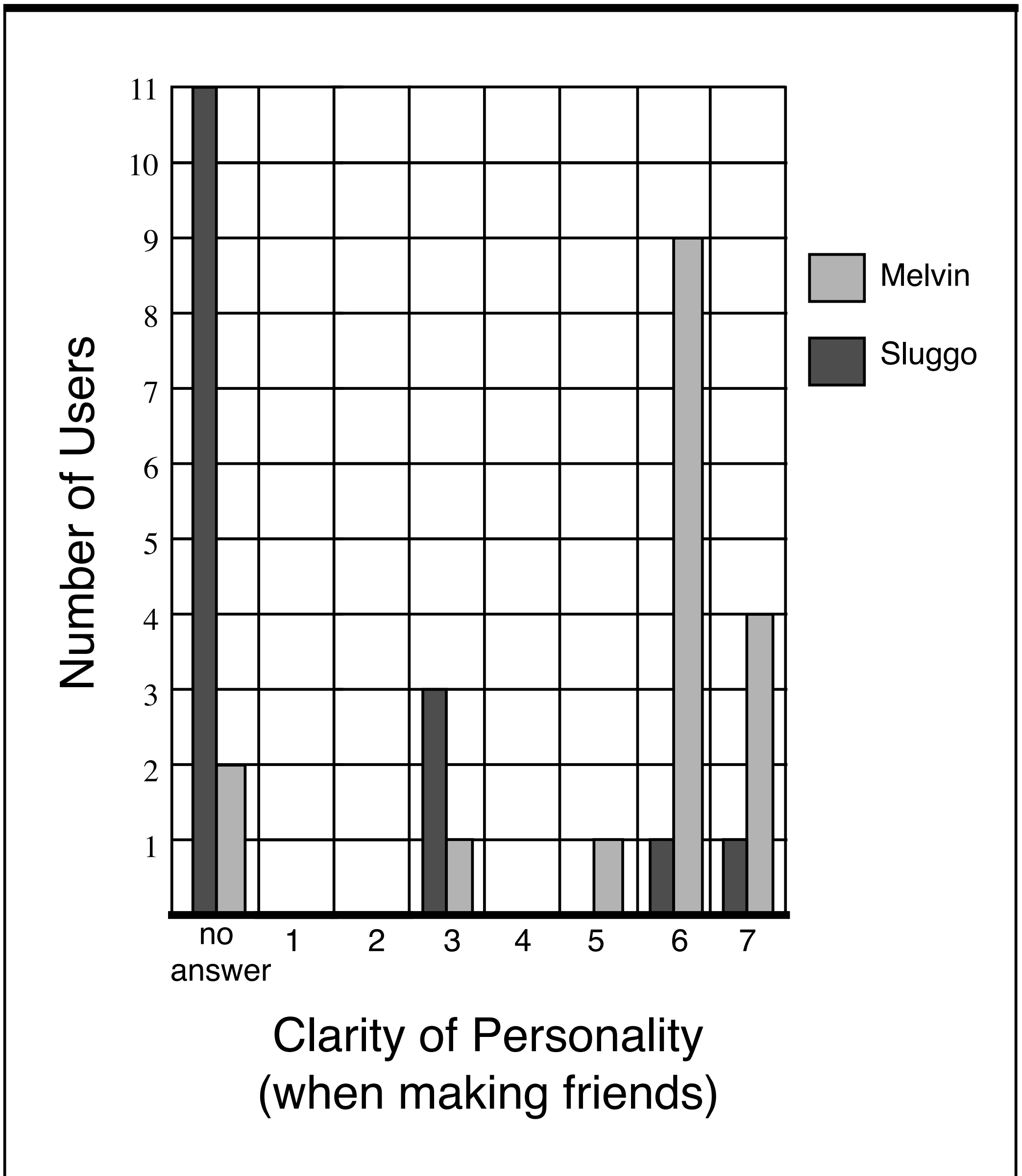


FIGURE 10-6 **Melvin vs. Sluggo: Clear Personalities (when making friends)?** When I asked users to rate Melvin and Sluggo in terms of how clear their personalities were when they were trying to make friends on a scale of 1 (not at all) to 7 (very much), I received the following distribution of responses. I asked users only to reply if they felt that the character was trying to make friends with them at some point.



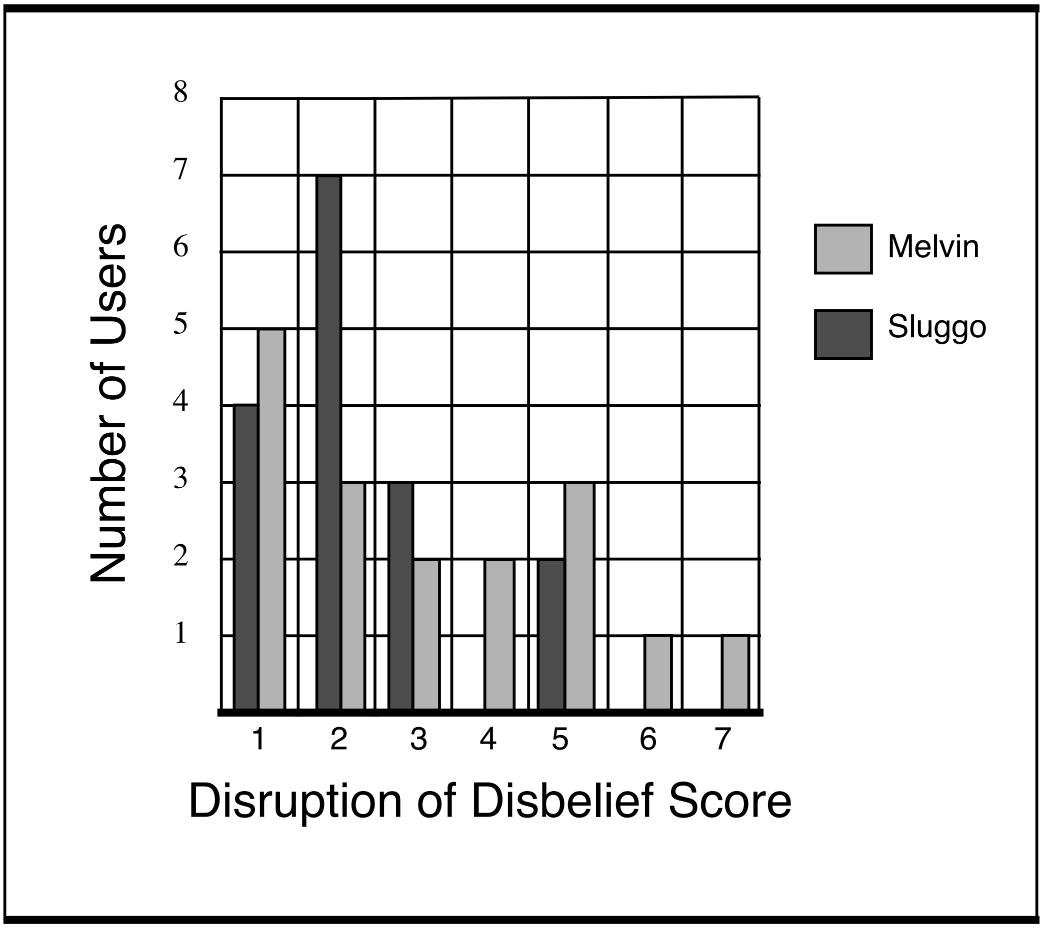
10.2.3 Robustness

The third question is whether the characters were able to stay in character throughout the interaction. My goal was to make the characters robust enough that they would only rarely do anything to break the user's suspension of disbelief.

Users were asked, "Did X ever do anything to disrupt your 'suspension of disbelief'?" With 17 users answering, Melvin's mean score was 3.12 (standard deviation 1.96). Sluggo, with 16 users responding, received a mean score of 2.31 (standard deviation of 1.25).

Figure 10-7 shows the distribution of scores. Most of the scores are bunched in the lower end of the graph, which is good. However, there were a few users who had their disbelief broken quite a bit, so even though the scores are promising on average and for most users, there are still times when the characters fail in ways that break the user's suspension of disbelief badly. I will discuss some of the reasons for this and some ideas about how to improve it in Chapter 11.

FIGURE 10-7 **Melvin and Sluggo: Disrupting disbelief?** When I asked users to rate Melvin and Sluggo in terms of how often they disrupted the user's suspension of disbelief on a scale of 1 (never) to 7 (all the time), I received the following distribution of responses. 17 users responded for Melvin; 16 responded for Sluggo.



10.2.4 Character Quality

The final question to answer is whether the characters are good characters. I asked users, “How good a character was X?” (1=awful character; 7=great character).

Melvin’s mean score for this question was 4.94 (standard deviation of 1.39); Sluggo’s was 5.31 (standard deviation of 1.25). Figure 10-8 shows the data in histogram form.

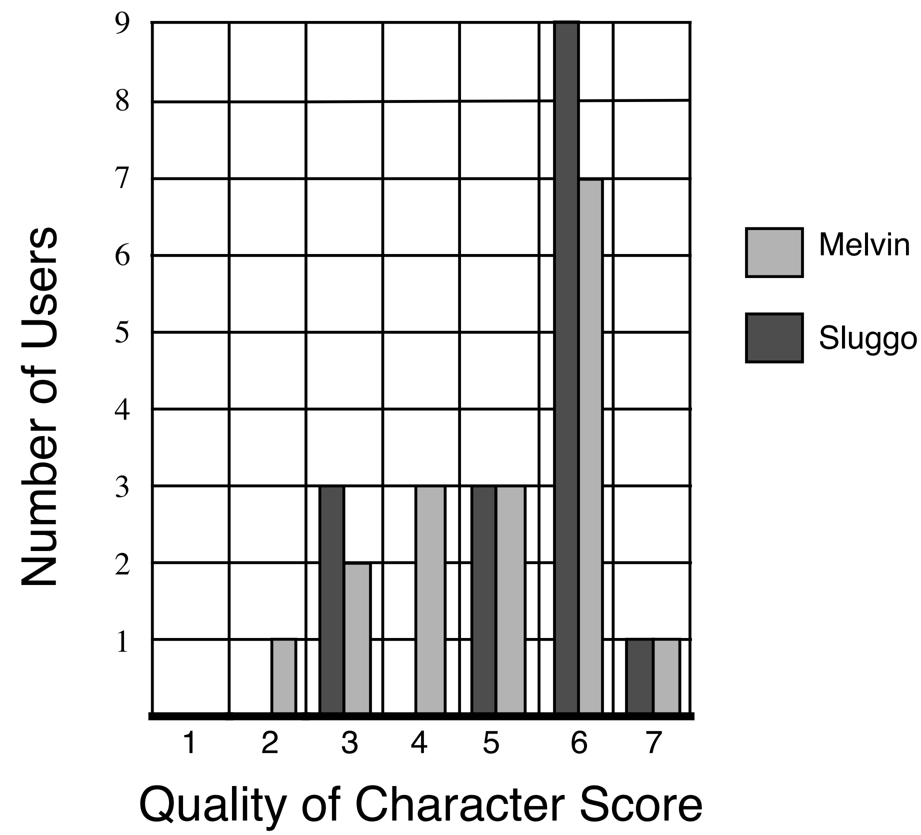
62.5% of all users gave Sluggo a score of 6 or 7 and 81.3% gave him a score 5 or above (4 is the middle of the 7-point scale). For Melvin, 47.1% of all scores were 6 or 7 and 64.7% of the scores were 5 or above. No users gave Sluggo a score of 1 or 2 and only 1 user gave Melvin a score of 2.

Also, using a t-test, I can claim with 95% confidence that the mean quality of character score for Sluggo is greater than 4.76 while the mean score for Melvin is somewhere above 4.35. This indicates that it is statistically highly probable that the scores for both Melvin and Sluggo are somewhat above the middle of the scale 7-point scale.

Having to compare the two characters to the mid-point of the scale is informative, but somewhat disappointing. It would be preferable to compare them to something other than the scale itself. As there is no other work directly related to creating believable social behaviors, I have been unable to compare the outcome of my methodology with other approaches. When other approaches to building believable social agents are created, I will be able to more fully evaluate this work.

The overall quality of the character is an important element of the believability of the character. I have shown that there is some evidence that it is possible to create quality characters using the methodology I have proposed for building believable social behaviors.

FIGURE 10-8 **Melvin and Sluggo: Good Characters?** When I asked users to rate Melvin and Sluggo in terms of how good they were as characters on a scale of 1 (awful character) to 7 (great character), I received the following distribution of responses. 17 users responded for Melvin; 16 responded for Sluggo.



10.3 Summary

The ability of the artist to control the competence of a character at social behaviors, to create characters that have distinctive personalities when engaging in social behaviors, to create characters that can engage in social behaviors without disrupting the user's suspension of disbelief, and to create good quality social characters are all important elements of being able to create believable social agents.

- I showed the negotiation and making friends behaviors described in Chapters 8 and 9 are appropriately competent. The behaviors are not incompetent nor overly competent, which is what I was aiming for. I found that judging competence can be difficult because users often consider other aspects of personality as well when judging competence. For instance, Sluggo's aggressive personality led some users to judge him to be a better negotiator than his actual skills supported.

- I showed that the characters have clearly defined personalities and that those personalities came across even when the characters are engaging in negotiation and making friends.

- I showed that the characters are reasonably robust, but that there are a few cases where they still cause users to break their suspension of disbelief. I will analyze these cases in the next chapter.

- I showed that the characters are good characters. The measurement of this feature, though, is difficult because there is nothing to compare these characters to yet.

- I showed that I was able to create good quality characters that negotiate and made friends reasonably competently while only occasionally disrupting the user's suspension of disbelief. I accomplished this using the methodology and techniques that were described in the second half of this thesis—I focused on a number of core aspects of personality when building the characters' behaviors and I relied on only simple representations of the other agents in the environment.

Part III:

Summary & Future Work

CHAPTER 11 Summary & Future Directions

I close with an overview of the main contributions of the thesis and a brief discussion about some of the possible future directions for this work.

11.1 Summary: Believable Emotional Agents

Part I of the thesis dealt with how to create believable emotional agents. In this section, I briefly summarize the contributions I made in this area and discuss some of the features of my approach. I also point out some of the limitations of my work.

11.1.1 Summary of Contributions & Discussion

An old adage goes something like, "If all you have is a hammer, everything looks like a nail." Well, the AI toolbox has more than a hammer in it, but most of the tools are of one of two types. They are either designed to create intelligent behavior or they are designed to model human cognition. In order to create artistic characters with emotions, neither of these tools is the right one to use.

Tools for intelligence are not appropriate because believable agents are not necessarily intelligent. There might be intelligent characters, but even they will rarely be perfectly rational; there are going to be many far-from-rational agents as well. Architectures designed to create intelligence are the wrong tools for this task.

Tools for creating cognitive plausibility aren't right either since believable agents are not necessarily cognitively plausible. As I have discussed before, artists in other media often create non-realistic characters because they are more effective as characters than more realistic characters would be; I expect the same to be true in the new medium. Furthermore, some artists will want to create non-human characters, like animals and space aliens. So, the tools I provide artists need to support the creation of agents besides those that fit into specific models of how humans work.

By studying the artistic nature of the problem of building believable emotional agents, I was able to create a new kind of tool, that I call Em. Em does not force either rationality or realism on characters, and was designed to provide artists freedom to create a wide range of characters. Many of the design decisions made for Em, like separating the emotion architecture from the emotion system and providing a behavioral feature system, were meant to provide artists with control over the emotional makeup of their characters.

The Em architecture is designed to provides a great deal of flexibility in the kinds of emotional characters that can be created. The default Em system is designed to help artists get started by providing reasonable default emotion processing.

The architecture supplies a set of inputs that the emotion generators can use for generating emotions. These inputs include information from the agent's goal processing system, perception system, physical body simulation, sensory memory, social system, and the emotion system.

The emotion generators are written in the Hap language, which proves to be suitable for this task because it provides demons, a flexible match language, the ability to perform internal (i.e., mental) actions, flexible control of actions, and goal priorities. The default set of generators is based on the cognitive emotion model of Ortony, Clore, and Collins [Ortony88]. I have extended, modified, and simplified their model of emotion generation in a number of ways to make it more suitable to the artistic nature of the task.

The emotion structures generated by the emotion generators are stored in a flexible hierarchy that is organized by the effects of emotion structures. The emotions decay over time in artist-specified ways. I provide default emotion decay function and emotion combination functions. The latter are useful when answering questions like, "Given these three distress emotion structures, what is the agent's overall distress intensity?"

The Hap language is again used to translate the set of emotion structures into a set of behavioral features. The mapping and the set of features used are determined by the artist. I provide a default mapping and set of features based on artistic and practical considerations.

Finally, the features are expressed by the agent in a number of ways. This is a difficult problem because artists tell us that emotions need to permeate the behavior of the agent and need to be expressed differently by each character. Where possible, I provide artists with default ways of expressing emotions. For example, I provide default mechanisms for changing an agent's body state, attitudes towards other agents, and goal priorities. For some forms of expression, particularly behavior, default expressions are a less practical approach. In these cases, I provide a set of mechanisms for expressing behavioral features and I leave the character-specific decisions about which mechanisms to use and how to use them to the artist.

Much of the power of the Em system comes from its placement in a broad agent architecture, called Tok. By using the other parts of the architecture, Em is able to create emotion structures for many different causes and to express those structures through a wide variety of means. Furthermore, by using other areas of the Tok architecture, like motivation and perception, I was able to create models of how emotions are generated that are often simpler and easier to use than models that rely on purely cognitive explanations as causes for emotions.

I built eleven characters using the Em tools (the seven described in the thesis as well as Lyotard the cat and the three Woggles). By running user studies, I was able to show that Em can be used to create agents that appear to be emotional. The emotions also improve the believability of the character.

11.1.2 Limitations of the Approach

There are (at least) two limitations to the approach that I have proposed for building believable emotional agents.

First, my character-specific approach to emotional expression means that there is a lot of work for the artist to do in order to create believable emotional expression. Every behavior that an artist builds needs to incorporate the agent's behavioral features if that behavior is going to be used to express the agent's emotions. Since I have argued (based on the arts) that the emotions should be expressed through many aspects of the agent, including in how the agent acts, most behaviors will need to be written to express emotions. Furthermore, I have provided only a limited number of general tools and default behaviors to simplify this process.

I believe, however, that this difficulty is as much a product of the domain as it is a product of my approach. Because artists claim that emotions need to permeate the personality of the character and be expressed in idiosyncratic methods, it is difficult to create useful default ways to express emotions. I have done just this for expressing emotions through facial expressions and a number of other simple means, but the task is much more difficult when expressing emotions through behavior. Even the facial expression default I have created will probably be changed for most characters.

The second difficulty is related to the first. Not only do all (or, at least, most) of the agent's behaviors need to take the behavioral features into account, but the number of behavioral features to incorporate into each behavior is reasonably large. This makes the task even more difficult. I tried to address this difficulty in the thesis by suggesting abstract features, like *good-mood*, that can ease this problem somewhat. While techniques like this can help, I don't know that the problem can be made simple. I expect that, just like writing a novel or a play, this process is inherently difficult and requires a tremendous attention to detail.

11.2 Summary: Believable Social Agents

Part II of the thesis dealt with how to create believable social agents. In this section, I briefly summarize the contributions I have made in this area and discuss some of the features of my approach. I also point out some of the limitations of my work.

11.2.1 Summary of Contributions & Discussion

As with the problem of creating believable emotional agents, the problem of creating believable social agents must be solved with the right set of tools. I believe that traditional AI approaches that stress modularity of mind are inappropriate for a task where the personality of the character is critical and permeates the entire character and the number of different types of social interactions is large.

In order to create characters with personality, artists need to have control over almost every aspect of every social behavior. They need to be able to create negotiation behaviors for characters from Hamlet to Bugs Bunny. A negotiation module that allows artists to tweak parameters is unlikely to provide the extreme sorts of flexibility that are necessary to create such distinctive variations.

Furthermore, an approach where the creation of characters is a matter of picking from a set of pre-coded social behaviors and tweaking them appropriately assumes that all of the possible social behaviors an agent needs to engage in are created ahead of time. The set of social behaviors is very large and is not constrained to “useful” behaviors like negotiation and cooperative plan execution. Artists need to be able to create behaviors like making friends and holding up stores.

By creating a methodology that has artists create social behaviors specifically for their particular characters, I have avoided both of these problems. Artists have the extreme flexibility that they need to create personality-rich characters and those behaviors are not limited to a pre-defined set.

The methodology that I have proposed has two parts. The first part is concerned with creating social behaviors that display the agent’s personality. The suggested approach is to build in the following aspects of the character’s personality: competence, robustness, emotion, attitudes and relationships, roles, norms, quirks, and the influence of other goals. For instance, a negotiation behavior should be robust enough to handle situations where the other agent walks off in the middle of the interaction—and the way this situation is handled should reflect the personality of the character in question. In this thesis, I have provided examples, technical solutions, and general suggestions about how to incorporate these various aspects of personality into believable social behaviors.

The second part of the methodology concerns how an agent should model other agents in the world. Two common approaches that AI uses for representation are to use rich and powerful representation with accompanying automated reasoning (the Distributed AI (DAI) approach) or to reject representation altogether (the behavior-based AI approach).

Once again, I found the AI toolbox lacking. I find the behavior-based approach to representation compelling because it has been shown to be effective for robustly controlling robots in complex, dynamic domains, whereas the DAI approach is typically better suited to more controlled environments. Since the worlds these agents inhabit can be fairly complex and unpredictable, the behavior-based approach seems like a natural choice. Unfortunately, the behavior-based approach, which relies on sensing instead of representation, is well-suited for robust, physical action but is less-well suited for social behaviors. Mataric [Mataric92] has shown that some social behaviors, like flocking, can be built without representation, but I found it impossible to create some social behaviors, like variations of negotiation, without any representation.

The methodology I propose is to use minimal amounts of representation of other agents. This means that when artists can create social behaviors without representation, they should, but when they find they are unable to create a behavior without representation, they should add just enough to allow them to build the behavior. This maintains many of the benefits of the behavior-based approach, like robustness and speed, without sacrificing the ability to create social behaviors that are very difficult (if not impossible) to create otherwise.

I built seven characters and nine different social behaviors using this methodology. By running user studies, I have shown that this methodology can be used to create social behaviors that are appropriately competent, that show off the personality of the character, that do not often break the user's suspension of disbelief, and that are quality characters. In section 11.3.1, I will address some of the problems that users had maintaining their suspension of disbelief as they provide some useful insights into future problems to be addressed.

11.2.2 Limitations of the Approach

The social behaviors that artists build using my methodology are not modular; they are character-specific and most of the code doesn't transfer to other characters with similar behaviors. This is an unfortunate side effect of my approach, but I have argued above that more modular approaches will not be able to create personality-rich, believable agents that can engage in a wide variety of social behaviors. If there are ways to bridge the gap between art and practicality, I think that would be a giant step forward for this area. In the meantime, artists have to work hard to create each character. In section 11.3.3, I will speculate about some approaches that might prove fruitful for simplifying this task, though I have no suggestions about how to make it easy.

A second, related drawback of this approach is that the approach to modeling other agents is also not modular. It would be simpler if an artist could plug in a powerful knowledge maintenance system instead of having to craft the representations on an agent-by-agent basis. The reason that I have not used a general, automated system is two-fold. First, as described above, such systems have not been shown to be as robust in complex, dynamic worlds as systems that do not rely on representations, in part because they tend to be slower and fail to maintain correct models. Many of the characters that artists create will need to respond in real time and pauses for thinking can break the user's suspension of disbelief. Second, I have not used a more powerful system because I do not want to limit the artists. One of the most important goals of this work is to allow artists to create varied and, sometimes, irrational agents. To build in a standard automated reasoning system runs the risk of taking this control out of the hands of the artists. By letting artists build the kinds of reasoning they want for their characters, they retain that control.

Fortunately, however, this problem is not as bad as it seems; forcing artists to create representations for each new agent is unfortunate, but creating those representations turns out to be a much simpler problem than it might at first appear. In the behaviors that I have created, which include, among others, negotiation, helping others achieve their goals, and holding up a convenience store, I have not had to resort to complex representations of other agents. I do not want to claim that they will never be necessary, but by taking the personality of the character and the specific world into account when designing the representation, I have found that simple, specific representations often suffice.

11.3 Future Directions

In this section, I discuss five main directions for future work. First, while running user tests, I found that there were a number of cases where the characters disrupted the user's suspension of disbelief. I briefly overview some of the reasons that the users gave me for the disruption, as these suggest areas that need to be addressed to create agents that are more believable. Second, in Chapter 6, I described a study I ran with three subjects to provide ideas about extending Em. As an addition to the ideas discussed in Chapter 6, I present here a number of my own ideas based on suggestions from the emotion-research literature. Third, I have found that the non-modular approach that I have taken to building believable social behaviors is controversial. I speculate about some ways to bridge the gap between my methodology and more modular approaches without losing artistic control. Fourth, I present some ideas for extending the Tok agent architecture to be even broader than it currently is. Finally, I suggest a number of other application areas where the work described in this thesis might be applicable.

11.3.1 Why Characters Fail

While running the user study described in Chapters 6 and 10, I found that the characters I built were reasonably good, but that in a few cases they broke the user's suspension of disbelief rather badly. I asked the users about the problems they had suspending their disbelief and got a number of interesting answers that might help direct future work towards making characters that are more believable.

- **Speed.** One of the most common complaints was that it was hard to become involved in the system when the response time is so slow. Each character only takes about 1-2 seconds to respond on average, but with two characters and the simulation processing that needs to be done, this can seem slow. Also, when the interactions get more interesting, such as during complex negotiations or fights, the agent's tend to have more processing to do, which can make them even slower. To top all this off, the system is written in Lisp and garbage collects a few times during any given twenty-minute interaction, which makes the system even less responsive. Past experience as well as evidence from Horswill's robots [Horswill94] indicates that speed of response is an important element in creating believability, so increasing the speed of this system should make it more effective. (A note: the Woggle system is written in C and is fast enough to produce real-time animation, so more speed even with the richer personalities discussed here seems an achievable goal.)

- **Natural language understanding.** Another problem that users pointed to was the inability of the characters to understand many simple sentences and questions. Although I did try to account for a good number of courses of conversation and seem to have done a reasonably good job in getting the characters to understand language related to trading baseball cards, the NLU system is still a simple keyword matcher with many limitations. Here's an example from user #15:

```
PLAYER> Sluggo: want to trade willie mays with ruth?

You are speaking to Sluggo.
Player's voice says ``want to trade willie mays with
ruth?''.
Sluggo is now red.
Sluggo is now scowling.
Sluggo is now tense.
Sluggo is speaking to you.
Sluggo's voice says ``I don't like making fancy trades,
dork!''.
```

In this case, Sluggo has misunderstood the offer of Willie Mays for Babe Ruth because the user used the word "with" instead of "for." Sluggo parsed this to be a multi-card, partial offer (like "What'll you give me for Ruth and Mays?") which he is not smart enough to handle, so he responds aggressively. In this case, however, his response is hard for the user to understand because the offer wasn't a very "fancy" offer.

- **Repeated speech.** One problem that almost always breaks believability is when a character repeats itself. Because there is no natural language generation system in these characters, I rely on canned text and simple templates. Although I have created a number of variations for each speech act, they will sometimes repeat. Here's an example from user #5:

```
You are speaking to Melvin.
Player's voice says ``I like Ted Williams better than Reggie
Jackson, though.  Who's your favorite baseball player? Mine
is Hank Aaron.''.
Melvin is speaking to you.
Melvin's voice says ``Sorry.  The aliens momentarily took
over my brain.  What trade do you want to make again?''.

PLAYER> Melvin: Who's your favorite player? Mine is Hank
Aaron. He's the coolest.
```

```
You are speaking to Melvin.
Player's voice says ``Who's your favorite player? Mine is
Hank Aaron. He's the coolest.''.
Melvin is speaking to you.
Melvin's voice says ``The Klingon Emperor wants to know if
you would be willing to part with Hank Aaron for Reggie Jack-
son?''.

PLAYER> Melvin: No way. Hank Aaron is way cooler than Reggie
Jackson.

You are speaking to Melvin.
Player's voice says ``No way. Hank Aaron is way cooler than
Reggie Jackson.''.
Melvin is speaking to you.
Melvin's voice says ``Sorry.  The aliens momentarily took
over my brain.  What trade do you want to make again?''.
```

In this example, Melvin has a canned response that he uses when the player seems to have made a strange trade ("Sorry. The aliens momentarily took over my brain. What trade do you want to make again?"). In this case, because Melvin is looking for player names as indications of trade offers, he is confused when the player mentions multiple players in a way that does not constitute a legitimate offer. The player does this again with different names in the third turn of the example and Melvin is still confused. The canned response is effective once—it is terrible twice. Some work in this area is already being pursued by Loyall [Loyall96].

- **Social Issues.** The only instance where a user mentioned social factors contributing to a break in the user's suspension of disbelief happened with user #2. This user started the interaction by hitting Melvin. Melvin immediately responded by running away and sulking. The user then left Melvin alone and traded with Sluggo for a while. When the player returned to the sandbox where Melvin was, Melvin initiated a trade.

  In this case, Melvin's emotional response had worn off and his attitudes about the player had changed to be less positive. This led Melvin to be less enthusiastic about trading with the player—he didn't smile and he didn't engage in any of his Star Trek banter. However, trading at all seemed unreasonable to the user. Melvin's reaction should have been to avoid all interactions with the player after being hit.

  I had decided that the player being mean to Melvin should cause Melvin to like the player less, but not dislike or fear the player enough to rule out all future interactions. This meant that Melvin was able to be more or less friendly depending on the course the interaction. I now realize that I should have differentiated being very mean from ways the player could be less mean (like insulting Melvin) and made the social interactions change more dramatically in response to the more extreme kinds of actions on the part of the player. This is a matter of increasing the importances of goals like not being hit; it could be done in a matter of minutes at most.

- **Emotion Issues.** There were two cases where users claimed that emotions were involved in breaking believability. The first turned out to be related to emotions in appearance only; the second was a legitimate emotion problem.

  The first case was user #12, who felt that Sluggo got angry much too quickly. It turned out, however, that Sluggo's aggression resulted from the following exchange:

```
Sluggo's voice says ``Cool.  I didn't know turd could
talk.''.
PLAYER> Sluggo:  Hey! Why did you call me a turd?

You are speaking to Sluggo.
Player's voice says ``Hey! Why did you call me a turd?''.
Sluggo is now red.
Sluggo is now scowling.
Sluggo is now tense.
Sluggo is speaking to you.
Sluggo's voice says ``Prepare to be pounded!''.
```

  Although this seems to be a problem with the emotion system, it is really an instance of the limitations of the natural language understanding system. In this case, Sluggo has simply matched the word "turd" and assumed that the player's utterance was an insult directed at him. Imagine replacing "Hey! Why did you call me a turd?" with "Hey, you're a turd!" and it becomes more clear why Sluggo reacted as he did.

  The second example came from user #4 who ran into difficulty when trying to interact with Melvin from the tree house. Because Chuckie felt no fear, he had no problems entering the tree house and interacting with the user even when Sluggo was in the tree house. When the user tried to engage Melvin from the tree house (Melvin was on the playground at the time), Melvin wanted to interact but was too scared of Sluggo to enter the tree house. Instead of having Melvin be direct and tell the user that he was too scared of Sluggo, Melvin tried to indicate his situation by looking at the player, looking nervously at Sluggo, and fidgeting. The user was unable to pick up on this subtlety and found Melvin unresponsive and unbelievable.

  In this case, the emotion, in combination with poor artistic decisions actually hurt believability. I tried to follow the "show, don't tell" advice of writing and have Melvin act out his emotions instead of just speaking them. Unfortunately, I failed to follow the "show it clearly" axiom. This left the user confused and unable to interact naturally with Melvin.

11.3.2 Extending Em

In Chapter 6, I described a three-subject study I ran to help suggest ways to improve the Em tools. In addition to the suggestions from this study, which have already been discussed, I have a number of additional suggestions of ways to extend Em that might be useful. My suggestions are ways to make Em even more general and flexible than it currently is, though I don't know which of these ideas will actually prove useful to artists.

The emotion generation suggestions are a sampling of ideas from the emotion research of Ortony, Clore, and Collins [Ortony88]. The ideas about emotion expression come from a wider range of sources and are cited below.

Some extensions to emotion generation

- The amount of control the agent has over the outcome of the goal can affect the intensity of various emotion structures, like hope, fear, and anger. I would attach an amount-of-control function with goals in a manner similar to that used for computing the likelihood of goals succeeding and failing.
- The effort expended towards achieving a goal can affect the intensity of emotion types like satisfaction and disappointment. Like the previous example, this might be accomplished with a new function annotation for goals.
- The degree of success or failure (when this is relevant) of the goal can affect the intensity and types of goal-based emotions. The idea of partial success or failure might not fit cleanly into the Hap language; if necessary, Hap would have to be extended.
- Whether the cause of the emotion is real, remembered, or imaginary can affect the type and intensity of the emotion. Currently only real causes exist. If other causes were to be used, there would either need to be new emotion generators written or the old generators would have to be modified to take information about the type of object being processed into account.
- If a group or an agent from a group that an agent is involved with does something, that agent might have emotions as if the agent had performed the action. This might be accomplished by modifying the existing standard-based emotion generators (e.g., shame, pride) to take these sorts of relationships into account.
- Social knowledge about how much an agent deserves a pleasing or displeasing event can affect both the type and intensity of happy-for, pity, resentment,

and gloating reactions. This might be accomplished by adding inferencing (possibly very simple inferencing) to these generators to reason about when agents are deserving or not deserving.

Some extensions to emotion expression

My belief is that between emotion generation and emotion expression, the most important to focus on for future improvements is emotion expression. Here are a few areas to explore.

- Memory. Emotions could be used to affect what characters remember. Being in one emotional state could result in previous experiences of that state being retrieved. Negative emotions (like sadness) could trigger goals to recall past positive situations in order to help overcome the emotion. In order to achieve these effects, some sort of episodic memory system is needed. The retrieval mechanisms should allow for mood-based queries, so that emotional experiences of a certain type can be recalled.
- Learning. Emotions could be used as signals to a learning system about when or what to learn. They could also be used in some circumstances to set a quiet state that is conducive to learning. To create this kind of effect, the agent clearly needs some sort of learning system. Any restrictions on the type of learning system are unclear. For ideas about the relationship between depression and learning, see [Webster92].
- Daydreaming. Emotions could drive many different kinds of daydreams, such as dreams of revenge or dreams being driven by anger at someone. Mueller [Mueller90] has examined the issue of the relationship between emotions and daydreaming in depth.
- Natural Language. Although I use some natural language in the characters I have built, I have not examined natural language issues in any depth in this research. This is the subject of [Loyall96], [Kantrowitz96], [Hovy88] and other related literature which I will not attempt to cite. Nonetheless, here are a few suggestions about how the natural language architecture should take emotions into account.
  - **Natural Language Generation**

    Lexical. Emotions should be able to affect word choice.

    Syntactic. Emotions should be able to affect how sentences are structured.

    Semantic. Emotions should be able to influence what to say.

    Speech. Emotions should be able to influence speech behaviors in order to achieve effects like talking quickly and nervous stuttering.

- **Natural Language Understanding**

  Attention. Emotions could cause attention to move away from or towards a particular speaker.

  Parsing. Emotions could influence parsing behaviors. For instance, an angry agent may be too impatient to parse an overly complex sentence.

  Understanding. Emotions could influence how a particular utterance is interpreted. An example is deciding whether something is a sincere compliment or a sarcastic insult.

11.3.3 Easing the Creation of Believable Social Behaviors

One drawback of my approach to building believable social behaviors is that by requiring artists to rebuild social behaviors for each new character they create, they lose all of the previous code that they wrote. For instance, if I were to create a negotiating behavior for a new character, I would have to start almost from scratch despite having created a number of variations of this behavior already. I have argued that my approach is reasonable because it promotes better characters and it allows a wide range of social behaviors. It would be nice, however, to keep the benefits of my approach while supporting more code re-use.

I don't have any full solutions to this problem, or I would have built them. I would speculate, however, that there might be tools that could ease the process of building believable social behaviors without taking control away from the artist. For example, it might be possible to create a tool that would help artists add some common aspects of the character's personality to behaviors. One such tool might assist in adding emotional tests to preconditions that affect how likely a character is to engage in a certain social behavior. The artist would still be responsible for much of the character- and behavior-specific code, but it would ease the task somewhat.

Another approach would be to create a large set of behavior libraries that artists can build from. Although I don't believe a full set of behaviors can be built and that artists will therefore have to build new behaviors, this approach can be useful at least some of the time. I also believe that much of the code in such library behaviors will need to be modified, extended, or deleted to create personality-rich behaviors. If only part of the code is useful, however, it still simplifies the task. This approach runs the risk of encouraging generic behavior, since it is easier to use the generic code that has already been written, but I would hope that good artists would make the effort to avoid such a trap.

One of the problems with making artists create social behaviors for each new character is that it makes the problem of scaling up more difficult. The characters

I have built were able to interact with users for 20 minutes at a time, with Sluggo going for two 20 minute interactions with the same user. This shows that it is possible to build short stories and individual scenes, but it is not clear that this same technique would scale up to a character that needed to be in many different scenes and have, say, a two hour life-span.

One suggestion is that it might be possible to create larger experiences out of a number of smaller ones. For instance, there might be an interaction with six scenes with a character that appears in all of them. There could be six different variations of the character, one for each scene, that are built specifically to handle interactions within a particular scene. This makes the problem of scaling up almost linear. Presumably, there will have to be some state that is shared by multiple instantiations of the same character so that early interactions affect later ones, but this state might be rather limited compared to the overall complexity of one of these agents.

11.3.4 Extending Tok

Another interesting area to pursue is the continued filling out of the Tok agent architecture. Tok is a fairly broad architecture, but it still has a number of holes, such as episodic memory. To fill in these holes requires the development of new capabilities for agents that are consistent with the artistic goals of building believable agents. The new capabilities also have to be integrated with the existing capabilities. For instance, I have spoken a number of times about the possibility for ties between the emotion and memory systems; once a memory system is in place, one would need to flesh these links out.

There are also parts of the architecture like perception where there is currently a working system that could be further developed artistically. For instance, perception should not just be a way of providing information to the other subsystems; it should also be a form of expression itself. Sad agents could look down, afraid agents could be wide eyed and focused on the cause of their fear, and nervous agents could look around quickly. Each of these examples shows how the eyes can be used not just as input devices, but as output devices for the emotion system. The current Tok perception system allows for a little of this, but not in any coherent, systematic way that also takes goals, actions, and other important facets of the agent's internal state into account.

11.3.5 Other Application Areas

There are a number of classes of applications that might benefit from believable social and emotional agents besides artistic applications. For instance, educational and training applications can put a student in a simulation with other agents for some educational (as opposed to dramatic) purpose. Work along these lines is being done by Kass et al. at Northwestern University [Kass92]. These systems have characters with social skills and some emotions, but they could be improved to be even more emotional and have better personalities.

Improving the personalities of agents in education and training system has two potential advantages. First, improving personalities would make the characters more engaging and make the experience more enjoyable. This seems especially important for children's applications. Second, by applying artistic techniques to create clearer characters, the system designer might be better able to focus on the important aspects of the agents' personalities from an educational standpoint. In other words, the agents become more clear, though less realistic. This could potentially be a powerful educational technique.

Another way that my tools might apply to training and education is to help create more immersive environments. I have spent a lot of time trying to create robust social agents—that is, agents that are able to act reasonably even when things go in somewhat unexpected directions. Because this is a hard problem, current training and education systems often use techniques like those used in games; they use video or animation clips and limit interactions to point-and-click or menu-based approaches (e.g., Kass's GuSS system [Kass92]). Using my approach to building social agents, it might be possible to develop interfaces for simulation-based training and education systems where the user has a greater feeling of freedom than is currently available. This feeling of freedom could potentially lead to more immersive and more effective educational experiences.

Another possible area to explore is robotics. Some initial work has already been done in the creation of emotional robots (e.g., [Pfeifer93], [Yamamoto94]), though with very simple tasks and emotion models. I also expect work like that of Sloman and Beaudoin [Sloman94, Beaudoin94] to be useful in this area, even though their goal is not explicitly a robotics one. Because my architecture has been designed to be flexible, it could be useful as a testbed for the development of emotion systems for robots that are not necessarily artistic.

Very little robotics research has been directed towards making human-robot interactions go smoothly. I think my methodology might be useful in this area. If it turns out that humans are more willing or more comfortable interacting with robots with interesting personalities, my methodology could be applied to this domain.

A similar issue comes up in the area of interface agents. Like robots, it isn't clear whether or not such agents should have engaging personalities. If personality-rich agents are found to be effective interfaces, however, my methodology could provide a starting point for the creation of such agents.

11.4 The Art of Building Social and Emotional Believable Agents

In this thesis, I have discussed a lot of technology and a little bit of art, mostly as it relates to the design of the technology. A critical element of success in this new artistic medium will be the development of artistic methods for using the tools I have built.

Artists develop their techniques through lots of practice and experimentation. For instance, one of the keys to success for the early Disney animators was the ability to use pencil sketches and other techniques to create quick prototypes; this allowed them to test lots of ideas very quickly [Thomas81]. Similarly, I expect that many of the artistic techniques for creating believable agents will only develop with time and practice. Throughout this thesis, I have tried to point out what *could* be done with my tools—I have not, however, made artistic claims about what *should* be done in order to be artistically most effective.

I look forward to the development of the art and to the new kinds of artists that will make believable agents and interactive drama possible.

Appendices

APPENDIX A Traces from the Simulation Systems

In this appendix, I include traces of interactions with the three simulation systems that I described in the thesis: *Robbery World*, *Office Politics*, and *The Playground*. I include two traces from each system. None of the traces in this appendix have been edited except to remove system output, such as garbage collection messages, and to fix some formatting in the preludes. Some of the output is a little rough because it is all computer generated. Remember that the characters are the important thing here, not the interface.

In these traces, the user of the system types commands at the “PLAYER>” prompt. Everything else is output by the computer.

A.1 Robbery World

In *Robbery World*, the player takes the part of a police officer who gets a call about a convenience store holdup that he/she must try to stop. Here are two traces from interactions with this world.

Trace #1

```
Welcome to the Pittsburgh police force!  You've been assigned to
the Squirrel Hill section of the city because of a rash of hold-
ups over the past few weeks.  So far things have been pretty qui-
et as you've been walking your beat.

You're just about to start heading back to the station when the
dispatcher's voice comes over your police radio:
```

```
   ``Report to the Quik-Go convenience store north of your
      current location.  Robbery in progress.  Backup has
      been notified.''

Well, it looks like it might not be such a quiet day after all.

   Good luck and be careful out there!

[Please hit CR.]

You are in the road.


   To the north, you see the parking lot.

   You are holding your gun.
   You are wearing your police uniform.
PLAYER> go north

   You go to the north.

You are in the parking lot.


   To the south, you see the road.

   To the north, you see the glass door.

   To the north, you see the convenience store.
   The counter, the cashier and the gunman are in the convenience
store.
   The gunman is holding his gun.
   The gunman is wearing the ski mask.
   The cashier is holding the bag.
   The cash register is on the counter.

   The car is in the parking lot.

   The gunman is now red.
   The gunman is now scowling.
   The gunman is now tense.
   The gunman is speaking to you.
   Gunman's voice says ``Back off and nobody gets hurt!''.
   The cashier is speaking to the gunman.
   Cashier's voice says ``
   Please don't kill me.  I'll give you whatever you want.''.
PLAYER> go north

   You go to the north.
```

```
You are in the convenience store.


   To the south, you see the glass door.

   To the south, you see the parking lot.
   The car is in the parking lot.

   The counter, the cashier and the gunman are in the convenience
store.
   The gunman is holding his gun.
   The gunman is wearing the ski mask.
   The cashier is holding the bag.
   The cash register is on the counter.

   The gunman takes hostage the cashier.
   The cashier is now pale.
   The cashier is now bug-eyed.
   The cashier is now trembling.
   The cashier waits.
PLAYER> gunman: aim my gun at the gunman

   You are speaking to the gunman.
   Player's voice says ``aim my gun at the gunman''.
   The gunman is speaking to you.
   Gunman's voice says ``I've got a hostage.  Back off or he gets
it.''.
   The cashier waits.
PLAYER> gunman: let him go now!

   You are speaking to the gunman.
   Player's voice says ``let him go now!''.
   The gunman is speaking to you.
    Gunman's voice says ``I'll do what I want to do.  Now shut
up!''.
   The cashier waits.
PLAYER> gunman: this is your last warning, free the hostage or
else!

   You are speaking to the gunman.
   Player's voice says ``
   this is your last warning, free the hostage or else!''.
   The gunman runs to the parking lot.
   The cashier runs to reluctantly the parking lot.
PLAYER> shoot the gunman

   Your action failed because you missed your target.

   You hear a gunshot.
   The gunman gets in the car.
```

```
   The cashier gets in reluctantly the car.
PLAYER> again
The thief drives off.  He seems to have escaped.
```

Trace #2

```
Welcome to the Pittsburgh police force!  You've been assigned to
the Squirrel Hill section of the city because of a rash of hold-
ups over the past few weeks.  So far things have been pretty qui-
et as you've been walking your beat.

You're just about to start heading back to the station when the
dispatcher's voice comes over your police radio:
   ``Report to the Quik-Go convenience store north of your
      current location.  Robbery in progress.  Backup has
      been notified.''

Well, it looks like it might not be such a quiet day after all.

             Good luck and be careful out there!

[Please hit CR.]


You are in the road.


   To the north, you see the parking lot.

   You are holding your gun.
   You are wearing your police uniform.
PLAYER> go north

   You go to the north.

You are in the parking lot.


   To the south, you see the road.

   To the north, you see the glass door.

   To the north, you see the convenience store.
   The counter, the cashier and the gunman are in the convenience
store.
   The gunman is holding his gun.
   The gunman is wearing the ski mask.
   The cashier is holding the bag.
   The cash register is on the counter.
```

```
   The car is in the parking lot.

   The gunman is now red.
   The gunman is now scowling.
   The gunman is now tense.
   The gunman is speaking to you.
   Gunman's voice says ``Back off and nobody gets hurt!''.
   The cashier is speaking to the gunman.
   Cashier's voice says ``
   Please don't kill me.  I'll give you whatever you want.''.
PLAYER> gunman: Come out now, you low-life scum!

   You are speaking to the gunman.
   Player's voice says ``Come out now, you low-life scum!''.
   The gunman is speaking to you.
   Gunman's voice says ``
    Hey, shut your trap!  I ain't listening to none of your
crap!''.
   The cashier opens the cash register.
PLAYER> gunman: Okay, jerk, this is you last warning!

   You are speaking to the gunman.
   Player's voice says ``Okay, jerk, this is you last warning!''.
   The gunman is speaking to the cashier.
   Gunman's voice says ``Move it or you'll be sorry!''.
   The cashier takes money.
PLAYER> shoot the gunman

   Your action failed because you missed your target.

   You hear a gunshot.
   The gunman aims his gun at you.
   The cashier places money in the bag.
PLAYER> shoot the gunman

   You shoot the gunman.
   You hear a gunshot.
   The gunman is now wounded.
   The gunman is now pale.
   The gunman is now bug-eyed.
   The gunman is now trembling.
   The gunman fires his gun at you.
    The bullet missed its target

   You hear a gunshot.
   The cashier offers the bag to the gunman.
PLAYER> shoot the gunman
You killed the gunman; the holdup is thwarted.  Too bad about all
that paperwork.
```

A.2 Office Politics

In this simulation, the player takes the part of a project manager at MegaCorp. The player is responsible for getting out an important project proposal by 5pm, but the email server is down. The player has to get help from the other characters in the world, which include Mary, the player's boss; Sarah, another busy project manager with a hidden desire to make the player look bad; and Gus, the one-man, overworked tech support department.

Trace #1

```
After a long time flipping burgers, repairing copiers, and sell-
ing used sporting equipment, you finally got your break with
MegaCorp. Congratulations and welcome to the corporate world.  If
all goes well, you might even be promoted to manager someday.

Your current task is to send some email to a prospective client.
If you want to get this project, you've got to get the mail out
by 5:00 today.  Unfortunately, the mail server is down and you
didn't learn how to fix email servers at your copier repair job.

Fortunately, there are a number of people around the office who
might be able (and willing) to help you.  Mary is your boss.  She
is usually very busy, but this is an important project and she
might be able to help.  Gus is the server maintainer.  He's usu-
ally the person to ask, but he can also be very busy and might
not be able to help you in time.  Sarah is a co-worker of yours.
She's got a deadline of her own to meet, so she's probably not
going to have time to help you out.

  Good luck!

[Please hit CR.]


You are in the office.


   To the north, you see the western hallway.

   The desk and the chair are in your office.

[You're not carrying anything.]

   You are wearing your business suit.
```

```
   You are speaking.
   Player’s voice says ``
     Welcome to ``Office Politics’’!  For some suggestions on how
to use
      Oz, type help.  For help with this simulation, type help-
world.’’.
[4:32] PLAYER> go north

   You go to the north.

You are in the western hallway.


   To the north, you see Mary’s office.
   The desk, the chair and Mary are in Mary’s office.

   To the south, you see your office.
   The desk and the chair are in your office.

   To the east, you see the eastern hallway.

   Mary works.
[4:33] PLAYER> go north

   You go to the north.

You are in the Mary’s office.


   To the south, you see the western hallway.

   The desk, the chair and Mary are in Mary’s office.

   Mary works.
[4:34] PLAYER> mary: can you help me fix the server?  it’s down
again.

   You are speaking to Mary.
   Player’s voice says ``
   can you help me fix the server?  it’s down again.’’.
   Mary is speaking to you.
   Mary’s voice says ``
   I don’t know how to fix the server.  Why don’t you ask Gus?’’.
[4:35] PLAYER> mary: okay, thanks

   You are speaking to Mary.
   Player’s voice says ``okay, thanks’’.
   Mary is speaking to you.
   Mary’s voice says ``You’re welcome.’’.
[4:36] PLAYER> go south
```

```
   You go to the south.

You are in the western hallway.


   To the north, you see Mary's office.
   The desk, the chair and Mary are in Mary's office.

   To the south, you see your office.
   The desk and the chair are in your office.

   To the east, you see the eastern hallway.

   Mary works.
[4:37] PLAYER> go east

   You go to the east.

You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah's office.
   The desk and the chair are in Sarah's office.

   To the south, you see Gus's office.
   The desk, the chair and Gus are in Gus's office.

   To the west, you see the western hallway.

   Sarah is in the eastern hallway.
   Sarah is holding the project report.

   Sarah is speaking to you.
   Sarah's voice says ``Could you fix the copier for me?
Please.''.
   Gus works.
[4:38] PLAYER> sarah: sure, if you fix the server for me.

   You are speaking to Sarah.
   Player's voice says ``sure, if you fix the server for me.''.
   Sarah is speaking to you.
   Sarah's voice says ``Great!  Thanks a lot.''.
   Gus works.
[4:39] PLAYER> sarah: so where is the copier again?

   You are speaking to Sarah.
   Player's voice says ``so where is the copier again?''.
```

```
   Sarah is speaking to you.
   Sarah's voice says ``It's in the equipment room.''.
   Gus works.
[4:40] PLAYER> sarah: ah, right, thanks.

   You are speaking to Sarah.
   Player's voice says ``ah, right, thanks.''.
   Sarah is speaking to you.
   Sarah's voice says ``Sure, anytime.''.
   Gus works.
[4:41] PLAYER> go east

   You go to the east.

You are in the equipment room.


   To the west, you see the eastern hallway.
   Sarah is in the eastern hallway.

   The email server and the copier are in the equipment room.

   Sarah goes to the equipment room.
[4:42] PLAYER> examine the copier

   You look at the copier.
   The copier is in the equipment room.
   The copier is not working.


   Sarah waits.
[4:43] PLAYER> repair the copier

   You fix the copier.
   The copier is now working.
   Sarah places the project report on the copier.
[4:44] PLAYER> sarah: could you fix the server before you do
that?

   You are speaking to Sarah.
   Player's voice says ``could you fix the server before you do
that?''.
   Sarah is speaking to you.
   Sarah's voice says ``
   Don't worry, I'll take care of that by 5:00 for you.''.
[4:45] PLAYER> sarah: are you sure?  this is really important.

   You are speaking to Sarah.
   Player's voice says ``are you sure?  this is really impor-
tant.''.
```

```
   Sarah copies the project report.
[4:46] PLAYER> z

   You wait.
   Sarah takes the project report.
[4:47] PLAYER> z

   You wait.
   Sarah takes the copy.
[4:48] PLAYER> z

   You wait.
   Sarah goes to the eastern hallway.
[4:49] PLAYER> w

   You go to the west.

You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah’s office.
   The desk and the chair are in Sarah’s office.

   To the south, you see Gus’s office.
   The desk, the chair and Gus are in Gus’s office.

   To the west, you see the western hallway.

   Sarah is in the eastern hallway.
   Sarah is holding the project report and the copy.

   Sarah goes to Sarah’s office.
   Gus works.
[4:50] PLAYER> n

   You go to the north.

You are in the Sarah’s office.


   To the south, you see the eastern hallway.

   The desk, the chair and Sarah are in Sarah’s office.
   Sarah is holding the project report and the copy.

   Sarah works.
[4:51] PLAYER> Sarah: it’s almost 5:00 -- could you fix the serv-
er now?
```

```
   You are speaking to Sarah.
   Player’s voice says ``
   it’s almost 5:00 -- could you fix the server now?’’.
   Sarah is speaking to you.
   Sarah’s voice says ``
   I can’t do it this minute, but I’ll get to it before the end
of the day.’’.
[4:52] PLAYER> s

   You go to the south.

You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah’s office.
   The desk, the chair and Sarah are in Sarah’s office.

   To the south, you see Gus’s office.
   The desk, the chair and Gus are in Gus’s office.

   To the west, you see the western hallway.

   Sarah works.
   Gus works.
[4:53] PLAYER> s

   You go to the south.

You are in the Gus’s office.


   To the north, you see the eastern hallway.

   The desk, the chair and Gus are in Gus’s office.

   Gus works.
[4:54] PLAYER> gus: please help me, I need the email server fixed
pronto

   You are speaking to Gus.
   Player’s voice says ``
   please help me, I need the email server fixed pronto’’.
   Gus is speaking to you.
   Gus’s voice says ``
   Sorry, bud.  I’d like to help, but I’m swamped today.’’.
[4:55] PLAYER> gus: this is really important; I have a 5:00 dead-
line so I’m in a hurry
```

```
   You are speaking to Gus.
   Player's voice says ``
    this is really important; I have a 5:00 deadline so I'm in a
hurry''.
   Gus is speaking to you.
   Gus's voice says ``Oh, OK.  I'll help you out, bud.''.
[4:56] PLAYER> gus: thanks a lot!

   You are speaking to Gus.
   Player's voice says ``thanks a lot!''.
   Gus goes to the eastern hallway.
[4:57] PLAYER> n

   You go to the north.

You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah's office.
   The desk, the chair and Sarah are in Sarah's office.

   To the south, you see Gus's office.
   The desk and the chair are in Gus's office.

   To the west, you see the western hallway.

   Gus is in the eastern hallway.

   Sarah works.
   Gus goes to the equipment room.
[4:58] PLAYER> e
The email server has been fixed and you are able to send
your mail in time.  Congratulations on a fine day's work.
```

Trace #2

After a long time flipping burgers, repairing copiers, and selling used sporting equipment, you finally got your break with MegaCorp. Congratulations and welcome to the corporate world. If all goes well, you might even be promoted to manager someday.

Your current task is to send some email to a prospective client. If you want to get this project, you've got to get the mail out by 5:00 today. Unfortunately, the mail server is down and you didn't learn how to fix email servers at your copier repair job.

Fortunately, there are a number of people around the office who might be able (and willing) to help you. Mary is your boss. She is usually very busy, but this is an important project and she might be able to help. Gus is the server maintainer. He's usually the person to ask, but he can also be very busy and might not be able to help you in time. Sarah is a co-worker of yours. She's got a deadline of her own to meet, so she's probably not going to have time to help you out.

```
  Good luck!

[Please hit CR.]

You are in the office.


   To the north, you see the western hallway.

   The desk and the chair are in your office.

[You're not carrying anything.]

   You are wearing your business suit.

   You are speaking.
   Player's voice says ``
   Welcome to ``Office Politics''!  For some suggestions on how
to use
   Oz, type help.  For help with this simulation, type help-
world.''.
[4:32] PLAYER> go north

   You go to the north.

You are in the western hallway.


   To the north, you see Mary's office.
   The desk, the chair and Mary are in Mary's office.

   To the south, you see your office.
   The desk and the chair are in your office.

   To the east, you see the eastern hallway.

   Mary works.
[4:33] PLAYER> go east

   You go to the east.
```

```
You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah's office.
   The desk, the chair and Sarah are in Sarah's office.

   To the south, you see Gus's office.
   The desk, the chair and Gus are in Gus's office.

   To the west, you see the western hallway.

   Sarah takes NO-REFERENT.
   Gus works.
[4:34] PLAYER> go south

   You go to the south.

You are in the Gus's office.


   To the north, you see the eastern hallway.

   The desk, the chair and Gus are in Gus's office.

   Gus works.
[4:35] PLAYER> Gus: Hi!

   You are speaking to Gus.
   Player's voice says ``Hi!''.
   Sarah goes to NO-REFERENT.
   Gus is speaking to you.
   Gus's voice says ``Hey, bud!''.
[4:36] PLAYER> Gus: Can you help me fix the email server?

   You are speaking to Gus.
   Player's voice says ``Can you help me fix the email server?''.
   Gus is speaking to you.
   Gus's voice says ``
   Sorry, bud.  I'd like to help, but I'm swamped today.''.
[4:37] PLAYER> Gus: Oh, okay.  Thanks anyway.

   You are speaking to Gus.
   Player's voice says ``Oh, okay.  Thanks anyway.''.
   Sarah goes to Gus's office.
   Gus is speaking to you.
   Gus's voice says ``No problem, dude.''.
[4:38] PLAYER> Sarah: Hi.
```

```
   You are speaking to Sarah.
   Player's voice says ``Hi.''.
   Sarah is speaking to you.
   Sarah's voice says ``Hi.''.
   Gus works.
[4:39] PLAYER> go north

   You go to the north.

You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah's office.
   The desk and the chair are in Sarah's office.

   To the south, you see Gus's office.
   The desk, the chair, Gus and Sarah are in Gus's office.

   To the west, you see the western hallway.

   Sarah is speaking to you.
   Sarah's voice says ``Could you fix the copier for me?
Please.''.
   Gus works.
[4:40] PLAYER> Sarah: Sorry, I'm too busy today.

   You are speaking to Sarah.
   Player's voice says ``Sorry, I'm too busy today.''.
   Sarah is speaking to Gus.
   Sarah's voice says ``Hey, would you fix the copier for me?''.
   Gus is speaking to Sarah.
   Gus's voice says ``
   Sorry, I can't do it right now, but I'll get to it first thing
tomorrow.''.
[4:41] PLAYER> go west

   You go to the west.

You are in the western hallway.


   To the north, you see Mary's office.
   The desk, the chair and Mary are in Mary's office.

   To the south, you see your office.
   The desk and the chair are in your office.

   To the east, you see the eastern hallway.
```

```
   Mary works.
[4:42] PLAYER> go north

   You go to the north.

You are in the Mary’s office.


   To the south, you see the western hallway.

   The desk, the chair and Mary are in Mary’s office.

   Mary works.
[4:43] PLAYER> mary: Mary, can you help me fix the email server?

   You are speaking to Mary.
   Player’s voice says ``Mary, can you help me fix the email
server?’’.
   Mary is speaking to you.
   Mary’s voice says ``
   I don’t know how to fix the server.  Why don’t you ask Gus?’’.
[4:44] PLAYER> Mary: I did, but gus is too busy to do it.

   You are speaking to Mary.
   Player’s voice says ``I did, but gus is too busy to do it.’’.
   Mary is speaking to you.
   Mary’s voice says ``
   Alright, I’ll talk to Gus, but I really don’t have time for
this.’’.
[4:45] PLAYER> mary: Thanks, Mary.

   You are speaking to Mary.
   Player’s voice says ``Thanks, Mary.’’.
   Mary goes to the western hallway.
[4:46] PLAYER> go south

   You go to the south.

You are in the western hallway.


   To the north, you see Mary’s office.
   The desk and the chair are in Mary’s office.

   To the south, you see your office.
   The desk and the chair are in your office.

   To the east, you see the eastern hallway.
```

```
   Mary is in the western hallway.

   Mary goes to the eastern hallway.
[4:47] PLAYER> go east

   You go to the east.

You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah’s office.
   The desk, the chair and Sarah are in Sarah’s office.

   To the south, you see Gus’s office.
   The desk, the chair and Gus are in Gus’s office.

   To the west, you see the western hallway.

   Mary is in the eastern hallway.

   Sarah works.
   Mary goes to Gus’s office.
   Gus works.
[4:48] PLAYER> go south

   You go to the south.

You are in the Gus’s office.


   To the north, you see the eastern hallway.

   The desk, the chair, Gus and Mary are in Gus’s office.

   Mary is speaking to Gus.
   Mary’s voice says ``
   Gus, could you fix the email server?  We’re in a bit of a rush.
Thanks.’’.
   Gus is speaking to Mary.
   Gus’s voice says ``I’ll do that right now.’’.
[4:49] PLAYER> wait

   You wait.
   Mary waits.
   Gus goes to the eastern hallway.
[4:50] PLAYER> go north

   You go to the north.
```

```
You are in the eastern hallway.


   To the east, you see the equipment room.

   To the north, you see Sarah's office.
   The desk, the chair and Sarah are in Sarah's office.

   To the south, you see Gus's office.
   The desk, the chair and Mary are in Gus's office.

   To the west, you see the western hallway.

   Gus is in the eastern hallway.

   Sarah works.
   Mary goes to the eastern hallway.
   Gus goes to the equipment room.
[4:51] PLAYER> go east
The email server has been fixed and you are able to send
your mail in time.  Congratulations on a fine day's work.
```

A.3 The Playground

In this world, the player is a child on a schoolyard playground. The player has a few baseball cards and wants to get a Willie Mays trading card. The other characters are two other children, Melvin and Sluggo.

Trace #1

```
The recess bell has just rung and it's time to really
start working.  Math, English, and Social Studies are
nothing compared to the harsh competition out on the
playground.

Try to collect baseball cards of players you like by
trading with the other kids on the playground.  Who
knows, this may even be your big chance to get that
Willie Mays card you've been trying to get for so long!

                     Have fun!


[Please hit CR.]

You are in the playground.
```

```
    The sand box, the jungle gym and the tree house are in the
playground.
   Sluggo is in the tree house.
   Sluggo is holding a Willie Mays trading card, a
   Jose Canseco trading card and a Catfish Hunter trading card.
   Melvin is in the sand box.
   Melvin is holding a Tom Seaver trading card, a
   Mickey Mantle trading card and a Reggie Jackson trading card.
   Melvin is wearing his eye glasses.

   You are holding a Babe Ruth trading card, a Ted Williams trad-
ing card
   and a Henry Aaron trading card.
PLAYER> get in the tree house

   You go into the tree house.
   Sluggo is speaking to you.
    Sluggo's voice says ``Wanna trade Catfish Hunter for Babe
Ruth?''.
   Melvin plays.
PLAYER> Sluggo: How about Ruth for Mays instead?

   You are speaking to Sluggo.
   Player's voice says ``How about Ruth for Mays instead?''.
   Sluggo is speaking to you.
    Sluggo's voice says ``I'll swap you Jose Canseco for Babe
Ruth.  Deal?''.
   Melvin is now smiling.
   Melvin looks nervously at Sluggo.
PLAYER> get in the sandbox

   You go into the sand box.
   Sluggo is speaking to you.
   Sluggo's voice says ``Hey, jerk, I was talking to you.''.
   Melvin is speaking excitedly to you.
   Melvin's voice says ``
   Hey Vulcan, I'll trade you Mickey Mantle for Babe Ruth, which
might help if you're trying to get Willie Mays from Sluggo!''.
PLAYER> Melvin: Cool.  Thanks.

   You are speaking to Melvin.
   Player's voice says ``Cool.  Thanks.''.
   Sluggo is now red.
   Sluggo is now scowling.
   Sluggo is now tense.
   Sluggo swears.
   Melvin fidgits.
PLAYER> give ruth to melvin
```

```
   You offer a Babe Ruth trading card to Melvin.
   Sluggo is now frowning.
   Sluggo glares at you.
   Melvin takes a Babe Ruth trading card.
PLAYER> melvin: okay, now give me mantle

   You are speaking to Melvin.
   Player's voice says ``okay, now give me mantle''.
   Sluggo scowls at you.
   Melvin offers a Mickey Mantle trading card to you.
PLAYER> take mantle

   You take a Mickey Mantle trading card.
   Sluggo swears.
   Melvin is speaking to you.
   Melvin's voice says ``
    I believe that your I.Q. may almost approach my own Vulcan-
like intelligence.''.
PLAYER> Melvin: Well thanks for saying so.

   You are speaking to Melvin.
   Player's voice says ``Well thanks for saying so.''.
   Sluggo smokes.
   Melvin is speaking to you.
   Melvin's voice says ``You're very welcome, Vulcan.''.
PLAYER> get in the tree house

   You go into the tree house.
   Sluggo swears.
   Melvin looks nervously at Sluggo.
PLAYER> Sluggo: Hi again.

   You are speaking to Sluggo.
   Player's voice says ``Hi again.''.
   Sluggo is speaking to you.
   Sluggo's voice says ``Cool.  I didn't know turd could talk.''.
   Melvin gets on the jungle gym.
PLAYER> Sluggo: So, do you want to trade mantle for mays?

   You are speaking to Sluggo.
    Player's voice says ``So, do you want to trade mantle for
mays?''.
   Sluggo is speaking to you.
   Sluggo's voice says ``Bud, it's a deal.''.
   Melvin looks at you.
PLAYER> wait

   You wait.
   Sluggo offers a Willie Mays trading card to you.
   Melvin gets on the jungle gym.
```

```
PLAYER> take mays

Congratulations.  You got Willie Mays.
```

Trace #2

```
The recess bell has just rung and it's time to really
start working.  Math, English, and Social Studies are
nothing compared to the harsh competition out on the
playground.

Try to collect baseball cards of players you like by
trading with the other kids on the playground.  Who
knows, this may even be your big chance to get that
Willie Mays card you've been trying to get for so long!

                    Have fun!


[Please hit CR.]

You are in the playground.


    The sand box, the jungle gym and the tree house are in the
playground.
   Sluggo is in the tree house.
   Sluggo is holding a Willie Mays trading card, a
   Jose Canseco trading card and a Catfish Hunter trading card.
   Melvin is in the sand box.
   Melvin is holding a Tom Seaver trading card, a
   Mickey Mantle trading card and a Reggie Jackson trading card.
   Melvin is wearing his eye glasses.

   You are holding a Babe Ruth trading card, a Ted Williams trad-
ing card
   and a Henry Aaron trading card.
PLAYER> get in the sandbox

   You go into the sand box.
   Sluggo spits.
   Melvin is now smiling.
   Melvin is speaking to you.
   Melvin's voice says ``
   Salutations, Vulcan ambassador!  The Klingon high command has
sent me in search of baseball cards.''.
PLAYER> melvin: What cards do the Klingon high command want?

   You are speaking to Melvin.
```

```
   Player’s voice says ``What cards do the Klingon high command
want?’’.
   Sluggo smokes.
   Melvin is speaking to you.
   Melvin’s voice says ``
   The Klingon Emperor wants to know if you would be willing to
part with Babe Ruth for Reggie Jackson?’’.
PLAYER> Melvin: No way!

   You are speaking to Melvin.
   Player’s voice says ``No way!’’.
   Sluggo spits.
   Melvin is speaking to you.
   Melvin’s voice says ``
   I wouldn’t be hasty if I were you.  Reggie Jackson for Babe
Ruth is a trade any Ferengi would be proud of.’’.
PLAYER> Melvin: How about Ruth for Jackson and Mantle?

   You are speaking to Melvin.
   Player’s voice says ``How about Ruth for Jackson and Man-
tle?’’.
   Sluggo swears.
   Melvin is speaking to you.
   Melvin’s voice says ``
   Captain Kirk would never settle for less than Babe Ruth and
Ted Williams for Reggie Jackson and Mickey Mantle.’’.
PLAYER> Melvin: I don’t know...

   You are speaking to Melvin.
   Player’s voice says ``I don’t know...’’.
   Sluggo smokes.
   Melvin whistles.
PLAYER> Sluggo: Hey, Sluggo, you shouldn’t smoke.

   You are speaking to Sluggo.
   Player’s voice says ``Hey, Sluggo, you shouldn’t smoke.’’.
   Sluggo is speaking to you.
   Sluggo’s voice says ``I do what I want, get it?’’.
   Melvin adjusts his eye glasses.
PLAYER> Sluggo: Okay, if you want to be a dork, go ahead.

   You are speaking to Sluggo.
   Player’s voice says ``Okay, if you want to be a dork, go
ahead.’’.
   Sluggo is now red.
   Sluggo is now scowling.
   Sluggo is now tense.
   Sluggo is speaking to you.
   Sluggo’s voice says ``Prepare to be pounded!’’.
   Melvin is speaking to you.
```

```
   Melvin's voice says ``
   Are there any cards we (the Klingon high command and I) can of-
fer you in return for Babe Ruth?''.
PLAYER> Sluggo: You don't scare me, you big jerk!

   You are speaking to Sluggo.
   Player's voice says ``You don't scare me, you big jerk!''.
   Sluggo goes into the sand box.
   Melvin is now pale.
   Melvin is now bug-eyed.
   Melvin is now trembling.
   Melvin goes into the tree house.
PLAYER> punch sluggo

   You punch Sluggo.
   Sluggo drops a Catfish Hunter trading card.
   Sluggo is now wounded.
   Sluggo punches you.
   You drop a Babe Ruth trading card.
   You drop a Ted Williams trading card.
   You drop a Henry Aaron trading card.
   You are now wounded.
   Melvin plays.
PLAYER> Sluggo: Uncle! Uncle! I give up.

   You are speaking to Sluggo.
   Player's voice says ``Uncle! Uncle! I give up.''.
   Sluggo punches you.
    The swing missed.

   Melvin plays.
PLAYER> punch sluggo

   Your action failed because you didn't land an effective blow.

   Sluggo punches you.
   You are now unconscious.
   Melvin plays.

******** You have been knocked unconscious. ********
```

APPENDIX B User Studies

This appendix includes some supporting materials to accompany the discussion of user studies that I ran.

Study 1 involved users interacting with two versions of *The Playground* and analyzing the characters. Section B.1 contains the instruction sheets and the questionnare for that study.

Study 2 involved subjects analyzing traces of three interactions with *Robbery World*. The traces included information about the emotion processing of one of the agents and the users were asked to analyze that information. Section B.2 contains the first of the three traces that I asked the subjects to analyze.

B.1 Study 1: Evaluating the Characters on The Playground

This study and its results are detailed in Chapters 6 and 10. This section includes the forms used in the study. First is the introduction and instruction sheet, followed by the questionnaire.

Instruction sheet (page 1 of 3):

Before you begin, please read the following instructions:

In the following experiment, you will be interacting with a number of characters is a simulated setting.

You get to control what actions you take by typing them in at the PLAYER> prompt. What you can see and hear in the world will be described to you.

You will be given as much time as you need to read the instruction sheet and the on-screen prelude. At any time during the encounter, you should feel free to refer back to the help sheet.

Feel free to interact with these characters any way you want, but your goal is not to break the characters or find their flaws. If the characters seem to "break" please continue with the simulation anyway.

You may use a pencil and paper to take notes during the encounter (e.g., to remember who owns what objects).

There will be occasional delays after you enter your action. These occur for many reasons. When this happens, please be patient. The system will return to the prompt after a few seconds.

Your interaction will end when either the simulation comes to a natural end or the experimentor call time.

Instruction sheet (page 2 of 3):

Here are some useful hints for interacting specifically with the playground simulation. You should read these hints even if you are familiar with other Oz worlds:

Welcome to “The Playground”. During this simulation, you will play the part of a child on a playground during recess.

During your interactions with the other characters in this environment, here are a couple of things to keep in mind.

SPEAKING

Speak to other characters by putting their name followed by a colon, followed by what you want to say. For example:

> PLAYER> Melvin: Hello.
> You are speaking to Melvin.
> Player’s voice says “Hello.”.

SWAPPING CARDS

One thing to do is to trade baseball cards with the other children. To do this, you will need to speak with whomever you are trading with. For instance:

> PLAYER> Melvin: Wanna trade Reggie Jackson for Ruth?

If the other child accepts your offer, or you accept one of their offers, you will be able to hand the cards back and forth. When handing over a card, be sure to indicate who you are handing it to. Also, you can only hand over one thing at a time. For example:

> PLAYER> Give Babe Ruth to Melvin

You will not be able to take a card until it is offered to you. In some cases, you may need to wait a moment until the other child offers you the card. For example:

> Melvin says, “OK, I’ll make that trade.”
>
> PLAYER> wait
> You wait.
> Melvin offers you the Reggie Jackson trading card.
>
> PLAYER> take the card
> You take the Reggie Jackson trading card.

Instruction sheet (page 3 of 3):

MOVING AROUND

There are a number of places in the playground for you to be and you can move freely between them. You can see from one location to another, but it is sometimes hard to interact effectively with other characters from a distance.

The locations are not in a north-south-east-west configuration. To get to another location you will need to indicate where you want to go. For example:

PLAYER> climb in the tree house
PLAYER> get in the sand box
PLAYER> get on the jungle gym

Particularly note this one, since it's a little unusual:

PLAYER> get in the playground

EXAMINING YOUR SURROUNDINGS

PLAYER> look around
Tells you what is around you. "l" is an abbreviation.

PLAYER> inventory
Tells you what you are carrying. "i" is an abbreviation.

Questionnaire: (page 1 of 7):

Impression of the Playground Characters

I. Here are some adjectives used to describe people. Please circle the number that indicates your impression of ***Sluggo****. If you think the value changed over the course of the interaction, please mark a high and a low value on the scale.*

Depressed	1	2	3	4	5	6	7	Cheerful
Unfriendly	1	2	3	4	5	6	7	Friendly
Pessimistic	1	2	3	4	5	6	7	Optimistic
Ignorant	1	2	3	4	5	6	7	Knowledgeable
Irresponsible	1	2	3	4	5	6	7	Responsible
Unintelligent	1	2	3	4	5	6	7	Intelligent
Foolish	1	2	3	4	5	6	7	Sensible
Disliked me	1	2	3	4	5	6	7	Liked me
Competetive	1	2	3	4	5	6	7	Cooperative
Passive	1	2	3	4	5	6	7	Dominant
Unemotional	1	2	3	4	5	6	7	Emotional
Poor negotiator	1	2	3	4	5	6	7	Great negotiator

Questionnaire (page 2 of 7):

II. Here are some questions about your impression of ***Sluggo****. Please circle the number that indicates your response.*

How much did you like Sluggo (as a person)?

Not at all	1	2	3	4	5	6	7	Very much

As a user, did you enjoy interacting with Sluggo (even if things didn't go your character's way and/or you didn't like him very much)?

Not at all	1	2	3	4	5	6	7	Very much

Did you feel Sluggo was being responsive to you or was he just following a script?

Script	1	2	3	4	5	6	7	Responsive

Do you think Sluggo had a clearly conveyed personality?

Not at all	1	2	3	4	5	6	7	Very much

Did Sluggo's personality come across even when he was negotiating with you?

Not at all	1	2	3	4	5	6	7	Very much

How good a character was Sluggo? (Pretend you're a movie critic critiquing the quality of the characters.)

Awful chcracter	1	2	3	4	5	6	7	Great character

Do you think Sluggo wanted to be your friend?

Not at all	1	2	3	4	5	6	7	Yes, definitely

Did Sluggo ever do anything to disrupt your "suspension of disbelief"?

Never	1	2	3	4	5	6	7	All the time

If you think Sluggo tried to become friends with you, please answer the following two questions:

How good a job did Sluggo do at winning you over to be his friend?

Terrible job	1	2	3	4	5	6	7	Great job

Did Sluggo's personality come across even when he was trying to become friends with you?

Not at all	1	2	3	4	5	6	7	Very much

Questionnaire (page 3 of 7): [This is identical to page 1, but for Melvin.]

*III. Here are some adjectives used to describe people. Please circle the number that indicates your impression of **Melvin**. If you think the value changed over the course of the interaction, please mark a high and a low value on the scale.*

Depressed	1	2	3	4	5	6	7	Cheerful
Unfriendly	1	2	3	4	5	6	7	Friendly
Pessimistic	1	2	3	4	5	6	7	Optimistic
Ignorant	1	2	3	4	5	6	7	Knowledgeable
Irresponsible	1	2	3	4	5	6	7	Responsible
Unintelligent	1	2	3	4	5	6	7	Intelligent
Foolish	1	2	3	4	5	6	7	Sensible
Disliked me	1	2	3	4	5	6	7	Liked me
Competetive	1	2	3	4	5	6	7	Cooperative
Passive	1	2	3	4	5	6	7	Dominant
Unemotional	1	2	3	4	5	6	7	Emotional
Poor negotiator	1	2	3	4	5	6	7	Great negotiator

Questionnaire (page 4 of 7): [This is identical to page 2, but for Melvin.]

*IV. Here are some questions about your impression of **Melvin**. Please circle the number that indicates your response.*

How much did you like Melvin (as a person)?

Not at all	1	2	3	4	5	6	7	Very much

As a user, did you enjoy interacting with Melvin (even if things didn't go your character's way and/or you didn't like him very much)?

Not at all	1	2	3	4	5	6	7	Very much

Did you feel Melvin was being responsive to you or was he just following a script?

Script	1	2	3	4	5	6	7	Responsive

Do you think Melvin had a clearly conveyed personality?

Not at all	1	2	3	4	5	6	7	Very much

Did Melvin's personality come across even when he was negotiating with you?

Not at all	1	2	3	4	5	6	7	Very much

How good a character was Melvin? (Pretend you're a movie critic critiquing the quality of the characters.)

Awful chcracter	1	2	3	4	5	6	7	Great character

Do you think Melvin wanted to be your friend?

Not at all	1	2	3	4	5	6	7	Yes, definitely

Did Melvin ever do anything to disrupt your "suspension of disbelief"?

Never	1	2	3	4	5	6	7	All the time

If you think Melvin tried to become friends with you, please answer the following two questions:
How good a job did Melvin do at winning you over to be his friend?

Terrible job	1	2	3	4	5	6	7	Great job

Did Melvin's personality come across even when he was trying to become friends with you?

Not at all	1	2	3	4	5	6	7	Very much

Questionnaire (page 5 of 7): [This is identical to page 1, but for Chuckie.]

V. Here are some adjectives used to describe people. Please circle the number that indicates your impression of ***Chuckie****. If you think the value changed over the course of the interaction, please mark a high and a low value on the scale.*

Depressed	1	2	3	4	5	6	7	Cheerful
Unfriendly	1	2	3	4	5	6	7	Friendly
Pessimistic	1	2	3	4	5	6	7	Optimistic
Ignorant	1	2	3	4	5	6	7	Knowledgeable
Irresponsible	1	2	3	4	5	6	7	Responsible
Unintelligent	1	2	3	4	5	6	7	Intelligent
Foolish	1	2	3	4	5	6	7	Sensible
Disliked me	1	2	3	4	5	6	7	Liked me
Competetive	1	2	3	4	5	6	7	Cooperative
Passive	1	2	3	4	5	6	7	Dominant
Unemotional	1	2	3	4	5	6	7	Emotional
Poor negotiator	1	2	3	4	5	6	7	Great negotiator

Questionnaire (page 6 of 7): [This is identical to page 2, but for Chuckie.]

VI. Here are some questions about your impression of ***Chuckie****. Please circle the number that indicates your response.*

How much did you like Chuckie (as a person)?

Not at all	1	2	3	4	5	6	7	Very much

As a user, did you enjoy interacting with Chuckie (even if things didn't go your character's way and/or you didn't like him very much)?

Not at all	1	2	3	4	5	6	7	Very much

Did you feel Chuckie was being responsive to you or was he just following a script?

Script	1	2	3	4	5	6	7	Responsive

Do you think Chuckie had a clearly conveyed personality?

Not at all	1	2	3	4	5	6	7	Very much

Did Chuckie's personality come across even when he was negotiating with you?

Not at all	1	2	3	4	5	6	7	Very much

How good a character was Chuckie? (Pretend you're a movie critic critiquing the quality of the characters.)

Awful chcracter	1	2	3	4	5	6	7	Great character

Do you think Chuckie wanted to be your friend?

Not at all	1	2	3	4	5	6	7	Yes, definitely

Did Chuckie ever do anything to disrupt your "suspension of disbelief"?

Never	1	2	3	4	5	6	7	All the time

If you think Chuckie tried to become friends with you, please answer the following two questions:

How good a job did Chuckie do at winning you over to be his friend?

Terrible job	1	2	3	4	5	6	7	Great job

Did Chuckie's personality come across even when he was trying to become friends with you?

Not at all	1	2	3	4	5	6	7	Very much

Questionnaire (page 7 of 7):

Please tell us about yourself:

Your age? __________

Your sex? __________

Your occupation? (Please be somewhat specific. I.e. “English grad student”, instead of “student”.____

How much do you know about baseball?

Nothing 1 2 3 4 5 6 7 A lot

How much experience do you have with interactive fiction/text adventure systems? (E.g., Zork)

None 1 2 3 4 5 6 7 A lot

Please leave this area blank:

Subject #:_______________
Time #1: ____________
Moves #1: __________
Time #2: ____________
Moves #2: __________
Order (M/C or C/M): ____________
Trace file: ___________

B.2 Study 2: Evaluating the Gunman's Emotions

I asked three subjects to evaluate the emotion processing of the gunman by analyzing three traces of interactions with the gunman with emotion-preocessing information included. The details of this study can be found in Chapter 6. In this section I include the first of the three traces that were analyzed.

This is the shortest of the three traces that I provided the subjects, which is why I chose to include it here. The others last six and ten turns (as compared to the four for this trace).

B.2.1. The Trace

```
Welcome to the Pittsburgh police force!  You've been assigned to
the Squirrel Hill section of the city because of a rash of hold-
ups over the past few weeks.  So far things have been pretty qui-
et as you've been walking your beat.

You're just about to start heading back to the station when the
dispatcher's voice comes over your police radio:
     ``Report to the Quik-Go convenience store north of your
       current location.  Robbery in progress.  Backup has
       been notified.''

  Well, it looks like it might not be such a quiet day after all.

             Good luck and be careful out there!

[Please hit CR.]

You are in the road.


   To the north, you see the parking lot.

   You are holding your gun.
   You are wearing your police uniform.

Tick 1...
[GUNMAN] Feature: (ENERGY NIL NIL) now set to 5.
[#<GUNMAN, name:GUNMAN>] Hope[a] increased by 4 in anticipation
of (SUBGOAL EXECUTE-HOLDUP).
[GUNMAN] Feature: (GOOD-MOOD NIL NIL) now set to 4.
[GUNMAN] Feature: (BAD-MOOD NIL NIL) now set to 0.
[GUNMAN] Feature: (ANTICIPATION NIL NIL) now set to 4.
GUNMAN is now SMIRKING.
```

```
PLAYER> Actor PLAYER is doing (*GO “DIRECTION (*NORTH*)” NIL NIL
NIL NIL).

Tick 2...
[#<GUNMAN, name:GUNMAN>] Fear[a] of OFFICER increased by 5 be-
cause of threat to (SUBGOAL PROTECT-SELF).
[#<GUNMAN, name:GUNMAN>] Fear[a] of OFFICER increased by 6 be-
cause of threat to (SUBGOAL EXECUTE-HOLDUP).
[#<GUNMAN, name:GUNMAN>] Fear[a] of OFFICER increased by 4 be-
cause of threat to (SUBGOAL DONT-BE-CAUGHT).
[#<GUNMAN, name:GUNMAN>] Distress[a] increased by 5 because (SUB-
GOAL PROTECT-SELF) more likely to fail.
[#<GUNMAN, name:GUNMAN>] Anger[a] towards OFFICER increased by 5
because of negative influence on goal (SUBGOAL PROTECT-SELF).
[#<GUNMAN, name:GUNMAN>] Distress[a] increased by 5 because (SUB-
GOAL EXECUTE-HOLDUP) more likely to fail.
[#<GUNMAN, name:GUNMAN>] Anger[a] towards OFFICER increased by 6
because of negative influence on goal (SUBGOAL EXECUTE-HOLDUP).
[#<GUNMAN, name:GUNMAN>] Distress[a] increased by 4 because (SUB-
GOAL DONT-BE-CAUGHT) more likely to fail.
[#<GUNMAN, name:GUNMAN>] Anger[a] towards OFFICER increased by 4
because of negative influence on goal (SUBGOAL DONT-BE-CAUGHT).
[GUNMAN] Feature: (GOOD-MOOD NIL NIL) now set to 0.
[GUNMAN] Feature: (BAD-MOOD NIL NIL) now set to 7.
[GUNMAN] Feature: (DEFENSIVE OFFICER NIL) now set to 6.
[GUNMAN] Feature: (DEFENSIVE NIL NIL) now set to 6.
[GUNMAN] Feature: (AGGRESSIVE OFFICER NIL) now set to 6.
[GUNMAN] Feature: (DEFENSIVE OFFICER (SUBGOAL EXECUTE-HOLDUP))
now set to 6.
[GUNMAN] Feature: (DEFENSIVE OFFICER (SUBGOAL PROTECT-SELF)) now
set to 5.
[GUNMAN] Feature: (UNFRIENDLY NIL NIL) now set to 3.
[GUNMAN] Feature: (DEFENSIVE OFFICER NIL) now set to 6.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL DONT-BE-CAUGHT))
now set to 4.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL EXECUTE-HOLDUP))
now set to 6.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL PROTECT-SELF)) now
set to 5.
[GUNMAN] Feature: (AGGRESSIVE NIL NIL) now set to 6.
[GUNMAN] Feature: (AGGRESSIVE OFFICER NIL) now set to 6.

   You go to the north.

You are in the parking lot.


   To the south, you see the road.

   To the north, you see the glass door.
```

```
   To the north, you see the convenience store.
   The counter, the cashier and the gunman are in the convenience
store.
   The gunman is holding the bag and his gun.
   The gunman is wearing the ski mask.
   The cash register is on the counter.

   The car is in the parking lot.

   The gunman is now red.
   The gunman is now scowling.
   The gunman is now tense.
   The gunman is speaking to you.
   Gunman's voice says ``Back off and nobody get's hurt!''.
   The cashier waits.

PLAYER> shoot gunman

Tick 3...
[#<GUNMAN, name:GUNMAN>] Distress from goal (SUBGOAL PROTECT-
SELF) decayed to 4.
[#<GUNMAN, name:GUNMAN>] Anger towards OFFICER (et al.) because
of negative influence on goal (SUBGOAL PROTECT-SELF) decayed to
4.
[#<GUNMAN, name:GUNMAN>] Distress from goal (SUBGOAL EXECUTE-
HOLDUP) decayed to 4.
[#<GUNMAN, name:GUNMAN>] Hope of NIL causing goal (SUBGOAL EXE-
CUTE-HOLDUP) to succeed decayed to 3.
[#<GUNMAN, name:GUNMAN>] Anger towards OFFICER (et al.) because
of negative influence on goal (SUBGOAL EXECUTE-HOLDUP) decayed to
5.
[#<GUNMAN, name:GUNMAN>] Distress from goal (SUBGOAL DONT-BE-
CAUGHT) decayed to 3.
[#<GUNMAN, name:GUNMAN>] Anger towards OFFICER (et al.) because
of negative influence on goal (SUBGOAL DONT-BE-CAUGHT) decayed to
3.
[#<GUNMAN, name:GUNMAN>] Fear[a] of OFFICER increased by 7 be-
cause of threat to (SUBGOAL PROTECT-SELF).
[#<GUNMAN, name:GUNMAN>] Distress[a] increased by 4 because (SUB-
GOAL PROTECT-SELF) more likely to fail.
[#<GUNMAN, name:GUNMAN>] Anger[a] towards OFFICER increased by 7
because of negative influence on goal (SUBGOAL PROTECT-SELF).
[GUNMAN] Feature: (ANTICIPATION NIL NIL) now set to 3.
[GUNMAN] Feature: (DEFENSIVE OFFICER NIL) now set to 7.
[GUNMAN] Feature: (GOOD-MOOD NIL NIL) now set to 0.
[GUNMAN] Feature: (BAD-MOOD NIL NIL) now set to 8.
[GUNMAN] Feature: (DEFENSIVE NIL NIL) now set to 7.
[GUNMAN] Feature: (DEFENSIVE OFFICER (SUBGOAL PROTECT-SELF)) now
set to 7.
```

```
[GUNMAN] Feature: (AGGRESSIVE OFFICER NIL) now set to 7.
[GUNMAN] Feature: (AGGRESSIVE NIL NIL) now set to 7.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL DONT-BE-CAUGHT))
now set to 3.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL EXECUTE-HOLDUP))
now set to 5.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL PROTECT-SELF)) now
set to 7.

   Your action failed because you missed your target.

   You hear a gunshot.
   The gunman aims his gun at you.
   The cashier waits.

PLAYER> again

Tick 4...
[#<GUNMAN, name:GUNMAN>] Distress from goal (SUBGOAL PROTECT-
SELF) decayed to 3.
[#<GUNMAN, name:GUNMAN>] Anger towards OFFICER (et al.) because
of negative influence on goal (SUBGOAL PROTECT-SELF) decayed to
6.
[#<GUNMAN, name:GUNMAN>] Anger towards OFFICER (et al.) because
of negative influence on goal (SUBGOAL PROTECT-SELF) decayed to
3.
[#<GUNMAN, name:GUNMAN>] Distress from goal (SUBGOAL EXECUTE-
HOLDUP) decayed to 3.
[#<GUNMAN, name:GUNMAN>] Hope of NIL causing goal (SUBGOAL EXE-
CUTE-HOLDUP) to succeed decayed to 2.
[#<GUNMAN, name:GUNMAN>] Anger towards OFFICER (et al.) because
of negative influence on goal (SUBGOAL EXECUTE-HOLDUP) decayed to
4.
[#<GUNMAN, name:GUNMAN>] Distress from goal (SUBGOAL DONT-BE-
CAUGHT) decayed to 2.
[#<GUNMAN, name:GUNMAN>] Anger towards OFFICER (et al.) because
of negative influence on goal (SUBGOAL DONT-BE-CAUGHT) decayed to
2.
[#<GUNMAN, name:GUNMAN>] Fear[a] of OFFICER increased by 9 be-
cause of threat to (SUBGOAL PROTECT-SELF).
[#<GUNMAN, name:GUNMAN>] Distress[a] increased by 3 because (SUB-
GOAL PROTECT-SELF) more likely to fail.
[#<GUNMAN, name:GUNMAN>] Anger[a] towards OFFICER increased by 9
because of negative influence on goal (SUBGOAL PROTECT-SELF).
[GUNMAN] Feature: (ANTICIPATION NIL NIL) now set to 2.
[GUNMAN] Feature: (GOOD-MOOD NIL NIL) now set to 0.
[GUNMAN] Feature: (BAD-MOOD NIL NIL) now set to 10.
[GUNMAN] Feature: (DEFENSIVE OFFICER NIL) now set to 9.
[GUNMAN] Feature: (UNFRIENDLY NIL NIL) now set to 4.
[GUNMAN] Feature: (ENERGY NIL NIL) now set to 7.
```

```
[GUNMAN] Feature: (AGGRESSIVE OFFICER NIL) now set to 8.
[GUNMAN] Feature: (DEFENSIVE OFFICER (SUBGOAL PROTECT-SELF)) now
set to 9.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL EXECUTE-HOLDUP))
now set to 4.
[GUNMAN] Feature: (AGGRESSIVE OFFICER (SUBGOAL PROTECT-SELF)) now
set to 8.
[GUNMAN] Feature: (DEFENSIVE NIL NIL) now set to 9.
[GUNMAN] Feature: (AGGRESSIVE NIL NIL) now set to 8.

   You shoot the gunman.
   You hear a gunshot.
   The gunman is now wounded.
   The gunman is now pale.
   The gunman is now bug-eyed.
   The gunman is now trembling.
   The gunman fires his gun at you.
   You hear a gunshot.
   The cashier waits.

******** You have died. ********
THANK-YOU-FOR-PLAYING-OZ
>
```

Bibliography

[Agre90] Agre, P. and Chapman, R. What are plans for? In *Robotics and Autonomous Systems*. Elsevier Science Publishers. 1990.

[Aristotle87b] Aristotle. Poetics. In *Introduction to Aristotle*. Translated by Richard Jenko. The Modern Library. New York. 1987.

[Bates92a] Bates, J., Loyall, A.B., Reilly, W.S. An Architecture for Action, Emotion, and Social Behavior. In *Proceedings of the Fourth European Workshop on Modeling Autonomous Agents in a Multi-Agent World*. S. Martino al Cimino, Italy. July 1992.

[Bates92b] Bates, J., Loyall, A.B., Reilly, W.S. Integrating Reactivity, Goals, and Emotion in a Broad Agent. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*. Bloomington, IN. July 1992.

[Bates92c] Bates, J. Virtual Reality, Art, and Entertainment. In *Presence: The Journal of Teleoperators and Virtual Environments*. Vol. 1. No. 1. MIT Press. Winter 1992.

[Bates91] Bates, J., Loyall, A.B., Reilly, W.S. Broad Agents. In *SIGART Bulletin*. Vol. 2. No. 4. August 1991.

[Beaudoin94] Beaudoin, L. Goal Processing in Autonomous Agents. Ph.D. Thesis. University of Birmingham, Birmingham, England. August 1994.

[Blank80] Blank, M. and Lebling, D. Zork I: The Great Underground Empire. Infocom. 1980.

[Blumberg94] Blumberg, B. Building Believable Animals. In *Working Notes of the AAAI Spring Symposium on Believable Agents*. Stanford, CA. March 1994.

[Brooks86] Brooks, R. A Robust Layered Control System for a Mobile Robot. In *IEEE Journal of Robotics and Automation*. Vol. 2. No. 1. IEEE Press. March 1986.

[Brooks91] Brooks, R. Intelligence Without Representation. In *Artificial Intelligence*. Vol. 47. 1991.

[Carbonell79] Carbonell, J. Computer Models of Human Personality Traits. Technical Report CMU-CS-79-154, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. November 1979.

[Cesta93] Cesta, A. and Miceli, M. In Search of Help. In *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*. Hidden Valley, PA. May 1993.

[Dyer83] Dyer, M. In-Depth Computer Understanding. MIT Press. Cambridge, MA. 1983.

[Elliott92] Elliot, C. The Affective Reasoner: A Process Model of Emotions in a Multi-agent System. Ph.D. Thesis. Technical Report No. 32, Institute for the Learning Sciences, Northwestern University. Evanston, IL. May 1992.

[Elliott93] Elliott, C. and Siegle, G. Variables Influencing the Intensity of Simulated Affectrive States. AAAI Technical Report SS-93-05. AAAI Press. Menlo Park, CA. 1993.

[Fikes72] Fikes, R. and Nilsson, N. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. In *Artificial Intelligence*. Vol. 2. 1972.

[Firby89] Firby, J. Adaptive Execution in Complex Dynamic Worlds. Ph.D. Thesis. Technical Report YALEU/CSD/RR #672, Yale University. New Haven, CT. January 1989.

[Forgy82] Forgy, C. Rete: A Fast Algortihm for the Many Pattern/Many Object Pattern Match Problem. In *Artificial Intelligence*. Vol. 19. September 1982.

[Frijda86] Frijda, N. The Emotions. Cambridge University Press. Cambridge, England. 1986.

[Frijda87] Frijda, N. and Swagerman, J. Can Computers Feel? Theory and Design of an Emotional System. In *Cognition and Emotion*. Vol. 1. No. 3. Lawrence Erlbaum Associates Limited. 1987.

[Gardner91] Gardner, J. The Art of Fiction. Vintage Books, New York, NY. 1991.

[Georgeff87] Georgeff, M. and Lansky, A. and Schoppers, M. Reasoning and Planning in Dynamic Domains: An Experiment with a Mobile Robot. Technical Report 380, Artificial Intelligence Center, SRI International. Menlo Park, CA. 1987.

[Hayes94] Hayes, P., Ford, K., and Agnew, N. On Babies and Bathwater: A Cautionary Tale. In *AI Magazine*. Vol. 15. No. 4. American Association for Artificial Intelligence. Winter 1994.

[Hatfield94] Hatfield, E. and Cacioppo, J. and Rapson, R. Emotional Contagion. Cambridge University Press. Cambridge, England. 1994.

[Hickman91] Hickman, S. and Shiels, M. Situated Action as a Basis for Cooperation. In *Decentralized AI 2, The Proceedings of the Second European Workshop on Modeling Autonomous Agents in a Multi-Agent World*. Y. Demazeau and J-P Muller, eds. Elsevier Science Publishers B.V., North Holland. 1991.

[Holmes86] Holmes, R. The Mystery of Edwin Drood. Vocal Score. Warner Bros. Publications. 1986.

[Horswill94] Horswill, I. Playing with Robots. In *Working Notes of the AAAI Spring Symposium on Believable Agents*. Stanford, CA. March 1994.

[Horton94] Horton, A. Writing the Character-Centered Screenplay. University of California Press. Berkeley, CA. 1994.

[Hovy88] Hovy, E. Generating Natural Language under Pragmatic Constraints. Lawrence Erlbaum Associates. Hillsdale, NJ. 1988.

[Huber95] Huber, M. and Durfee, E. Deciding When to Commit to Action During Observation-Based Coordination. In *Proceedings of the First International Conference on Multi-Agent Systems*. San Francisco, CA. MIT Press. 1995.

[Jones89] Jones, C. Chuck Amuck: The Life and Times of an Animated Cartoonist. Farrar, Straus & Giroux, New York. 1989.

[Kantrowitz96] Kantrowitz, M. Ph.D. Thesis. School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. Forthcoming 1996.

[Kass92] Kass, A., Burke, R., Blevis, E., and Williamson, M. The GuSS Project: Integrating Instruction and Practice through Guided Social Simulation. Technical Report #34, The Institute for the Learning Sciences, Northwestern University. Evanston, IL. September 1992.

[Kelso92] Kelso, M.T., Wehyrauch, P. and Bates, J. Dramatic Presence. In *PRESENCE: The Journal of Teleoperators and Virtual Environments*. Vol.2. No. 1. MIT Press. 1992.

[Laird87] Laird, J. Soar: An Architecture for General Intelligence. In *Artitifial Intelligence*. Vol. 33. 1987.

[Laurel90] Laurel, B. The Art of Human-Computer Interface Design. Addison-Wesley, Reading, MA. 1990.

[Lebowitz85] Lebowitz, M. Story-Telling as Planning and Learning. In *Poetics*. Vol. 14. No. 6. December 1985.

[Lebowitz84] Lebowitz, M. Creating Characters. In *Poetics*. Vol.13. No. 3. December 1984.

[Lesser95] Lesser. Distributed AI Report. *IJCAI Technical Session*. Montréal, Canada. 1995.

[Loyall96] Loyal, A.B. Believable Agents that Act and Generate Language. Ph.D. Thesis. School of Computer Science, Carnegie Mellon University. Pittsburgh, PA. Forthcoming 1996.

[Loyall93] Loyall, A.B. and Bates, J. Real-time Control of Animated Broad Agents. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society.* Boulder, CO. June 1993.

[Maes95] Maes, P. , Darrell, T., Blumberg, B., and Pentland, A. The ALIVE System: Wireless, Full-Body Interaction with Autonomous Agents. In *Communcations of the ACM, Special Issue on New Horizons of Commercial and Industrial AI*. Vol. 38. No. 1900. November 1995.

[Mataric92] Mataric, M.. Designing Emergent Behaviors: From Local Interactions to Collective Intelligence. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*. MIT Press. Cambridge, MA. 1992.

[McCloud93] McCloud, S. Understanding Comics. Tundra Publishers. Northampton, MA. 1993.

[Meehan76] Meehan, J. The Metanovel: Writing Stories by Computer. Research Report #74. Computer Science Department, Yale University. New Haven, CT. 1976.

[Moore89] Moore, D. and McCabe, G. Introduction to the Practice of Statistics. W.H. Freeman and Company, New York, NY.

1989.

[Mor95] Mor, Y., Goldman, C., and Rosenschein, J. Learn Your Opponent's Strategy (In Polynomial Time)! In *Working Notes of the IJCAI-95 Workshop on Adaptation and Learning in Multi-Agent Systems*. Montreal, Canada. 1995.

[Mueller90] Mueller, E. Daydreaming in Humans and Machines. Ablex Publishing Corporation. Norwood, NJ. 1990.

[Newell76] Newell, A. and Simon, H. Human Problem Solving. Prentice-Hall Book Company. Englewood Cliffs, NJ. 1973.

[Oatley92] Oatley, K. The Best Laid Schemes: A Psychology of Emotions. Cambridge University Press. Cambridge, England. 1992.

[Ortony88] Ortony, A., Clore, A, and Collins G. The Cognitive Structure of Emotions. Cambridge University Press. Cambridge, England. 1988.

[Pfeifer93] Pfeifer, R. On the Need to Study "Fungu Eaters." In *Working Notes of the Workshop on Architectures Underlying Motivation and Emotion*. Birmingham, England. August 1993.

[Rao95] Rao, A. and Georgeff, M. BDI Agents: From Theory to Practice. In *Proceedings of the First International Conference on Multi-Agent Systems*. San Francisco, CA. MIT Press. 1995.

[Read93] Read, T. and Sloman, A. The Technological Pitfalls of Studying Emotion. In *Working notes of the Workshop on Architectures Underlying Emotion*. Birmingham, England. August, 1993.

[Reilly94] Reilly, W. S. Synergistic Capabilites in Believable Agents. In *Working Notes of the AAAI Spring Symposium on Believable Agents*. Stanford University. March 1994.

[Schacter62] Schacter, S. and Singer, J. Cognitive, Social, and Physiological Determinants of Emotional State. In *Psychological Review*. Vol. 69. 1962.

[Schank77] Schank, R. and Abelson, R. Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures. L. Erlbaum Associates. Hillsdale, NJ. 1977.

[Sengers96] Sengers, P. Controlling Behaviors in a Schizophrenic Agent. Thesis Proposal. School of Computer Science, Carnegie Mellon University. Pittsburgh, PA. February 1996.

[Simon66] Simon, N. The Odd Couple. Random House. New York, NY. 1966.

[Simon67] Simon, H. Motivational and Emotional Controls of Cognition. In *Psychological Review*. Vol. 74. 1967.

[Sloan91] Sloan, S. Interactive Fiction, Virtual Realities, and the Reading-Writing Relationship. Ph.D. Thesis, The Ohio State University. Columbus, OH. 1991.

[Sloman94] Sloman, A. Explorations in Design Space. In *Proceedings of the 11th European Conference on Artificial Intelligence*. April 1994.

[Sloman86] Sloman, A. Motives, Mechanisms, and Emotions. Cognitive Studies Research Paper No. CSRP 062. University of Sussex, Brighton. 1986.

[Suchman88] Suchman, L. Plans and Situated Actions: The Problem of Human Machine Communication. Cambridge University Press. Cambridge, England. 1988.

[Sycara88] Sycara, K. Resolving Goal Conflicts via Negotiation. In *Proceedings of the Sixth National Conference on Artificial Intelligence*. St. Paul, MN. July 1988.

[Tambe95] Tambe, M., Johnson, W. L., Jones, R. M., Koss, F., Laird, J. E., Rosenbloom, P. S., and Schwamb, K. Intelligent Agents for Interactive Simulation Environments. In *AI Magazine*. Vol. 16. No. 1. 1995.

[Terzopoulus95] Terzopoulus, D. and Waters, K. Analysis and Synthesis of Facial Image Sequences using Physical and Anatomical Models. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 15. No. 6. 1993.

[Thomas81] Thomas, F. and Johnston, O. Disney Animation: The Illusion of Life. Abbeville Press, New York. 1981.

[Vidal95] Vidal, J. and Durfee, E. Recursive Agent Modeling Using Limited Rationality. In *Proceedings of the First International Conference on Multi-Agent Systems*. San Francisco, CA. MIT Press. 1995.

[Warner91] Warner, R.L. A Computational Model of Human Emotions. M.S. Thesis. Department of Computer Science, VPI & SU. Blacksburg, VA. 1991

[Wavish92] Wavish, P. Exploiting Emergent Behavior in Multi-Agent Systems. In *Decentralized AI 3, The Proceedings of the Third European Workshop on Modeling Autonomous Agents in a Multi-Agent World*. Y. Demazeau and E. Werner, eds. Elsevier Science Publishers B.V., North Holland. 1992.

[Webster92] Webster, C. Why Intelligent Systems should get Depressed Occasionally and Appropriately. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*. Bloomington, IN. 1992.

[Wehyrauch96] Wehyrauch, P. Guiding Interactive Drama. Ph.D. Thesis. School of Computer Science, Carnegie Mellon University. Pittsburgh, PA. Forthcoming 1996.

[Weizenbaum66] Weizenbaum, J. Eliza. In *Communications of the ACM*. Vol. 9. 1966.

[Wish76] Wish, M., Deutsch, M., and Kaplan, S. Perceived Dimensions of Interpersonal Relations. In *Journal of Personality and Social Psychology*. Vol. 33. No. 6. April 1976.

[Yamamoto94] Yamamoto, M. SOZZY: A Hormone-Driven Autonomous Vacuum Cleaner. In *SPIE Proceedings*. Vol. 2058. Boston, MA. 1994.