THE JOURNAL OF THE LEARNING SCIENCES, 5(4), 357–410
Copyright © 1996, Lawrence Erlbaum Associates, Inc.

Learning From Cases and Questions: The Socratic Case-Based Teaching Architecture

Daniel C. Edelson
Institute for the Learning Sciences and
School of Education and Social Policy
Northwestern University

The case-based teaching architecture provides a framework for computer-based learning environments that couple the benefits of active learning with learning from cases. A case-based teaching system consists of two interdependent components: a task environment that provides a learner with an engaging task, and a storyteller that monitors the learner's interactions with the task environment looking for opportunities to present instructive cases that will help the student learn from his or her situation. In Socratic case-based teaching, the task environment engages the learner in a dialogue by posing open-ended, thought-provoking questions. The development of Creanimate, a system designed to teach elementary school age students about animal adaptation, provided an opportunity to explore important research issues for the implementation of Socratic case-based teaching systems: dialogue management, indexing of cases in a computer memory, and reminding strategies for case presentation. Testing of Creanimate revealed patterns of use that illustrate the strengths and limitations of Creanimate as an implementation of the Socratic case-based teaching architecture.

The case-based teaching architecture[1] is a novel architecture for computer-based learning environments designed to enhance learning through the presentation of cases (Burke, 1993; Edelson, 1993; Schank, 1991). Building on theories from

---

Requests for reprints should be sent to Daniel C. Edelson, Institute for the Learning Sciences, Northwestern University, 1890 Maple Avenue, Evanston, IL 60201. E-mail: d-edelson@nwu.edu

[1] To distinguish the specific approach to computer-based learning environments presented in this article from teaching with cases in general, I refer to this approach as *the case-based teaching architecture* rather than the more general *case-based teaching.*

cognitive science and observations of effective teachers, the case-based teaching architecture takes advantage of artificial intelligence programming techniques and multimedia technologies. This article explores the case-based teaching architecture through a discussion of Creanimate, an early implementation of the case-based teaching architecture. Creanimate is an interactive learning environment that draws on a library of video clips to teach elementary school age students about animal adaptation. Creanimate is an example of a specific form of the case-based teaching architecture called the Socratic case-based teaching architecture. The Socratic case-based architecture uses thought-provoking questions to engage students. The primary research goal of the Creanimate program was to develop technologies for the implementation of the case-based teaching architecture and to demonstrate the ability of the architecture to engage learners. This research on the implementation of the case-based teaching architecture was designed to pave the way for future research on the validity of the theories of learning and understanding that underlie the architecture and its educational effectiveness.

This article is divided into seven sections. The first two sections introduce the case-based teaching architecture and the Creanimate system. The third and fourth sections describe the implementation of the central components of a case-based teaching system: the task environment and the storyteller. The final three sections place the case-based teaching architecture in the context of other research in the learning sciences; present the results of studies conducted with Creanimate in public school settings; and discuss the strengths, limitations, and future directions of this research.

CASE-BASED TEACHING ARCHITECTURE

Two pedagogical principles underlie the case-based teaching architecture:

1. Active learning: Effective learning takes place when students are engaged in the active pursuit of tasks that provide them with both motivation and opportunities for learning.
2. Learning from cases: When opportunities for learning arise, students should be provided with cases that will help them learn from their situations.

The principle of active learning demands that a student be actively engaged in a personally meaningful task. A meaningful task provides both a motivation for learning and a context for interpreting and retaining new knowledge. The principle of learning from cases recognizes that reasoning from cases is an important element of human reasoning, and that the acquisition of cases is a prerequisite for effective case-based reasoning. The case-based teaching architecture is based on the observation that a great deal of learning, both formal and informal, comes from hearing stories and cases. In the case-based teaching architecture, a student's active learning is augmented by just-in-time teaching in the form of relevant cases. When combined, cases provide a way to supplement the learning that a student can achieve through activity, and active learning provides a meaningful context for learning from cases.

A case-based teaching system supports these goals through two interdependent components, the *task environment* and the *storyteller.* The task environment provides a student with an engaging, motivating activity. The storyteller[2] monitors the task environment looking for opportunities to present cases that will help students learn from their interactions with the task environment. An effective task environment exposes students to a variety of situations that provide valuable experience with the subject matter. An effective storyteller is able to recognize opportunities for learning that arise in the course of a student's interactions with the task environment and to capitalize on those opportunities by presenting cases that will help the student learn from his or her situation. To achieve this goal, a storyteller must have a sufficiently wide range of cases to cover the opportunities for learning that may arise in the task environment. In addition, these cases must be indexed in such a way that the storyteller can locate appropriate cases to present to a student based on its observations of the student's actions.

In this architecture, the task environment helps a student understand the underlying structure of the domain, and the storyteller provides the student with cases to support that structure. These two forms of knowledge complement each other. The framework for the domain that the student learns from the task environment and the cases that the student gains from the storyteller mutually reinforce each other in the student's memory.

In practice, the construction of case-based teaching systems is both an art and a science. Creating engaging task environments that are motivating, challenging, and interesting is in many ways an art, and developing systems that can present cases to help students learn draws on the science of artificial intelligence. When humans teach with cases and stories they rely on two critical skills: the ability to recognize an opportunity to present a case or a story and the ability to identify the right one to tell to suit that opportunity. Reproducing these two important human qualities is the primary technical challenge of case-based teaching.

Why Teach With Cases?

Parents and teachers often tell stories to convey lessons. Teaching with stories and learning from them occurs naturally and unconsciously. Stories are but one form that cases take in human learning. Professional schools, such as business, law, and

---

[2] Because of the similarity between the retrieval and presentation of cases in a case-based teaching system and human storytelling, the module that performs this process is called a storyteller.

medical schools, have employed cases in their teaching for decades with approaches known as the case method (McNair, 1954; Redlich, 1914; Reed, 1921), case-based instruction (Williams, 1992), and problem-based learning (Barrows, 1986; Barrows & Tamblyn, 1980). For the purposes of this article, a case is a description of a specific incident or observation that can be used to convey one or more lessons. Cases are useful in instruction to the extent that they can provide a learner with material for drawing generalizations and with precedents that are useful for reasoning about novel situations. Traditionally, cases have been conveyed orally and in writing, but multimedia technologies now provide for case presentation through video, graphics, and animation.

Support for teaching with cases comes from the theory of case-based reasoning (Kolodner, 1993; Riesbeck & Schank, 1989; Schank, 1982). This theory argues that many situations are too complex for people to deal with by reasoning from first principles. Instead, they reason using previously stored cases. Researchers have observed evidence of people using case-based reasoning in practical situations in a wide variety of problem-solving domains (Klein & Calderwood, 1988; Kolodner, 1991; Ross, 1989). Because complex subjects require case-based reasoning, it is important to teach those subjects in a way that will assist the natural process of case-based reasoning. Although the prevalence of the case method in professional education reflects this recognition, much of K–12 and undergraduate education does not.

To be an effective case-based reasoner in a particular domain, an individual must have a library of cases to draw on and enough understanding of the domain to evaluate a case's relevance to specific situations. Thus, to support case-based reasoning, it is important to help students acquire cases with which to reason. Students can effectively gather cases through either first- or secondhand experience. In schools, firsthand experience has traditionally been provided by laboratories, demonstrations, role playing, simulations, and other hands-on activities. These experiences are important for helping students develop personal case libraries. However, firsthand experience is not always feasible or even advisable, so cases can be provided secondhand through storytelling or other forms of presentation.

Cases in Context

In order to perform case-based reasoning effectively, a person must possess a library of cases that is large enough to cover an adequate variety of situations. The accumulation of such a library is one of the goals of human learning in natural settings. However, having a large library with wide coverage is not sufficient in and of itself. It is also necessary to have the case library organized, or indexed, so that appropriate cases can be retrieved at the moment when they are most useful. According to theories of case-based reasoning, cases in memory are organized according to features that describe situations in which the case may be useful. The collection of features that label a particular case is called an *index.* A case library is only as useful as the indexes that organize the cases. If cases are stored under irrelevant or incorrect features, the reasoner will not be able to retrieve those cases when they might be useful in the future.

Because of the need for an appropriately indexed case library, case-based teaching must attend to the issue of how students develop appropriate indexes for the cases in their memories. When people learn from experience, they pay attention to the circumstances in which an experience occurs. This context helps individuals store experiences as cases in their memories. The context provides learners with features that they can use to index cases. When they find themselves in situations that have features that match the context of the earlier experiences, they are able to retrieve the earlier case to help decide how to act in the new situations. When teaching with cases, it is important to convey them to a learner in a context that provides the learner with features with which to index that case. Effective case-based teaching takes advantage of context to help the learner index cases effectively in his or her own memory so that they can be retrieved when they might be useful for dealing with novel situations.

CREANIMATE: A CASE-BASED TEACHING SYSTEM

Creanimate is a case-based teaching system that teaches children about biology. It was conceived partially in response to the dry way in which science is often taught in schools. Therefore, an important motivation behind the Creanimate project always was the desire to create a science learning experience that would both be fun and allow learners to engage their imaginations. From a pedagogical standpoint, Creanimate has three primary goals: (a) provide students with an engaging interaction, (b) teach the conceptual relations that are fundamental to understanding animal adaptation, and (c) help students develop their own personal case libraries of animal adaptation, organized in a way that will help them to use cases to reason about other animals.

An Overview

The subject matter of Creanimate is animal adaptation. Specifically, the program focuses on the relations between the physical features of animals, the way they behave in the wild, and how they survive in nature. The central element of the program is a dialogue between the student and the computer about an animal that the student would like to create. Hence, the name *CreANIMate,* obtained by combining the words *create* and *animal.* A session begins with an invitation to the student to create his or her own animal by taking an existing animal and changing it in some way. Following the choice of an animal, Creanimate engages the learner in a series of natural language dialogues in which the student considers the ramifications of the proposed modification for his or her animal. For example, a student might ask for a dog with a big nose or a bear that can dance. In response, Creanimate would engage the student in a discussion of how the dog will use its big nose or what good it might do a bear to dance. Creating a new animal was selected as the central task because it offers children the opportunity to be imaginative. This premise has been used by a number of others in both commercial and experimental applications that allow users to create their own animals (Coderre, 1988; Karakotsios, 1992; Resnick, 1991; Travers, 1988). Inventing a new animal is compelling to children because it rewards their natural inclination to push beyond the limits of the world around them.

Following an instructional tradition dating back to Socrates, Creanimate helps students learn from their animals by posing thought-provoking questions. It raises questions about the students' animals and discusses possible answers with them. For example, if a student were to ask for a fish with wings, the program would respond by asking how the student's fish will use its wings. In the ensuing discussion, the student could propose answers (e.g., to fly) or ask the program to suggest answers (e.g., to dance, to keep cool). Because the answers to Creanimate's questions usually entail making additional changes to a student's animal, they typically give rise to new questions. As a result, the student and the computer pursue an ongoing dialogue in which the student proposes a modification to an animal, Creanimate raises questions about that modification, the student resolves those questions by making new modifications, and the system raises new questions about the new modifications. For example, after considering several possibilities, a student might decide that his or her fish should use its wings to help it fly. The addition of flying to a fish raises new questions for the student to consider, such as what other modifications the fish would require in order to fly and how flying might help the fish survive.

To help the student in the consideration of answers to its questions, Creanimate presents video clips of animals in the wild. These video clips present cases of animal adaptations. In presenting these cases, Creanimate capitalizes on the powerful impact of video. Nature is full of dramatic and surprising animal adaptations, and no medium captures this better than film. In the course of answering questions about their animals, students see dramatic video clips that show concrete examples of the answers to those questions. For every answer discussed, Creanimate attempts to locate relevant video clips from its library. For example, in one session, a student who asked for a bee with a big nose answered the question "What will it use its big nose for?" with, "so it can suck up honey." Creanimate responded:

> I think that might work. Elephants use their big noses to suck up liquids. I have an interesting video about that. If you like baby elephants, then you'll love this video. Would you like to see that?

The student then saw a video clip showing a young elephant and its mother using their trunks to drink and to bathe themselves. In this interaction, Creanimate showed a video clip to confirm that a student's answer could be supported by an example of animal behavior in the wild. In other situations, Creanimate shows video cases for other reasons, such as to suggest answers to questions, to show interesting similarities, and to show surprising exceptions to general rules. Creanimate helps students to resolve questions about animal adaptation by presenting them with vivid cases. These memorable cases are designed to help students develop the ability to apply the natural process of case-based reasoning for thinking about animal adaptation.

Creanimate in Action

To demonstrate the nature of an interaction with Creanimate, this section presents a transcript recorded during the session of a fourth-grade girl using Creanimate in a trial involving 23 students at a Chicago-area public school.

Creanimate has a colorful interface designed to allow students to express themselves through a combination of mouse clicks and typing (Figure 1). The top portion of the screen contains the program's output. The text in this section may include explanations, invitations to view video, and questions for students. The middle section of the screen is used for student input. Students respond to the system by clicking on the options provided in this section. Students may enter their own text by selecting an option containing an ellipsis and completing the phrase with their own text. The picture at the lower right shows the animal that the student is creating. When Creanimate offers a student the opportunity to see a video, the picture of the student's animal is replaced by a single "preview" frame from the clip. Along the bottom edge of the screen are user control buttons that allow students to exert control over their interactions by skipping ahead, backing up, obtaining additional information, or starting over with a different animal.

Choosing an animal. The first step in the program is selecting an animal. The student is presented with a list of animals and modifications. These options were selected based on consideration of the interests of children, a desire for diversity, and the availability of video. A student may request an animal or modification that is not on the list of options by clicking on "A ... " or "Something else" instead of the listed animals or modifications and entering the answers in free text. Ideally, there would be many more choices available or simply free entry. However, with only 1 hr of video available in early versions of the system, it was necessary to provide choices for which the system was guaranteed to have interesting video. With a larger knowledge base and video library, the choices could be increased or eliminated.

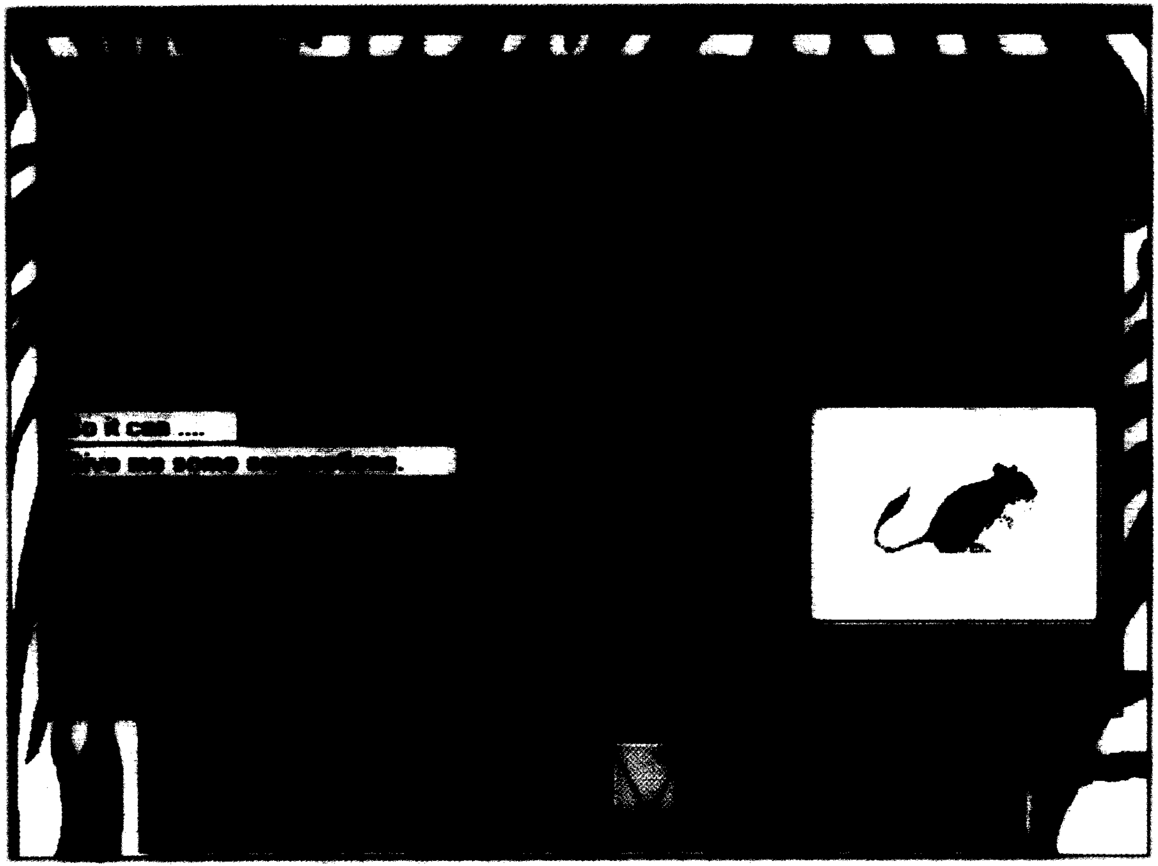

FIGURE 1 A sample screen from the Creanimate program.

If you could create a new animal by taking an existing animal and changing it some way, what would you make? (Pick one animal from the list on the left and one change from the list on the right.)

A Fish (pike)
A Frog
A Bear
A Bee
A Butterfly
A Dog
A Bird (robin)
A ...

With a big nose.
With a beak.
With wings.
That can dance.
That can hide.
That can hunt.
That can fight.
Something else.[3]

The student selected "A ... " from the list on the left, typed in "monkey," then selected "With wings" from the list on the right.

Student: **A *monkey* with wings**[4]

If the student selects an animal from the list provided, the program shows the student a large picture of her unmodified animal in a box labeled "Before" with a picture of a question mark in a box labeled "After." This establishes the forthcoming reward of seeing what the modified animal might look like.

An incidental reminding. The system's case library includes a video showing a surprising animal that is similar to the animal the student wants to create. Therefore, it gives the student the option of seeing a video about it. In this case, the storyteller was "reminded" of the fox bat, another mammal with wings. Incidental remindings like this one are designed to challenge a learner's generalizations and broaden his or her experience of the exceptions to general rules.

Actually there are mammals that have wings. For example, fox bats have leathery wings.
Is that something you would like to see?
Student: **No**

---

[3] The transcript was edited for brevity. Deletions are indicated by ellipses. In many cases, choices presented to the student were omitted. Except where noted, all inputs and outputs in the transcripts in this article appear exactly as they appeared on the screen.

[4] In this transcript and those that follow, input selected by students using the mouse appears in boldface; input that was typed by students appears in bold italics. The remainder was input by selecting an option on the screen with the mouse.

Posing a question. The system immediately responds to the student's proposal by posing a question about that animal.

> If your monkey is going to have wings, that should help it to do something.
> Why would you like your monkey to have wings?
> So it can ...
> Give me some suggestions.
> Student: **So it can** ***fly away from its enemys***

The dialogue manager selected the question "Why have wings?" after determining that the system's video library includes clips that show ways animals use wings. The questions that the system asks are intended to help students learn the basic relations that underlie animal behavior. In this case, the question emphasizes the connection between the features an animal has and the actions the feature enables it to perform.

Example cases. The computer determines from its knowledge base that flying is a reason for having wings and locates the best video in its library to show wings used for flight. This example case is used as a way of confirming the student's answer and providing case-based reinforcement for the knowledge in the student's memory. Because the program is only prepared to talk about basic actions (e.g., flying, smelling, biting), in answer to this particular question, the dialogue manager ignores the student's mention of getting away from enemies. The program will return later to the higher level question of why flying might help the monkey survive if the student commits to flying as a reason for her monkey to have wings. The student asks to watch the video.

> That's a good idea. White breasted sea eagles use their feathered wings to help them fly.
> Do you want to see that ?
> Student: **Yes**
> *[Video: "Sea Eagle Catches Fish"]*

This particular video shows a dramatic scene of a white-breasted sea eagle pouncing on a fish nearly as big as itself and carrying it away. Because the case library contains several videos about wings being used to fly, the storyteller offers another one to the student. She chooses to view it. As soon as the student refuses a video on a particular subject, the dialogue manager moves on to the next step in the dialogue.

> White breasted sea eagles are not the only animals that use their wings in order to fly. For example, black skimmers have wings to help them to fly.

I have a great video about that.
Would you like to see that?
  Yes.
  What other videos can I see?
  No.
Student: **Yes**
[Video: "Skimmer Skims"]
...

Committing to an answer. Once the student has seen examples that illustrate her answer for how her monkey might use its wings, she has the opportunity to commit to that answer. If she commits, she will go on to other questions about her monkey with wings. Otherwise, she can consider other uses for wings. Because flying is the most obvious use for wings, the portion of the dialogue up to this point does little more than reinforce knowledge the student already has. When students go on to consider other uses for wings that Creanimate knows about, such as keeping cool, swimming, and performing mating dances, the opportunities for learning become much richer.

So, your monkey could have wings to fly. Why would you like your monkey to have wings?
  To fly
  Show me more reasons.
Student: **To fly**
Ok, your monkey will have wings so that it can fly.

If the student had selected one of the seven animals that the program originally suggested, she would at this point see a predrawn picture of her animal with its wings. Instead, she sees a drawing of a pile of bones in a workshop next to a sign reading "Under Construction."

A second question. Now that the student has decided what her monkey will do with its wings, the program raises a new question: What other changes will be required for it to be able to fly? This question emphasizes the fact that many actions require a combination of different physical attributes.

In order to fly, animals must be able to keep themselves in the air. Let's talk about how your monkey will keep itself up in the air.
Having wings is not enough for your monkey to keep itself up in the air. It needs more.

> How else do you want to change your monkey so it can keep itself up in the air?
> Give it ...
> Make it ...
> Give me some suggestions.
> Student: **Give me some suggestions**

Rather than answer the question, the student asks the program to give her some suggestions.

Cases to provide suggestions. When students ask for suggestions, the program assumes that they are doing so because they do not know any answers to the current question. This constitutes an opportunity for learning. Rather than simply tell the student an answer, Creanimate provides the answer through a case. In this example, it proposes larger wings. Because of the structure of Creanimate's knowledge base, it treats combinations of features that enable an animal to achieve an action as aggregate features. Creanimate's knowledge base contains several combinations of animal size and wing size that enable flying, such as small animals with small or medium-size wings, and medium-size animals with large wings.[5] (Monkeys fall into Creanimate's category of medium-size animals.)

> Large wings and medium size can work together to help an animal to keep itself up in the air.
> Your monkey already has medium size.
> Swainson hawks use their large wings and medium size to help them keep themselves in the air.
> I have a good video about that. They had to be up in the air in order to film this video.
> Is that something you would like to see?
> Student: **Yes**
> *[Video: "Swainson Hawk Hunts"]*
> ... *[Additional video deleted]* ...

---

[5] Currently, Creanimate's knowledge representation contains no way of representing the negative relation between size or weight and flight, or the manner in which wing size can compensate for overall size in enabling flight. All aggregate features are treated as if the subcomponents of the features work together toward the enablement of an action. Therefore, the program proposes large wings to the student in an admittedly awkward manner, by proposing the combined feature of large wings and medium size, as if size contributed to, rather than conflicted with, flight. Limitations such as these in Creanimate's knowledge representation, their impact on the system's effectiveness, and proposed solutions are discussed in the article's final section.

What would you like to add to your monkey so it can keeps itself up in the air?
Large wings
Show me more ways.
Student: **Large wings**
OK, your monkey will use its large wings to help it keep itself up in the air.
...

In a discussion that was omitted from this transcript, the student considers how her monkey will steer in flight.

Similarity in case retrieval. Next Creanimate moves on to the question of how flying will help the student's monkey survive.

Let's see if we can find a reason for your monkey to fly. Can you think of a reason for your monkey to fly?
Student: **So it can *get away from its enemys***
I did not understand that. Try picking one of the choices below.
Chase its mate
Eat
Fight for its young
Grasp vegetation
Migrate
Pounce on its prey
Pursue its prey
Run away from its enemies
Remove competition for food
Remove competition for their mates
Drink
Fight for its mate
Flee predators
Ingest food
Migrate from cold
Process food internally
Scare predators
Search for its prey
Steal food
Student: **flee predators**
That's a good idea. Monkeys could fly to help them to flee predators.
There are other animals that flee predators, only they don't fly. Frogs are one example. They leap so they can flee predators.
Would you like to see a video about that?
...

In this interchange, the storyteller was unable to locate a video about flying to flee predators, but it was able to locate a video about frogs that leap to flee predators and offered that instead. The ability to use this sort of similarity in case retrieval enables the Creanimate storyteller to get maximum utility out of the cases in its library.

Eventually, Creanimate exhausts its questions about flying, but the student chooses to stick with her monkey and add a new characteristic.

> That's all I can think of for now. You can either add something else to your monkey or try another animal.
> Student: **Add something else to my monkey**
> ...
> How would you like to change your monkey?
> Student: **So it can fight**
> ...

In the remaining portion of the dialogue, the student considers two more issues: why her monkey will fight and how it will fight. She resolves the former issue by having it fight for its mate, and the latter by having it kick other animals.

In this 45-min dialogue, the student considered six questions about her animal and viewed seven videos. She was unusual in the length of time she devoted to one animal. Most of the 23 fourth graders in the study in which she participated chose to switch animals after one or two questions, and most saw slightly more videos (7.8 videos per session on average).

The Status of Creanimate

Creanimate is a working prototype that has been tested successfully in a variety of settings. The most recent version contains 140 indexed video clips totaling 60 min. It conducts dialogues about six different types of questions, each focusing on a different relation between the physical features, actions, and behaviors of animals. The knowledge base includes more than 200 animals and more than 1,000 different animal characteristics. Included in this knowledge base is the information that allows Creanimate to construct natural English sentences and to understand students' typed input.[6] Tests of the program indicate that students can use the current prototype for at least 3 hr without exhausting its capabilities. Although it has received considerable use, Creanimate remains an experimental prototype.

THE TASK ENVIRONMENT

An effective task environment gives a student the opportunity to learn in the course of pursuing goals that are natural to him or her. Simulation, design, problem-solving, and diagnosis environments are all possibilities for effective task environments. Four criteria for effective task environments are (Edelson, 1993):

---

[6] The natural language parser is based on the Direct Memory Access Parser (DMAP) architecture (Martin, 1989, 1990). It is described in Fitzgerald (1992, 1995).

1. Opportunities to form and explore hypotheses: Active learning demands that students have the opportunity to learn by building their own understanding and refining it in response to experience. A task environment must give the student chances to form, express, test, and revise hypotheses. In Creanimate, the student's choice of an animal to create serves the role of expressing a hypothesis. Students explore their hypotheses by considering questions about how their hypothetical animals might survive.
2. Opportunities for learning: The task environment must be constructed in a way such that students experience opportunities to extend their understanding as a result of interacting with it. Opportunities for learning often arise in the form of expectation failures (Schank, 1982). In Creanimate, opportunities for learning arise from the questions posed by the system and the responses offered by students. Creanimate's questions can point out gaps in students' understanding or offer opportunities to enhance their existing understanding with novel cases.
3. Authenticity: A task environment must be authentic both to the student and to the world. It must be authentic to the student in that it must interact with the student's personal goals. It must be authentic to the world in that the knowledge and skills that a student gains through experience with the environment must be encountered in a context that is consistent with the way they will be encountered and employed in the real world (Brown, Collins, & Duguid, 1989; Collins, Brown, & Newman, 1989; Schoenfeld, 1992). Authenticity to the student's goals insures that the student will have the motivation to engage with the environment and to learn. Authenticity to the world increases the likelihood of a student's being able to apply what he or she learns in settings that arise naturally. Creanimate's authenticity to the student comes from the nature of the task. Children find the opportunity to invent new animals naturally engaging. Creanimate's authenticity to the world comes from the questions that they consider. These questions are authentic to explaining how an animal is adapted to its environment.
4. Motivation: The key to active learning is appropriate motivation. To be motivating, a task environment should provide sufficient challenge, curiosity, control, fantasy, satisfaction, and relevance (Keller, 1983; Lepper & Hodell, 1989; Malone, 1981). Creanimate attempts to motivate students through the inventive nature of the task, the challenge of the questions the system poses, its responsiveness to the student, and the drama of the video clips.

Any specific task environment must address these criteria in a way that is appropriate for the subject matter it is designed to teach, for the learners who will use it, and for the environment in which they will be using it. In Creanimate, the task of designing a new animal was selected to be engaging for the target age group. The central activity, a dialogue that focuses on open-ended questions, was selected to suit the nature of the subject matter and the pedagogical goals of the system, as well as to provide ample opportunities for learning from cases.

The Creanimate Task Environment: Socratic Case-Based Teaching

As the transcript presented earlier illustrates, questioning is at the heart of Creanimate's dialogues. In this respect, it follows in the tradition of two early intelligent tutoring systems (ITSs), SCHOLAR (Carbonell, 1970a, 1970b) and WHY (Stevens & Collins, 1977). Because it conducts question-and-answer dialogues with students, Creanimate's variant of the case-based teaching architecture is called the Socratic case-based teaching architecture. Although the dialogue manager does not employ all of the techniques associated with Socratic-style teaching (Clancy, 1987; Collins & Stevens, 1982), it adheres to the Socratic tradition of encouraging a student to make hypotheses and then leading the student through an exploration of those hypotheses with thought-provoking questions.[7] The dialogue manager's responses to a student's proposed animal capitalize on the student's investment in the animal to introduce the student to the issues involved in animal adaptation. A student's interest in his or her own animal serves as a natural motivator to engage in the process of answering the explanation questions that apply to the animal. These explanation questions also provide a context for presenting example cases that supply appropriate answers.

Teaching with questions. Creanimate uses questions to help establish a context for learning from cases. In their attempt to elicit curiosity on the part of the learner, these questions provide an important motivational context for learning. In addition, they provide important information to assist a student in developing a representational framework for the subject matter and in indexing the information in cases.

Questions are central to expertise and to learning. Knowing the right questions to ask is an important element of expertise because it permits the expert to deal with unfamiliar situations. Explanation, the central process of understanding, alternates between asking questions and constructing plausible answers. With respect to learners, Chi, Bassok, Lewis, Reimann, and Glaser's (1989) studies of students engaged in self-teaching identified questioning as a skill that distinguishes between good and poor learners in such self-teaching situations. But where do these questions for understanding come from? The questions that one asks in the process of explanation in any domain derive from knowledge of the basic relations that underlie that domain. These questions, called *explanation questions* in the terminology of Schank (1986), express the relations that tie together concepts in the

---

[7] The use of the name *Socratic case-based teaching* is supposed to reflect the spirit of Socratic teaching, not the exact technique employed by Socrates. In true Socratic teaching, the instructor poses questions but refrains from providing information.

domain. The explanation questions for a field are important because they express the relations with explanatory power in that field. Someone who understands the explanation questions for a particular subject will use the relations they express to organize his or her knowledge about that subject. Therefore, teaching students the explanation questions in a domain is the same thing as teaching them the important relations that underlie that domain.

Once learners understand the important relations that tie concepts together in a domain, they can use those relations as the basis for indexing cases in their memories. Using this framework, they can further extend their understanding by both asking explanation questions and organizing their observations according to the relations that the questions express. Presenting cases to learners in the context of a discussion of explanation questions provides students with motivation, an indexing framework, and some of the important features that will enable them to index those cases appropriately in their own memories.

Explanation questions for animal adaptation. Teaching the explanation questions underlying animal adaptation is a central pedagogical goal of Creanimate. Although an individual can understand the relations within this subject area at ever-increasing levels of sophistication, the Creanimate project has identified five explanation question categories that support an introductory-level understanding of animal adaptation:

1. WHY FEATURE? The *Why feature?* explanation question attempts to connect an animal's physical feature with some action that the feature supports. For example, "Why do cheetahs have long legs?"
2. HOW ACTION? The *How action?* explanation question explores the same relation as *Why feature?* However, *How action?* takes an action performed by an animal and looks for the features that are necessary to perform that action. For example, "What does an eagle have to help it to fly?"
3. WHY ACTION? The *Why action?* explanation question attempts to connect an action performed by an animal with some survival behavior. It assumes that the actions that animals perform enable them to survive in some way. For example, "Why do female alligators dig holes?"
4. HOW BEHAVIOR? The *How behavior?* explanation question is the reverse of *Why action?* It starts with a survival behavior and seeks actions that enable an animal to achieve that behavior. For example, "How do chimpanzees get food?"
5. WHY BEHAVIOR? Behaviors can be performed for more than one reason. For instance, finding shelter can be in the service of both keeping warm and avoiding danger. Therefore, the fifth category of explanation question is *Why behavior?* For example, "Why do bears fight?"

These five explanation question categories correspond to the relations among three different conceptual categories: features, actions, and behaviors. These basic concepts and the relations between them can provide a student with a solid groundwork to understand animal adaptation. The goal of the Socratic dialogues in Creanimate is to ask questions in a way that will help students develop an understanding of adaptation according to these relations.[8] In the construction of the Creanimate system, these five explanation questions and three conceptual categories formed the basis for the indexing vocabulary and reminding strategies of the storyteller as well as the knowledge representation used by the dialogue manager.

The concern over fostering misconceptions about evolution is further complicated in Creanimate by the design goal of using everyday language and avoiding stilted constructions. For example, in the extended transcript in the previous section, the dialogue manager says, "White breasted sea eagles are not the only animals that use their wings in order to fly. For example, black skimmers have wings to help them to fly." Stating that "black skimmers have wings to help them to fly" could imply that black skimmers evolved wings in order to fly. However, this wording is characteristic of the way people discuss adaptation in ordinary conversation and was chosen for its naturalness over other more accurate but less natural-sounding constructions.

The Dialogue Cycle

The Creanimate task environment employs explanation questions in a Socratic-style dialogue with the goal of achieving the learning benefits previously described. These dialogues are implemented through the *dialogue cycle.* The dialogue cycle provides a framework for the discussions between student and computer in Creanimate.

The dialogue cycle in Creanimate has its origins in an experiment conducted in 1987. In this "Wizard of Oz" test, seventeen 11- and 12-year-old children were told

---

[8] One of the risks of Creanimate's approach to teaching animal adaptation is that it may lead students to develop a view of evolution that is inconsistent with the accepted scientific view, that is, they may develop intentional or Lamarckian views of evolution. Creanimate is built around the assumption that a view of animals as being adapted to survive in their environment is an important step toward understanding evolution, the process by which the adaptations developed. Creanimate is designed to teach about the adaptations of animals without focusing on the evolutionary process. As such, its dialogue is based on the assumption that the attributes that animals possess all serve a survival function. Although it is not accurate to say that features are developed through evolution in order to serve particular functions, it is fair to generalize that, as a result of evolution, animals have features and perform actions that serve particular survival functions. It is an assumption of Creanimate, albeit an untested one, that establishing a strong grounding in adaptation is an appropriate first step toward developing a sophisticated understanding of the evolutionary process.

that they were testing an unfinished computer program. However, instead of using a program, the children were conversing through a terminal with a professor of biology[9] at a terminal in an adjoining room. The goal of this experiment was to gather information about how children would respond to a program like Creanimate and what they might say. In addition, the conversations provided a human model for Creanimate to emulate. An analysis of the transcripts from this experiment revealed that most of the conversations followed a simple, but effective, pattern. This conversational pattern consisted of (a) the student asking for a modification, (b) the professor asking a question about that modification, (c) the student proposing an answer to the question or asking the professor for assistance, and (d) the professor presenting a case that related to or provided an answer to the professor's question. The dialogue cycle in Creanimate was developed using this pattern as a model. The steps in Creanimate's dialogue cycle are shown here with examples drawn from a student transcript:

1. Pose explanation question:
   That's a good idea, is there a reason you want your frog to fight?
2. Get answer from student or knowledge base:
   Student: **So it can protect its self**
3. Present example case (if available):
   That's a good idea. Bees protect themselves by fighting enemies. Would you like to see that?
4. Allow student to commit to the answer:
   So, your frog can fight to protect itself.
   You can commit to that now, or we can look at some other reasons why animals fight and you can decide later.
   Why would you like your frog to fight?
      To protect itself
      Show me more reasons
   Student: **To protect itself**

These four steps are repeated as the student proposes or the dialogue manager suggests different answers. Once the student commits to an answer (Step 4), the current dialogue terminates, and a new dialogue is initiated. Because of the variety of places in the cycle that the storyteller can intervene and the range of responses that students can make, the four steps of the dialogue cycle shown are simply the skeleton of a dialogue. Actual dialogues are more elaborate and varied than the simple dialogue cycle. For example, if a student provides an answer to a question

---

[9] The experiment described in this section was conducted by William Purves at Harvey Mudd College. Additional transcripts from these experiments appear in Edelson (1993).

that is inconsistent with the dialogue manager's knowledge base, the system offers the student the opportunity to digress from the discussion of the current explanation question in order to see video clips of his or her answer. This *unverifiable answer* reminding strategy is discussed in the next section.

The dialogue cycle is managed by the *Socratic Dialogue Manager.* This dialogue manager is supported by a large knowledge base about animal adaptation. The knowledge base includes information for natural language understanding and generation, so the dialogue manager is able to translate responses typed by students into an internal representation (Edelson, 1993; Fitzgerald, 1995) and then evaluate the correctness of a student's answer with respect to the program's own knowledge base. Using this evaluation, the dialogue manager generates an appropriate response to the student in informal English by using the natural language generation information in its knowledge base to complete template sentences (Edelson, 1993). In practice, this simple dialogue cycle gives rise to a very wide range of student interactions. The dialogue cycle enables students to formulate their own hypotheses in response to open-ended explanation questions and provides opportunities for them to learn from relevant cases that respond to these hypotheses.

THE STORYTELLER: INDEXING AND REMINDING

The storyteller is the companion to the task environment in a case-based teaching system. The goal of the storyteller in a case-based teaching system is to harness the reminding process to help students learn from their interactions with the task environment. To retrieve cases when they are appropriate, the storyteller must have a well-organized case library and strategies for searching that library that suit the particular opportunities for learning that occur in the accompanying task environment. The central issues for the design and construction of a storyteller are as follows:

- Case selection: As a fundamental requirement, a storyteller must have a sufficiently large library of cases to cover the range of opportunities for learning that can occur in the task environment.
- Communication with the task environment: A storyteller must have a means of bidirectional communication with the task environment that will allow it to effectively monitor the student's interactions and to intervene when it sees the opportunity to present a case.
- Indexing scheme: An indexing scheme is a method for labeling cases in memory so that the storyteller can recognize situations in which they are relevant. To be effective, the indexing vocabulary must be sufficiently expressive to cover both the range of cases in the storyteller's library and the range of opportunities for learning from cases that occur in the task environment.

- Reminding strategies: Reminding strategies are algorithms used by a storyteller to search its case library and locate appropriate cases to present. The input to a reminding strategy is a description of a situation in the task environment, and the output is one or more cases that are relevant to that situation. Reminding strategies must be able to search the case library in a computationally efficient fashion in order to retrieve stories in a timely manner.
- Presentation of cases: Once the storyteller has identified a case to present, it must be able to present it in a way that engages the student and helps to make clear to him or her the relevance of the case to the current situation.

In any case-based teaching system, the resolution of these five issues is partially determined by the design of the task environment and the character of the cases that make up the storyteller's library. Thus, in the Creanimate system, the storyteller was designed to suit a task environment that conducts Socratic-style dialogues. Of the issues just described, the two that were the primary foci of the Creanimate researcher effort were indexing and reminding.

Reminding Strategies

The reminding strategies for any particular task environment must correspond to the types of opportunities for learning that arise in the task environment. Pedagogical reminding strategies include correcting misconceptions, giving advice, providing warnings, and presenting examples. Although there are undoubtedly general-purpose pedagogical reminding strategies, specific settings and pedagogical goals require special-purpose reminding strategies. For example, Burke and Kass (1992) described a system that uses a social simulation to teach salespeople the skills necessary to sell products and services (Kass, Burke, Blevis, & Williamson, 1993–1994). Their strategies are specialized for a system that teaches a student how to construct and execute plans. Burke and Kass (1992) pointed out that their strategies differed from the reminding strategies in Creanimate because the reminding strategies in Creanimate focus on the ramifications of design decisions, whereas those described by Burke and Kass focus on the process by which an individual makes decisions.

Creanimate employs several reminding strategies, each of which serves a specific pedagogical objective. Each strategy places particular demands on the information available in the indexes that label cases in the program's memory. Creanimate's reminding strategies fall into two categories, *example remindings* and *incidental remindings*. Example remindings display examples of the issues that arise in the course of a discussion. Incidental remindings present cases that relate tangentially to the current focus of the discussion. The specific types of example and incidental remindings employed by the Creanimate storyteller are displayed in Table 1.

TABLE 1
The Reminding Strategies in Creanimate

Example Remindings	*Incidental Remindings*
Correct answer remindings	Similarity-based remindings
Unverifiable answer remindings	Expectation-violation remindings
Suggest answer remindings	

Example remindings provide students with concrete examples of the abstract principles that arise in the course of dialogues. These examples are intended to enlarge a student's personal case library of animal adaptations. Example remindings reinforce the structure of the domain provided by the dialogue manager's explanation questions. Incidental remindings are designed to broaden students' exposure and increase their interest. They are called *incidental* because they digress from the central point of the discussion in order to present interesting sidelights. These sorts of digressions are, in fact, characteristic of good teachers. The example reminding strategies provide the foundation for the discussion of explanation questions, and the incidental reminding strategies pepper the discussion with relevant, interesting digressions.

Example remindings in Creanimate. Examples are the bread-and-butter reminding for the Creanimate system, just as they are for any teacher. In the following transcript, the dialogue manager asks the student an explanation question of the category *Why feature?* in response to his request for a bear with a big nose and responds to the student's answer with an example reminding.

> ... What would you make?
> Student: **A bear with a big nose**
> If your bear is going to have a big nose, that should help it to do something.
> Why would you like your bear to have a big nose?
> Student: **So it can smell good**
> That might work. Tapirs have big noses to help them to smell.
> This next video is really hilarious. I like this one because it has strange animals in it. Is that something you would like to see?
> Student: **Yes**
> *[Video: "Tapir Smells"]*

In the previous transcript, the student responds with an answer that the system evaluated as correct. In other words, it is able to determine from its knowledge base that there is at least one animal with the combination of attributes in the student's answer, in this case a big nose to smell effectively. When a student provides a correct answer to an explanation question, the storyteller employs the example reminding strategy in an attempt to identify a case that provides an example to support the student's answer. The ability to perform example remindings requires that an index describe the questions and answers that a case exemplifies.

To account for the three types of answers that students may give to explanation questions, the Creanimate storyteller uses three varieties of example reminding strategy. In addition to the *correct answer* reminding strategy shown in this example, there is an *unverifiable answer* reminding strategy and a *suggest answer* reminding strategy. The unverifiable answer reminding strategy is employed when the student provides an answer that the dialogue manager cannot verify according to its knowledge base. An unverifiable answer reminding shows the student an example of a question for which the student's answer would have been correct. In the following transcript, the student proposes smelling as a reason for his animal to have a beak. The dialogue manager is unable to verify from its knowledge base that beaks are used to smell, so it employs the unverifiable answer reminding strategy to locate a story that shows something that is used to smell.

> OK, Let's find a reason for your bear to have a beak. Is there a reason you want your bear to have a beak?
> Student: **So it can smell better**
> I've never heard of any animals that have beaks to smell, but there are some other ways that animals smell.
> For example, pikes smell. Pikes use their nostrils to help them to smell.
> I have an interesting video about that. I like this one because it has mean fish in it.
> Would you like to see a video about that?
> Student: **Yes**
> *[Video: "Pikes Smell"]*

The third form of example reminding, the suggest answer strategy, is employed when a student chooses not to answer an explanation question and asks the computer to suggest an answer instead. In response, the program suggests an answer in the form of an example, as in the following transcript.

> We need a reason for your dog to fly. Is there a reason you want your dog to fly?
> Student: **Give me some suggestions**
> Canada geese fly. They fly to migrate from the cold. I have a really good video about that. Would you like to see it?
> Student: **Show me a video**
> *[Video: "Geese Migrate"]*

Incidental remindings in Creanimate. The two types of incidental remindings generated by the Creanimate storyteller are *similarity-based remindings* and *expectation-violation remindings.* The similarity-based reminding strategy is designed to retrieve a case that is similar to an immediately preceding case but is also different from it in an interesting way. Similarity-based remindings help students draw appropriate generalizations, and they introduce them to things they might not see in other contexts. The second form of incidental reminding, the expectation-violation reminding, retrieves cases that violate students' likely expectations. This reminding strategy uses the element of surprise to introduce students to uncommon phenomena.

Similarity-based remindings. One of the risks of teaching with examples is that students may miss the opportunity to make generalizations from the specific examples they see. Therefore, a case-based teaching system should assist students to form generalizations at the appropriate levels of abstraction. The strategy that Creanimate uses to help the student form appropriate generalizations is called similarity-based reminding. In similarity-based reminding, the system retrieves a case that illustrates the same basic principle as a previous example, but which is sufficiently different to allow the student to form an appropriate generalization. The following example of a similarity-based reminding was initiated by a student's request for a tortoise that could run fast.

> Cheetahs run fast. Do you know why cheetahs run fast? (I have an impressive video about that).
> Student: **So they can catch other animals**
> That is right. Would you like to see that?
> Student: **Yes**
> *[Video: "Cheetah Pursuing Prey"]*
>
> Similarity-based reminding:
> That reminds me of a cool video. Fishing bats also move fast in order to get food. Only, instead of running fast to pursue their prey, they fly to pounce on their prey. Would you like to see that?
> Student: **Yes**
> *[Video: "Fishing Bat"]*

In order to perform similarity-based reminding, the system must be able to identify cases that are similar, but not identical, to the given example. This is done by searching through its knowledge base for videos that share an abstraction with the preceding example. In this case, the descriptions of the action in both videos share the abstraction *move fast to hunt.*

Expectation-violation remindings. Surprise is a great motivator for learning. When some expectation that you have is violated by an experience or observation, you become surprised and are motivated to understand why your expectation was not met. This is what Schank (1982) called *failure-driven learning*. Expectation failures do not just promote learning, they provoke interest. The reminding strategy called expectation-violation reminding capitalizes on surprise to provoke interest on the part of a student, while also broadening his or her exposure. To present videos that violate a student's expectations, the system must have knowledge of what sorts of things students are likely to believe. Expectations take the form of generalizations, such as "Mammals do not fly." This particular generalization leads to the following expectation-violation reminding.

> If you could create a new animal by taking an existing animal and changing it some way, what would you make?
> Student: **A dog that can fly**
> Actually some mammals do fly. For example, fishing bats fly. I have a cool video about that. Would you like to see that?
> Student: **Yes**
> *[Video: "Fishing Bat"]*

Part of the index for the fishing bat video contains the information that bats violate the expectation that mammals do not fly. When the student asks for an animal that violates the same expectation (in this case, a dog that can fly), this matches the expectation-violation in the video and the program gets reminded of that video.

All three reminding strategies capitalize on a student's current context to present a case that is directly relevant at that moment. The student sees cases that respond to an action or answer that the student has made in the course of his session. The cases retrieved by the reminding strategies provide concrete cases to illustrate the current discussion, promote appropriate generalizations, and widen a student's exposure. In order to facilitate these reminding strategies, however, a storyteller must have an appropriately indexed library of cases.

Indexing

In the course of a dialogue, a case-based teaching storyteller examines indexes in its case library in order to retrieve cases that are relevant to a student's situation. To support effective case retrieval, the indexing vocabulary must be expressive enough to describe the range of storytelling situations that arise in the task environment. An index must capture enough information about a case that the storyteller can identify situations to which the case applies. An index is neither the complete representation of a case, nor is it necessarily the summary of a case. The role of an index is to describe when and how a case should be presented. Therefore, an index may make no reference to the contents of the case but may instead describe the sorts of settings in which the case is appropriate.

An index in a case-based teaching system is typically a structured data representation implemented in the form of slots and fillers. The set of slots that makes up an index for a particular case library provides the structure of the index, and the allowable fillers comprise the indexing vocabulary. Just as with reminding strategies, the structure and contents of indexes are determined by the educational goals of the system, the structure of the task environment, and the nature of the subject matter.

The indexing vocabulary in Creanimate. Although the primary role of the indexing vocabulary in a case-based teaching system is the support of indexing and retrieval, the indexing vocabulary in Creanimate also plays a critical role in supporting the dialogue manager. The representation language that provides the indexing vocabulary also enables the dialogue manager to draw inferences about animal adaptations and to understand and generate natural language in order to manage its open-ended question-and-answer dialogues. The indexing vocabulary in Creanimate supports five major activities of the storyteller and dialogue manager:

- Case retrieval (storyteller).
- Communication (between dialogue manager and storyteller).
- Dialogue initiation (dialogue manager).
- Evaluation of student responses to questions (dialogue manager).
- Natural language understanding and generation (dialogue manager).

To support these activities, the Creanimate indexing vocabulary is capable of expressing both the specific information about cases required by the reminding strategies of the storyteller and the general information about animal adaptation employed by the dialogue manager in conducting dialogues. These capabilities are provided by the Creanimate knowledge representation language, implemented as a semantic network.

In Creanimate, a student's situation for the purposes of reminding, can be characterized by the explanation question under consideration, the most recent output by the system and the student's response, and the cases and explanation questions that the student has already seen. Therefore, the explanation questions that structure the dialogues also determines the structure of indexes. Similarly, the relations expressed by the explanation questions and the conceptual categories to which they refer determines the indexing vocabulary. For instance, the implementation of *How Action?* requires that the knowledge representation include a vocabulary to describe physical features, actions, and the different ways in which actions can depend on physical features. In addition, it has to provide a way of associating a particular feature with a particular action for a particular animal. For the storyteller to retrieve cases that are relevant to a discussion of the *How Action?* explanation question, the indexing vocabulary has to be able to indicate that a particular case illustrates the use of one or a set of physical features to support an action.

The Creanimate indexing vocabulary is composed of two classes of representational structures: objects and relations. Objects are used to represent individual concepts such as animals, features, and actions. Relations link two of these objects together in a structured relation so that they can be associated with an animal. For example, a relation may associate a physical feature and an action to a particular animal that uses the feature to perform the action. The object classes in Creanimate include

- Features: Physical characteristics of animals, that is, beak, claws, fur, and small size.
- Actions: Activities of animals that are performed using features, that is, running, swimming, biting, and scraping.
- Behaviors: High-level, goal-directed activities of animals, for example, hunting, fleeing predators, and attracting a mate.
- Animals: Specific animals or abstract categories of animals, for example, cheetah, raptor, pike, and mammal.
- Phys-obj's: Physical objects other than animals that are found in the world, for example, plant, rock, and seed.

Objects are represented in Creanimate as nodes in hierarchical networks with links to other objects. Objects are connected to other objects of the same type through a variety of abstraction and packaging hierarchies. For example, behaviors may be connected to other behaviors by *part-of, achieves,* and *isa* relations. Objects are connected to objects of other types in ways that indicate how they relate to each other in the natural world. For example, actions are connected to the behaviors that they support, the features that are required to perform them, and the animals that perform them.

The Creanimate indexing vocabulary contains three primary relations. They connect objects together in the following ways.

- Feafuns: A feafun connects a physical feature to an action that it is used to perform, for example, *long legs in order to run fast.* The name *feafun* is an abbreviation of *feature for a function.*
- Plans: A plan connects an action to a behavior that it supports, for example, *run fast in order to pursue prey.*
- Bplans: A bplan connects a lower level behavior to the higher level behavior that it supports, for example, *pursue prey in order to hunt.*

Relations are used to tie features, actions, and behaviors together and associate them with a particular animal. For example, the animal *giraffe* possesses the feafun *long neck in order to reach.* These relations correspond directly to the explanation questions that the system poses. For instance, the questions "What feature could an animal use to reach for food?" and "Why might an animal have a long neck?" are both answered by the feafun *long neck in order to reach.*

In addition to the primary relations previously listed, Creanimate contains a special kind of relation called a *rule.* Rules are used to support the expectation-violation reminding strategy.

- Rules: A rule encodes a common generalization, for example, "No fish have wings," that is known to have exceptions.

Rules differ from the other relations in Creanimate in that the information they encode is not true under all conditions. Rules are used to capture generalizations that students are likely to believe for the purposes of expectation-violation remindings. However, every rule in Creanimate has at least one significant exception that is illustrated by a case in the case library. The role of a rule in the storyteller is to trigger expectation-violation remindings about an exception to that rule.

The structure of indexes. An index in Creanimate is implemented as a frame with slots and values. The values are drawn from the indexing vocabulary. Each index describes the attributes and activities of one animal that appears in the video clip. Any particular video clip may have several indexes. Thus, a clip about a robin chasing a cat away from a nest full of chicks might have three indexes, one for the robin, one for the cat, and one for the chicks. Figure 2 shows an index for a case about a woodpecker using its beak to probe for insects. This index is relatively simple; more elaborate indexes can be found in Edelson (1993).

The slots in indexes are divided between primary slots and secondary slots. The primary slots describe an animal and the attributes that it displays in the video clip. The secondary slots serve pragmatic requirements of the storyteller. They assist the storyteller in locating clips on a laser disc, deciding which clips to show first, and preparing textual descriptions of clips to introduce them.

Use of indexes by reminding strategies. The reminding strategies in Creanimate are implemented as searches through abstraction hierarchies. Because indexes are linked to animals, features, actions, and behaviors by bidirectional links, the reminding strategies are able to locate relevant cases by searching through hierarchies of these objects and then retrieving indexes that are linked to them. This is made possible by the hierarchical knowledge representation that makes up the indexing vocabulary. As an example, consider the following transcript showing an example reminding.

```
(DEFFRAME index woodpecker-pecks
     :animals ([animal red-cockadoo-woodpecker])
     :features ([feature small-beak])
     :behaviors ([behavior search-for-insects])
     :actions ([action peck])
     :feafuns ([feafun :feature [feature small-beak  ]
                       :action [action peck  ]
                       ...])
     :plans ([plan :action [action peck  ]
                   :behavior [behavior search-for-insects  ]
                   ...])
     :story [story woodpecker-eats]
     :impact ([impact cool][impact slow-motion])
     :rating 8
     :video ((101001 102500 (disk 'nat-and-brit) "A")))
```

FIGURE 2 An index for a video that shows a woodpecker using its beak at a tree in search of insects.

> OK, let's find a reason for your frog to have wings. Is there a reason you want your frog to have wings?
> Student: **So it can fly**
> That's a good idea. White breasted sea eagles use their feathered wings to help them to fly.
> Would you like to see that?
> Student: **Yes**
> *[Video: "Sea Eagle Catches Fish"]*

In this transcript, the explanation question is "Why have wings?" and the student's answer is "So it can fly." Once the dialogue manager verifies that flying is a reason that animals have wings, the correct answer reminding algorithm initiates a search for an example case. To be an appropriate correct answer reminding, an index must contain the feafun *wings in order to fly,* which is called the *target concept.* To find a case that contains the target concept, the Creanimate storyteller searches down abstraction hierarchies starting from one of the values in the target concept. The abstraction hierarchy beneath the feature *wings* includes *feathered-wings, insect-wings, leathery-wings,* and *manta-wings.* The abstraction hierarchy beneath the action *fly* includes *soar, climb,* and *dive,* among others.

The correct answer reminding strategy is the simplest of the reminding strategies. Other reminding strategies use different information and pursue different search algorithms through the system's knowledge base. For example, the similarity-based reminding strategy employs a search algorithm resembling spreading activation that starts from a description of the most recent case and examines similar cases to see whether they meet the criteria for a similarity-based reminding. The expectation-violation reminding strategies employ search algorithms that are similar to those used by the example reminding strategy, however, they focus on the information contained in the *expect-viols* slot in indexes. The algorithms used to implement these reminding strategies were described in detail by Edelson (1993).

Coverage of indexes. The most recent Creanimate knowledge base includes 206 indexes. However, the prototype version tested in the study described had access to only 60 min of video, which limited it to 135 indexes. The indexes in Creanimate cover a wide range of birds, mammals, fish, reptiles, and amphibians. Creanimate is able to get a great deal of coverage out of the clips that it possesses because each clip can be used in several different contexts. To maximize the value of clips, versatility is emphasized in the case selection process. The versatility of a clip is determined by the number of animal attributes that can be recorded in an index for that clip. For example, an index with five different features can be retrieved to illustrate five different *Why feature?* dialogues by the example reminding algorithm. The average index in Creanimate includes at least two different values for each type of animal attribute, that is, more than two features, two actions, and two behaviors. Because actions and behaviors can be used in both *how* and *why* dialogues, this means that the average index can be used to illustrate discussions of at least eight different, specific explanation questions.

THE CASE-BASED TEACHING ARCHITECTURE IN CONTEXT

The development of the case-based teaching architecture is taking place in an environment of increasing attention on the learning sciences. This research on the case-based teaching architecture has important ties to other methods for teaching with cases, to other research on artificial intelligence in education, and to other developments in technology in education. In this section, I briefly describe the relations between this work and some closely related work in these three areas.

The Case-Based Teaching Architecture and the Case Method

The case-based teaching architecture has much in common with the case method of instruction that is commonly employed in business schools, law schools, and other professional education (McNair, 1954; Williams, 1992). Both the case-based teaching architecture and the case method share an emphasis on learning from cases and a recognition of the importance of a learner's development of a personal case library. However, the case-based teaching architecture focuses on a specific element of teaching with cases: the presentation of cases in response to opportunities that arise in the course of learning interactions. As it is commonly practiced, the case method employs cases to establish the context for a discussion or problem-solving episode. Within the context of a case, students might discuss interpretations of the situation or present solutions to posed problems. Instead of focusing on the use of cases to establish such a context, the case-based teaching architecture focuses on the presentation of cases that respond to a student's context within a task environment. This recall and presentation of cases in response to learning interactions is by no means unique to the case-based teaching architecture; it is an important element of the case method as it is practiced in many settings.

The Case-Based Teaching Architecture and Intelligent Tutoring Systems

With few exceptions, research on ITSs has not considered the use of cases in instruction. Two notable exceptions are Clancy's (1987) GUIDON system that used a case method approach to teaching medical diagnosis and Ashley and Aleven's (1992) approach to teaching the appropriate use of precedents in legal reasoning. Instead, the primary focus of most ITSs has been on student modeling and diagnosis. The goal of the ITS approach is to respond intelligently and dynamically to the needs of a student by representing expert and novice understanding for the particular domain and inferring the student's understanding from limited information. The use of rule-based representations generally restricts the applicability of the ITS approach to domains that lend themselves to such representations, primarily formal domains, such as algebra, physics, and computer programming.

One of the goals of the case-based teaching architecture is to extend beyond well-structured domains by adopting a case-based approach to the challenge of responding to students' needs. The case-based teaching architecture aims to reduce the amount of representation and inference necessary to track the student while increasing the richness of the communication with students through the presentation of cases. Typically, the criterion for intervention in an ITS is "Can the system infer that the student is lacking a particular rule, fact, or skill?" The case-based teaching architecture reduces this criterion for intervention to "Is a case relevant to the student's situation?" by using cases with content rich enough that a student will profit from the case in that situation regardless of the exact state of the student's knowledge. By reducing the reasoning requirements of the system to reasoning about the student's situation in the task environment rather than the student's knowledge, the case-based teaching architecture is designed to widen the range of subjects that can be taught by computer-based systems.

In addition to attempting to reduce the burden of inference, the case-based teaching architecture also addresses the issue that Wenger (1987) called *knowledge communication.* Because most ITSs are rule-based systems and have at their hearts a model of a student as a rule-based reasoner, the form that their communication with students usually takes is rules. In contrast, the case-based teaching architecture is designed to support case-based reasoning through the presentation of cases in context.

The Case-Based Teaching Architecture and Goal-Based Scenarios

A recent strand of research in educational technology that is closely related to the case-based teaching architecture is directed at implementing a framework called the Goal-Based Scenario (GBS) framework (Schank, Fano, Bell, & Jona, 1993–1994). A GBS provides a context in which a learner pursues a set of authentic goals and increases his or her understanding of a particular subject matter through activities that lead to the achievement of those goals. The implementation of the GBS framework on a computer takes the form of a learn-by-doing environment in which a user acquires a particular set of skills and knowledge by adopting a role in a simulated scenario. Implemented GBSs on the computer include Broadcast News (Kass, Dooley, Luksa, & Conroy, 1994), a system that teaches high school students about political science and current events by enabling them to assemble a television news show; Sickle Cell Counselor (Bell, Bareiss, & Beckwith, 1993–1994), a museum installation that teaches about sickle cell disease by allowing the user to play the role of a genetic counselor for couples concerned about passing on sickle cell disease to their children; and Human Resources Management (Acovelli & Nowakowski, 1994), a module of a corporate training system on business practices that teaches about human resources management by allowing the learner to play the role of manager.

The core of the GBS framework is a carefully selected scenario that provides students with the motivation and opportunity to learn. In this respect, the GBS framework represents an important approach to the construction of effective, engaging task environments. The framework is broad enough to encompass a wide range of strategies for responding to the opportunities for learning that arise as a student participates in the scenario. Presenting cases as a response to student actions fits within the GBS framework, as do coaching, outcome-driven simulation, and direct tutorial intervention. Therefore, the case-based teaching architecture and the GBS framework are compatible approaches. The GBS framework provides recommendations for effective task environments that could be used within the case-based teaching architecture, and the case-based teaching architecture provides a mechanism for capitalizing on opportunities for learning through case presentation that can be used within the GBS framework.

EXPERIENCE WITH CREANIMATE

In addition to a significant amount of informal testing, formal evaluations of Creanimate were conducted in two suburban Chicago public schools. The results reported here are from a study involving 23 fourth graders. The studies of Creanimate were designed to examine three central questions: (a) Did the software succeed in engaging students? (b) What patterns of use emerged among students? and (c) Did the use of Creanimate have any impact on questioning and reasoning with cases by students within the animal adaptation domain? The examination of the third question, on learning outcome, yielded no results of statistical significance and are not discussed in this section. They are, however, discussed in the following section as part of the discussion of future evaluation. Because the interactions described here represent the first time the software was used extensively by children, they were also treated as an opportunity to identify and repair flaws in the software and to enhance the natural language parser. As a result, the software was modified significantly both in the course of and following these evaluations.

Setting

The primary study site was a socioeconomically and racially mixed elementary school in suburban Chicago. Detailed demographic information about the particular students was not available, however the 12 girls and 11 boys were primarily from lower socioeconomic, working-class backgrounds. Nineteen of the 23 students were African American. All students reported previous experience with computers, but half had never used a mouse before. Students were called out of regular class activities singly or in pairs to use Creanimate for 45 min, approximately once a week for three sessions. The students' sessions were clearly affected by the unfinished nature of the software. A careful reading of the students' transcripts reveals parsing problems and "buggy" output in virtually every transcript. Most errors, such as grammatical errors in the system's output, did not appear to disrupt students. However, students were occasionally frustrated by the parser, confused by its incomplete or inaccurate understanding of their input, or puzzled by incoherent or incorrect output.

Student Engagement

With few exceptions, the students used the system for as long as we would allow them, 40 to 45 min, at all three sessions. Typically when the researcher announced that a session was over, students would ask to continue beyond the allotted time. At the end of each session, students were asked if they wanted to come back and use the program again the following week. All students responded positively.

All students' sessions were observed by researchers who kept notes about both student and program behavior. It was evident from these observations that students found the software engaging. Research notes recorded that many students laughed and exclaimed at the videos, carefully composed answers to questions, and begged to use the program for "just 2 more minutes." Subjects often tried to draw a researcher or a student at a neighboring computer into their session. The observations also recorded that, although students were engaged, they were not necessarily engaged in ways that completely met with the designers' expectations. Some students did not appear to be reading the questions the program posed. Others appeared to read them, but not to be interested in answering them. Some students seemed to find the questions compelling and devoted a great deal of attention to answering them. However, some of these same students then showed little or no interest in viewing the videos that were offered in response to their answers. Some students found ways to view videos without ever answering questions, and several chose to view the same video over and over again. One girl, a dancer, asked for a butterfly that could dance at the beginning of all three of her sessions and consequently watched the same videos showing animals dancing in all three sessions.

After the students' first session with Creanimate, they were interviewed about how they liked different aspects of the program. The average rating for all aspects of the program was positive (Table 2).

In a pairwise comparison, ratings for answering questions and for the user control buttons were significantly lower than the ratings for the other features. The lower rating for the user control buttons may reflect the limited amount of use these buttons received. The lower rating for answering questions (between *neutral* and *like*), although positive, is lower than the mean rating for watching videos by 1.12, $t(19) = 6.319, p < .0001$, and the mean rating for creating an animal by 1.22, $t(19) = 3.340, p < .01$. Students' relatively low rating for how they liked answering the program's questions is also reflected in the use patterns described in the next section, where some students' transcripts revealed a clear tendency to avoid answering questions. This low rating relative to other aspects of the software indicates that a closer examination of the motivational aspects of these questions may be necessary.

In the students' interview responses and researchers' observations, there is significant evidence that the students found Creanimate engaging. However, the observations raised questions about the nature of their engagement, which were further examined through an analysis of the use patterns that appeared in their transcripts.

TABLE 2
Student Opinions on Various Aspects of Creanimate

	M
Creating an animal	4.22**
Answering questions	3.10*
Watching videos	4.12**
How the program looks	4.25**
User control buttons	3.58*
Using the mouse	4.14**

Note. $N = 20$. Students were asked how they felt about different elements of the program and were asked to choose from 1 *(dislike a lot)*, 2 *(dislike)*, 3 *(neutral)*, 4 *(like)*, or 5 *(like a lot)*. Means followed by two astericks are signifigantly higher then those with a one asterick in paired *t* tests ($p < .05$).

Patterns of Use

Following the observation that students were responding to different aspects of the program in different ways, an analysis of their interactions was performed in an effort to identify specific patterns of use and to quantify the prevalence of the various use patterns (Cleave, Edelson, & Beckwith, 1993). All the students' interactions with Creanimate were recorded in transcripts that captured every action of both the software and the student. For this study, more than 60 transcripts were analyzed, excluding only the 5 transcripts from sessions that were significantly abbreviated because of technical problems. To facilitate analysis, student–computer interactions were coded into categories, such as those shown in Table 3. Because transcripts varied significantly in the number of events ($n = 58–247$, $M = 120$) per session, the percentage of total events, rather than the number of events, for each category was used in the analysis.

An analysis of the correlations of the frequency of particular events in student transcripts revealed an interesting result about the relation between answering open-ended questions and watching videos. There was a negative correlation between typing text in response to questions (as opposed to selecting an answer from a menu) and watching videos when they were offered, $r = -.60$, $p < .01$. This indicates that students who were motivated to respond to questions were less likely to watch videos. Conversely, students who asked the program for suggestions of answers to questions instead of entering their own answers were more likely to watch videos, $r = .53$, $p < .01$. If, as this limited sample suggests, interest in question-and-answer dialogues does not correlate with interest in video cases, there could be important implications for the success of Socratic case-based teaching

TABLE 3
A Sample of the Categories of Student Actions Used in Coding *Creanimate* Transcripts

EQ-PLAN	Student began a dialogue about an explaination question for a plan.
COM-PLAN	Student completed a dialogue about an explaination question for a plan.
IN-EQ <text>	Student inputted an answer for an explaination question posed by the program.
SUGGEST	Student selected "Give me some suggestions" in response to an explaination question.
NO-VIDEO	Student responded "no" to an offer to see a video.
VIDEO	Student selected "yes" in response to an offer to see a video.
WHAT	Student selected "What other videos can I see?" in response to an offer to see a video.

Note. The coding system included a total of 59 event categories.

systems. However, this result may only be a limited reflection of individual preference because the analysis considered each event category as a percentage of all events. Therefore, an increase in one event category, such as answering questions, must be accompanied by a decrease in some other event category. Further study will be necessary to clarify this result.

To better understand the overall patterns of use, a cluster analysis was performed on the event counts, using Euclidean distance and a furthest neighbor technique. *Z* scores of the variables representing the count of each event as a percentage of all events were used in the cluster analysis to balance events that occurred more frequently (e.g., watching a video) with events that occurred more rarely (e.g., completing a dialogue). The cluster analysis produced five clusters of users, which we named *balanced users, video avoiders, compliant users, video hoppers,* and *resisters.* Representative transcripts from three of these groups are included in the Appendix.

The characteristics that distinguished clusters were students' preference for typing responses or asking for suggestions, the amount of video that they watched, and their tendency to exert control over the flow of the dialogue or follow the program's lead. Figure 3 shows the frequency of certain events, expressed as an average of the total events, both by cluster and for all students.

Analyzing the clusters based on field notes and transcript events, we developed the following characterizations of the members of each cluster.

- Balanced users ($n = 4$): The balanced users took advantage of the greatest range of the features offered by the software. Balanced users stood out as the group that typed in answers to questions more than any other group, but with the exception of video hoppers, they also watched the most video clips. When they typed responses, their answers were significantly longer than were those of the other groups. Finally, unlike compliant users, who also answered questions and viewed videos in large numbers, balanced users took advantage of opportunities to exert control and were persistent in their efforts to express themselves in free text. They commonly typed in second and third answers after the parser failed to understand an initial input.

- Video avoiders ($n = 5$): The cluster analysis showed video avoiders and balanced users to be the two most similar groups. The primary difference was that video avoiders watched fewer videos, the fewest of any group, and declined offers to view video more than did any other group. Like balanced users, video avoiders showed a preference for typing in responses, rather than asking for suggestions. After balanced users, video avoiders typed input and entered a second input when the first was not understood more often than did any other group. However, they were the group most likely to type in answers that Creanimate could not understand and to give unverifiable answers leading to tangent dialogues about their answers. Video avoiders also exerted more control than did balanced users, using the "Skip This" button more than did any other group.
- Compliant users ($n = 5$): Like balanced users, compliant users were characterized by balance in their use of the program. However, as a group they were

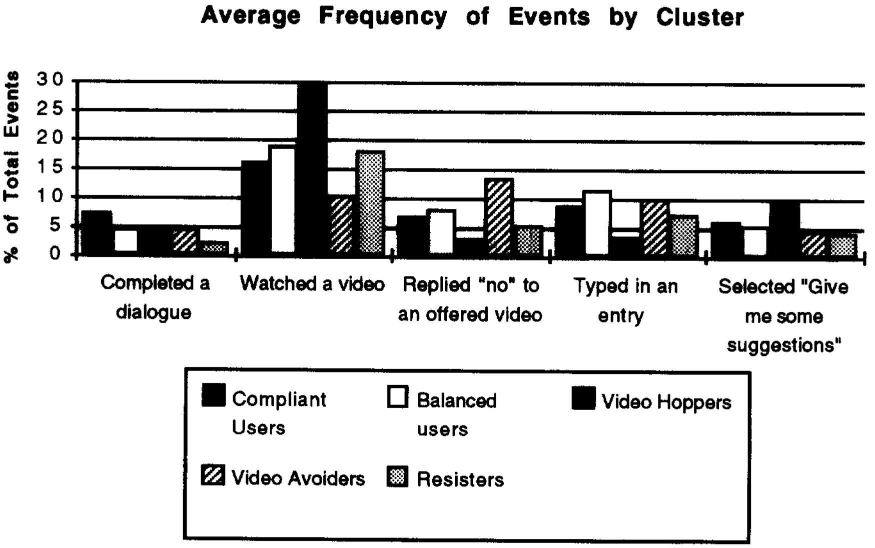

FIGURE 3 A graph of the average frequency of selected events by cluster.

distinguished by minimal efforts to control the dialogue and lack of persistence in response to failures by the parser to understand their answers. Their transcripts often read as if the compliant users were trying to cause the least trouble for the program. They answered questions an average amount of times, but they asked for suggestions instead of typing answers slightly more often than average. Unlike balanced users or video avoiders, they rarely tried a second time when the parser could not understand their input. In contrast to the other groups, compliant users appeared to have the completion of dialogues as a goal. They generally declined opportunities to take tangents from the central dialogue, and they completed dialogues almost twice as often as did any other group. Compliant users saw an average number of videos.

- Video hoppers ($n = 2$): Video hoppers seemed to be motivated to watch videos at the expense of all the other aspects of the program. The two video hoppers saw 48 and 52 videos each across their three sessions, more than twice the average for all students, which was 23.5. Their transcripts reveal strategies that allowed them to navigate between one video and the next in the shortest possible time. As a result, they input text the least of any students, averaging 6 text inputs over all three sessions, compared to an average of 12 for all students.
- Resisters ($n = 7$): Members of the resister cluster can only be characterized as not accepting or not understanding the basic structure of the Creanimate dialogues. Their failure to comply to the program's normal flow was illustrated by the fact that they rarely completed dialogues (half the class average), and used the "Start Over, Skip This," and "What's the point?" buttons more than did any other students. Although resisters had the lowest number of successful parses, they were the most likely to choose their own animal (e.g., shark, lion, panther, boar, horse, koala) instead of the suggested animals, and they had the highest number of responses that were anthropomorphic, for example, "boxing gloves," "bonnet," "do the mc hammer," and "kick someone's butt." Field notes record that resisters were motivated by the act of creating an animal more than were the other students and were frustrated by the inability to see their animal right away or draw it themselves. Resisters watched an average number of videos, but, accounting for the disjointed nature of their dialogues, their attitude toward video probably most resembled that of the video hoppers. As a group, the resisters appeared to respond to the idea of creating an animal and even to seeing the video, but not to participating in Creanimate's dialogues.

The cluster analysis provides reason to be both encouraged and discouraged about Creanimate and teaching with cases. The balanced users, video avoiders, and compliant users (14 of 23 students) appeared to respond positively to the question-based dialogue structure. Similarly, all but the video avoiders (18 of 23) appeared to respond positively to the video cases. On the other hand the video avoiders (5 of

23), the video hoppers (2 of 23), and the resisters (7 of 23) appeared to respond negatively to one or more central elements of the program. The existence of the balanced user and compliant user clusters shows that a significant number of students responded to Creanimate's combination of the dialogue structure with video cases. However, the video avoiders, the video hoppers, and the resisters raise questions about the relative value of a Socratic case-based teaching system like Creanimate to a user who does not fully engage with either the dialogue or the video cases.

The evidence from observations, interviews, and transcript analysis—that all of the students found Creanimate engaging but that different individuals responded to different elements—raises the important question of how differences in engagement impact learning outcomes. The theory that argues for the importance of context in learning from cases that underlies this research predicts that neither engagement with cases nor with the dialogue alone would lead to an understanding that is as well developed as that which would result from engagement with both. Students who only engage with the video would be expected to develop poorly indexed case libraries, and those who only engage with the dialogue would be expected to develop a framework for understanding that is not well supported by cases. Further examination of these issues will require additional research.

STRENGTHS, LIMITATIONS, AND FUTURE DIRECTIONS

The Creanimate system is an attempt to put the case-based teaching architecture into practice. Its primary research goal was to develop a framework for the presentation of educational cases in the context of an engaging task environment. The Socratic case-based teaching architecture embodied by Creanimate is such a framework. This architecture provides for a question-and-answer dialogue between the computer and a student that establishes a context for learning from video cases. The indexing scheme and reminding strategies enable the storyteller to capitalize on opportunities for learning that arise in the course of a dialogue with relevant cases from its video library. The construction of Creanimate, a system encompassing over 200 video clips that is capable of engaging fourth-grade students for over 2 hr, represents an important level of success for this design and engineering effort.

However, the initial success of this particular implementation effort only emphasizes the need for further research on the case-based teaching architecture. Three areas in particular stand out. First, additional work is necessary to develop architectures for more sophisticated and engaging task environments. Second, the architecture must be generalized to allow for the application of the case-based teaching architecture to a broad range of domains. Finally, additional experience and evaluation are necessary to investigate the impact of case-based teaching systems on students' understanding and abilities. Although additional experience and evaluation are possibly the most important of the three areas, the technology is still quite immature, and it would be a mistake to be overly influenced by the lack of demonstrable, positive learning outcomes at the early stages of the development of this technology. For this reason, it may be necessary to continue development and testing of the technology for quite some time on the strength of the pedagogical theory, before robust empirical results to support the theory become available.

Engagement

Although the development of Creanimate was a significant effort and its initial use provided reason for encouragement, clear limitations are evident in its ability to engage students in active learning. The most significant limitations in the Creanimate task environment's engagement of the student are (a) the reactive role of the learner in the dialogues, (b) the limited flexibility of the dialogues, (c) the lack of a clearly articulated goal for the student, and (d) the passive role of the student with respect to the video cases.

First, although the question-and-answer dialogues that the Creanimate dialogue manager is capable of conducting provide for rich learning opportunities and a great deal of student control, they fall short of the truly engaging active learning that is the goal of the case-based teaching architecture. Although they give a learner room to propose hypotheses and give direction to an interaction, they place the student in a reactive, overly passive position. Rather than having the option to ask questions him- or herself, the student is placed in the position of responding to questions posed by the computer. This reactive role falls short of the truly active learning that is the goal of the case-based teaching architecture.

Second, compared to both human conversation and other styles of human–computer interaction, the dialogues are limited in their flexibility. At any point in a Creanimate dialogue, a student has a fixed number of options for responding, which will result in predetermined and predictable responses by the computer. Question-and-answer discussion can be an extremely efficient mechanism for establishing and capitalizing on opportunities for learning, but other mechanisms that allow students to interact more directly with subject matter are often more engaging. For example, design, modeling, and simulation environments can provide even richer, more motivating, active learning opportunities. Without needing to discard Socratic-style dialogue as one aspect of the interaction, Creanimate, as a case-based teaching system, could be made significantly more engaging if students had the ability to construct their animals through a direct-manipulation interface and to observe their creations interacting with a simulated environment. In fact, other efforts to teach biology through design focused on this construction and simulation approach (Coderre, 1988; Resnick, 1988, 1991; Travers, 1988).

Such an approach would help resolve the third major limitation, the lack of a clearly articulated goal for the student. In the current implementation of Creanimate, students are invited to create a new animal without any indication of what objectives are being served by creating the animal. Only through the questions asked by the dialogue manager do students get the idea that they are creating an animal in order to consider how it might survive. Some students did not expect to be asked the sorts of questions that the dialogue manager posed, and they were clearly put off by them. A task environment that made this goal clearer early in the interaction and that gave the student the opportunity to manipulate and observe that animal in a simulated habitat would likely prove significantly more engaging than the current Creanimate.

Within such an enhanced task environment, the Socratic dialogue manager could take its place as one of the mechanisms by which a student receives feedback and experiences opportunities for learning. The dialogue manager might respond to certain student actions as one element of this richer environment. In such an integrated system, the storyteller architecture would also be enhanced to respond to opportunities for learning that arise through students' interactions with other elements of the task environment.

The fourth limitation of Creanimate's Socratic case-based teaching architecture to engage the student in active learning lies at the borderline between the task environment and the storyteller. The problem is that in the current implementation, the student's exposure to cases is essentially passive. Students watch a video and, at most, are asked to answer a simple question about it, for example, "Now do you know why sage grouses dance? [*So they can ... , Show me the video again, What other videos can I see?*]" Creanimate never asks a student to reason in any meaningful way with the cases he or she sees. Additional reasoning could both expand and reinforce the indexing in a student's memory that resulted from the initial viewing of the case and the context in which it was viewed. In addition, observations revealed that video clips very often raised questions in students' minds. In the current architecture, these questions are lost opportunities for learning. Creating a system that could answer arbitrary students' questions is currently unreasonable, however, simply allowing them to record their questions for future discussion would be an improvement.

Currently, Creanimate's ability to engage students comes from the questions it poses and the videos it presents. Adding a more active, less reactive task environment, and encouraging students to reason more deeply with the cases, could greatly enhance the program's ability to engage students in active learning.

Sophistication of Subject Matter

Beyond its ability to engage, the Socratic case-based teaching architecture as it is instantiated in the Creanimate system is also limited by its inability to deal with complexity in the subject matter. The limitation is its relatively simple knowledge representation. The simplicity of Creanimate's knowledge representation was motivated by the desire to produce a functional system that could be used for formative evaluation. However, with additional effort, the knowledge representation and indexing scheme could be extended to account for many issues of animal adaptation that it is currently unable to deal with effectively. These issues include conflicts among animal attributes (e.g., weight interfering with flying and fur interfering with keeping cool), effects of habitat, developmental stages of animals, and activities involving multiple animals (e.g., communication, conflict, and nurturing).

An example of the knowledge representation's current limitations is revealed in the transcript in the first section of this article. In that transcript, the student created a monkey with wings. After she gave her monkey wings, the program asked her what other features the animal would require to be able to stay aloft. The dialogue manager's goal in this dialogue was to ensure that the student's animal had one of the combinations of body size and wing size that it knew worked for flying. The student could achieve an appropriate balance by adjusting either the wings' size or the animal's size so that they were in balance (e.g., large wings for medium-size animals, medium-size or small wings for small animals). A good representation to handle the relation between wing size and body size would be able to represent the fact that larger wings are necessary for larger animals because the wings compensate for the fact that large size interferes with flying. This would enable the dialogue manager to say, "Large wings can enable an animal with medium size to keep itself up in the air." However, the current representation can only account for the fact that particular combinations of features allow an animal to perform a particular action, not the possible negative or positive interactions between the individual features and the action. Thus, it simply knows that medium body size and large wings or small body size and small wings are two combinations of features that enable an animal to stay aloft. For that reason, the dialogue manager makes statements such as, "Medium size and large wings can work together to help an animal to keep itself up in the air." The Creanimate knowledge representation would need to be extended to support this additional level of sophistication. Such an enhancement would by no means push the state of the art of knowledge representation, but it raises the question that haunts any knowledge representation effort: How good does the representation need to be to be good enough?

With an extension of its knowledge representation and inferential capabilities, Creanimate could conduct dialogues about a wider range of explanation questions, as well as both handle student's answers and generate explanations with a greater degree of sophistication. The quality of the interaction and the opportunities for learning that would result would clearly be improved as a result of such an extension. Although the power of Creanimate is clearly limited as a result of the simplicity of its knowledge representation, extending it would be relatively straightforward. Extensions to the knowledge representation, indexing scheme, and inferential capabilities would be possible without taxing the techniques of conventional, symbolic artificial intelligence employed by Creanimate.

Generalization of the Architecture

At this early stage, it is difficult to say with any certainty how effectively the Socratic case-based teaching architecture will generalize beyond Creanimate in terms of educational effectiveness. However, from the technical point of view, it is an important strength of Creanimate's architecture that the underlying structure is not tied in any way to the specific domain of animal adaptation. Explanation questions for a different domain, along with the knowledge representation and inferences they would require, could be accommodated with simple modifications to Creanimate's current infrastructure. Likewise, the indexing scheme and reminding strategies are not domain independent and could be applied with minimal adaptation to new knowledge bases.

The key to resolving the technical issues associated with applying the Socratic case-based teaching to new subjects is the development of software engineering tools to support the process. Developing a knowledge representation; defining inference rules to pose questions and evaluate students' answers; collecting and indexing a case library; and, in the case of the expanded architecture previously described, creating a direct-manipulation interface and simulation environment are time- and labor-intensive processes. The development of Creanimate began to confront this challenge through the creation of a knowledge representation and indexing tool that allowed a nonprogrammer to create the entire knowledge base and case library (Edelson, 1993). This tool was used subsequently to develop other knowledge representations for other systems.

The question of which domains are appropriate for the architecture remains open. It is easy to think narrowly about how to extend Creanimate to other domains in which students can learn by considering new inventions in the way they do in Creanimate. These domains include design and engineering, in which the creation of objects is a defining characteristic of the domain, but also other domains, like animal adaptation, in which design is not the ordinary way of approaching the domain, but which can provide valuable opportunities for learning. These domains are not limited to the sciences. For example, designing a government or a religion as a way of studying political science or comparative religions are applications of this approach that have been discussed. However, beyond these relatively *near transfer* applications of the Socratic case-based teaching architecture to learning through design are tasks such as planning, problem solving, and diagnosis, all of which lend themselves to the use of explanation questions and cases as a means for exploring the implications of decisions.

Evaluating the Architecture

The research presented here focused primarily on the technological issues of instantiating the case-based teaching architecture. Obviously, resolving these issues is on the critical path of developing effective case-based teaching systems, but the ultimate goal of this research is not to develop new technologies but to provide effective educational experiences for students. Therefore, the evaluation of the impact of case-based teaching systems must be an element of this research. However, the inability to demonstrate positive learning outcomes while the technology is still immature should not be taken overly seriously. The risk of throwing the baby out with the bath water is great in weighing early assessments too heavily, especially when the theories on which the architecture is based are as compelling as the theories of case-based reasoning. These cautions aside, it will be important to address the evaluation of the case-based teaching architecture seriously and to expand on the results of the studies reported here.

The success of the case-based teaching architecture, or any educational interaction, depends critically on motivating the learner to expend effort and attention on the interaction. The goal of the case-based teaching architecture is to provide the student with an intrinsic motivation that will engage him or her in active learning within the task environment. For that reason, evaluating the motivational aspects of the architecture was the first priority. The studies described in this article provided valuable information about the different ways in which students respond to Creanimate. These studies need to be extended to gather information about differences by age, sex, computer experience, and other individual characteristics that often prove to be significant in educational settings.

After motivation and engagement, the next issues for evaluation are learning outcomes. These evaluations must be driven by the objectives of the system being evaluated. The objective of Creanimate is to provide students with a framework for understanding animal adaptation, as expressed by the explanation questions employed by the system, and a case library of animals, their attributes, and activities that will help students reason about animal adaptation in novel situations. As part of this first phase of research, we developed a set of evaluations designed to explore Creanimate's impact on the questions student ask and their use of cases in constructing explanations when presented with unfamiliar animals. Specifically, students were given pre- and posttests that consisted of line drawings of eight exotic animals. For four of these drawings, students were asked to generate questions about the animals. For the other four, they were asked to answer specific explanation questions and provide justifications for their answers. The hypotheses being explored were that students would ask more explanation questions about the relations between physical features, actions, and behaviors, and would refer more often to cases of other animals in their explanations as a result of using Creanimate. Our analysis of their answers yielded some encouraging trends but no results with statistical significance. There are many possible explanations for this lack of results, including a ceiling effect on the pretest, insufficient exposure to the program, and limitations of the experimental software. Further evaluation in different settings will be necessary to gain more informative results. Although it may be possible to obtain immediate results that do show an impact of interacting with Creanimate, such as direct recall of cases, we refrained from those forms of evaluation in favor of an approach that directly pursued the objectives of the design.

A previously unanticipated topic for evaluation that emerged as a direct result of the studies described in the previous section is the impact of different patterns on students' learning. Students' patterns of use appear to be a reflection of persistent individual differences. The implications of these differences could range from (a) the differences correspond to styles of use but have no influence on learning and attitudinal outcomes to (b) Creanimate or the Socratic case-based teaching architecture is not appropriate for certain students because they derive no significant positive outcomes from using it to (c) Creanimate and Socratic case-based teaching are appropriate for all students, but different students derive different sorts of benefits from its use. The examination of these questions will require further investigation of students' patterns of use to see whether these initial findings are robust and to identify their ramifications for learning.

CONCLUSION

From the foregoing discussion of the strengths and limitations of Creanimate, it is clear that as an educational program Creanimate has its share of flaws. However, it is important to remember that the primary goal of this research was to explore the computational challenges of implementing the case-based teaching architecture. Even if the Creanimate program itself never reaches its potential as an educational system, future work that addresses the limitations in technology discussed here should enable the case-based teaching architecture to achieve that potential in other domains. Future research should also continue to attempt to address the motivational, attitudinal, and learning impacts of the approach.

According to theories of case-based understanding and learning, the case-based teaching architecture offers promise as a means for combining two effective instructional techniques: active learning and learning from cases. The development of such an architecture was made possible by important advances in artificial intelligence, cognitive science, and multimedia technology. The Socratic case-based teaching architecture is a specific form of case-based teaching that takes advantage of the power of questions to establish opportunities to learn from cases. As a prototype of a Socratic case-based teaching system, Creanimate is a first step in the exploration of the case-based teaching architecture. The development of Creanimate required technological solutions to the problems of Socratic-style dialogue management, indexing schemes for large case libraries, and the development of reminding strategies. In the development of a working prototype that incorporates more than 200 cases and a knowledge base that includes more than 1,000 animals, physical features, actions, and behaviors, this design and engineering research was successful. On the pedagogical front, the initial experiences with the Creanimate system in public school environments demonstrated the power of such systems to engage students at the same time these experiences raised important questions about the nature of that engagement and the learning that results. It is hoped that further technological developments coupled with additional experience will pave the way for systems that expand the case-based teaching architecture in the future.

ACKNOWLEDGMENTS

This research was supported by a grant from IBM's Program for Innovative Uses of Information Technology in K–12 Education and by the Defense Advanced Research Projects Agency monitored by the Office of Naval Research under Contracts N00014–91–J4092 and N00014–90–J–4117. The National Geographic Society and the Encyclopedia Britannica Educational Corporation provided video footage for experimental use in Creanimate. The Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting. The Institute received additional support from Institute Partners Ameritech and North West Water.

I thank Mitchel Resnick, Alex Kass, Andrew Fano, and two anonymous reviewers for helpful comments on earlier drafts of this article. The idea for Creanimate originated with Roger Schank, and as my doctoral advisor, he made significant contributions to every aspect of this research. The data analysis for the evaluation studies was performed by John Cleave with Richard Beckwith and Bryan Lester. In addition to myself, the Creanimate development team included Bob Kaeding, Riad Mohammed, John Cleave, Will Fitzgerald, Diane Schwartz, Ken Greenlee, and Ian Underwood. William Purves of Harvey Mudd College provided expert consultation on the biology domain.

REFERENCES

Acovelli, M., & Nowakowski, A. (1994). The Business Practices Course: Self-study learning reengineered. *Educational Technology, 34*(9), 21–27.

Ashley, K. D., & Aleven, V. (1992). Generating dialectical examples automatically. In *AAAI–92: Proceedings of the 10th National Conference on Artificial Intelligence* (pp. 654–660). San Jose, CA: AAAI Press/MIT Press.

Barrows, H. S. (1986). A taxonomy of problem-based learning methods. *Medical Education, 20,* 481–486.

Barrows, H. S., & Tamblyn, R. (1980). *Problem-based learning: An approach to medical education.* New York: Springer.

Bell, B., Bareiss, R., & Beckwith, R. (1993–1994). Sickle Cell Counselor: A prototype goal-based scenario for instruction in a museum environment. *The Journal of the Learning Sciences, 3,* 429–454.

Brown, J. S., Collins, A., & Duguid, P. (1989). Situated cognition and the culture of learning. *Educational Researcher, 18*(1), 32–42.

Burke, R. (1993). *Representation, storage and retrieval of tutorial stories in a social simulation.* Unpublished doctoral dissertation, Northwestern University, Evanston, IL.

Burke, R., & Kass, A. (1992). Integrating case presentation with simulation-based learning-by-doing. In *Proceedings of the 14th Annual Conference of the Cognitive Science Society* (pp. 629–634). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Carbonell, J. R. (1970a). AI in CAI: An artificial intelligence approach to computer-assisted instruction. *IEEE Transactions on Man–Machine Systems, 11*(4), 190–292.

Carbonell, J. R. (1970b). *Mixed-initiative man–computer instructional dialogues.* Unpublished doctoral dissertation, Massachusetts Institute of Technology, Cambridge.

Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science, 13,* 145–182.

Clancy, W. J. (1987). *Knowledge-based tutoring: The GUIDON program.* Cambridge, MA: MIT Press.

Cleave, J. B., Edelson, D. C., & Beckwith, R. (1993, April). *A matter of style: An analysis of student interaction with a computer-based learning environment.* Paper presented at the annual meeting of the American Educational Research Association, Atlanta, GA.

Coderre, B. (1988). Modeling behavior in Petworld. In C. Langton (Ed.), *Artificial life* (pp. 407–419). Reading, MA: Addison-Wesley.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In L. B. Resnick (Ed.), *Knowing, learning, and instruction: Essays in honor of Robert Glaser* (pp. 453–494). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Collins, A., & Stevens, A. L. (1982). Goals and strategies of inquiry teachers. In R. Glaser (Ed.), *Advances in instructional psychology* (Vol. 2). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Edelson, D. C. (1993). *Learning from stories: Indexing and reminding in a case-based teaching system for elementary school biology.* Unpublished doctoral dissertation, Northwestern University, Evanston, IL.

Fitzgerald, W. A. (1992). *Direct memory access parsing in the Creanimate biology tutor.* Unpublished manuscript, Institute for the Learning Sciences, Northwestern University, Evanston, IL.

Fitzgerald, W. A. (1995). *Building embedded conceptual parsers.* Unpublished doctoral dissertation, Northwestern University, Evanston, IL.

Karakotsios, K. (1992). SimLife [Computer software]. Walnut Creek, CA: Maxis.

Kass, A., Burke, R., Blevis, E., & Williamson, M. (1993–1994). Constructing learning environments for complex social skills. *The Journal of the Learning Sciences, 3,* 387–428.

Kass, A., Dooley, S., Luksa, F., & Conroy, C. (1994). Using broadcast journalism to motivate hypermedia exploration. In T. Ottmann & I. Tomek (Eds.), *Educational multimedia and hypermedia annual, 1994* (pp. 317–322). Charlottesville, VA: Association for the Advancement of Computing in Education.

Keller, J. M. (1983). Motivational design of instruction. In C. M. Reigeluth (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 383–434). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Klein, G. A. & Calderwood, R. (1988). How do people use analogues to make decisions? In J. L. Kolodner (Ed.), *Proceedings of the DARPA Case-Based Reasoning Workshop* (pp. 209–223). San Mateo, CA: Kaufmann.

Kolodner, J. L. (1991, Summer). Improving human decision making through case-based decision aiding. *AI Magazine, 12*(2), 52–67.

Kolodner, J. L. (1993). *Case-based reasoning.* San Mateo, CA: Kaufmann.

Lepper, M. R., & Hodell, M. (1989). Intrinsic motivation in the classroom. In R. Ames & C. Ames (Eds.), *Research on motivation in education* (Vol. 3, pp. 73–105). New York: Academic.

Malone, T. (1981). Toward a theory of intrinsically motivating instruction. *Cognitive Science, 4,* 333–369.

Martin, C. (1989). Case-based parsing. In C. K. Riesbeck & R. C. Schank (Eds.), *Inside case-based reasoning* (pp. 319–352). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Martin, C. E. (1990). *Direct memory access parsing.* Unpublished doctoral dissertation, Yale University, New Haven, CT.

McNair, M. P. (Ed.). (1954). *The case method at the Harvard Business School: Papers by present and past members of the faculty and staff.* New York: McGraw-Hill.

Redlich, J. (1914). *The common law and the case method in American university law schools* (Bulletin No. 8). New York: Carnegie Foundation for the Advancement of Teaching.

Reed, A. Z. (1921). *Training for the public profession of the law* (Bulletin No. 15). New York: Carnegie Foundation for the Advancement of Teaching.

Resnick, M. (1988). LEGO, logo, and life. In C. Langton (Ed.), *Artificial life* (pp. 407–419). Reading, MA: Addison-Wesley.

Resnick, M. (1991). Animal simulations with *Logo: Massive parallelism for the masses. In J. A. Meyer & S. W. Wilson (Eds.), *From animals to animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior* (pp. 534–539). Cambridge, MA: MIT Press.

Riesbeck, C. K., & Schank, R. C. (1989). *Inside case-based reasoning.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Ross, B. H. (1989). Some psychological results on case-based reasoning. In K. Hammond (Ed.), *Case-based reasoning: Proceedings of a workshop on case-based reasoning* (pp. 144–147). San Mateo, CA: Kaufmann.

Schank, R. C. (1982). *Dynamic memory.* Cambridge, England: Cambridge University Press.

Schank, R. C. (1986). *Explanation patterns: Understanding mechanically and creatively.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Schank, R. C. (1991). *Case-based teaching: Four experiences in educational software design* (Tech. Rep. No. 7). Evanston, IL: Northwestern University, Institute for the Learning Sciences.

Schank, R. C., Fano, A., Bell, B., & Jona, M. (1993–1994). The design of goal-based scenarios. *The Journal of the Learning Sciences, 3,* 305–346.

Schoenfeld, A. H. (1992). Learning to think mathematically: Problem solving, metacognition, and sense-making in mathematics. In D. A. Grouws (Ed.), *Handbook of research on mathematics teaching and learning.* New York: Macmillan.

Stevens, A. L., & Collins, A. (1977). *The goal structure of a Socratic tutor.* Paper presented at the National Association for Computing Machinery Conference, Seattle, WA.

Travers, M. (1988). Animal construction kits. In C. Langton (Ed.), *Artificial life* (pp. 421–442). Reading, MA: Addison-Wesley.

Wenger, E. (1987). *Artificial intelligence and tutoring systems: Computational and cognitive approaches to the communication of knowledge.* Los Altos, CA: Kaufmann.

Williams, S. M. (1992). Putting case-based instruction into context: Examples from legal and medical education. *The Journal of the Learning Sciences, 2,* 367–427.

APPENDIX
Selected Transcripts

This appendix contains selected portions of transcripts for three of the clusters that were identified in the analysis of patterns of use among students. The transcripts were selected to be illustrative of the features that distinguish the particular clusters.

Balanced Users

On average, balanced users took advantage of more of the aspects of Creanimate than did users in the other groups. In this transcript, the user types in three different answers for the explanation question and sees several video clips.

Student: **A butterfly that can dance**
[Picture: Butterfly]
How do you want to change your butterfly so it can dance?
    Change its ...
    Give it something new.
    Use its current ...
    Give me some suggestions.
Student: **Give it something new**
Ok, what should we give your butterfly to help it dance?
    Give it ...
    Make it ...
    Give me some suggestions.
Student: **Give it *wings***
That's a good idea. Sage grouses use their feathered wings to help them dance.
Do you want to see that?
Student: **Yes**
[Video: "Grouse Dances"]
Sage grouses are not the only animals that dance by using their wings. For example, ostriches also use their wings to help them dance.
This is a funny video.
Would you like to see that?
Student: **No**
So, your butterfly could use wings to help it dance. You can commit to that now, or we can look at some more ways animals dance and you can decide later.
What would you like to give it?

Wings
Show me more ways.
Student: **Show me more ways**
Can you think of another way to change your butterfly so it can dance?
Student: **Give it something new**
Ok, what should we give your butterfly to help it dance?
Student: **Give it *feathers***
I don't know of any animals that have feathers so they can dance. But, I do know some other things that feathers are used for.
For example, night jar birds use their feathers to help them camouflage themselves.
Would you like to see a video about that?
Student: **No**
That's the only thing that I know of that feathers are used for.
Those are all the reasons I know for having feathers. Are you sure you would like your butterfly to have feathers or would you like to see some other ways that animals are able to dance?
Stick with feathers
See ways that animals dance
Student: **See ways that animals dance**
Can you think of another way to change your butterfly so it can dance?
Student: **Give it *legs***
That's a good idea. Whooping cranes use their long legs to help them dance. Is that something you would like to see?
Student: **Yes**
[Video: "Cranes Dance"]
Whooping cranes are not the only animals that dance by using their legs. For example, gulls also use their legs to help them dance.
I have a funny video about that.
Do you want to see that ?
Student: **No**
So, your butterfly could use legs to help it dance.
What would you like to give it?
Legs
Feathers
Wings
Show me more ways.
Student: **Show me more ways**
Can you think of another way to change your butterfly so it can dance?
Student: **Give me some suggestions**
...

Video Hoppers

The following portion of a transcript is characteristic of the two video hoppers. The student never types in any input, watches one of the video clips three times in a row, and uses the "What other videos can I see?" option to see all the available video clips.

> ...
> We need a reason for your frog to dance. Is there a reason you want your frog to dance?
> So it can ...
> Give me some suggestions.
> Student: **Give me some suggestions**
> Bees dance. Do you know why bees dance?
> So they can ...
> Show me a video.
> What other videos can I see?
> Skip this.
> Student: **What other videos can I see?**
> Which would you like to see next?
> bee
> ostrich
> crowned crane
> whooping crane
> gull
> Student: **Ostrich**
> Do you know why ostriches dance?
> (I think this video is funny.)
> So they can ...
> Show me a video.
> Skip this.
> Student: **Show me a video**
> *[Video: "Ostrich Dances"]*
> Now, do you know why ostriches dance?
> So they can ...
> Show me the video again.
> Tell me.
> Student: **Tell me**
> Ostriches dance to attract their mates.
> That last video made me think of another one. This one is pretty good. I like this one because it has ferocious animals in it.

Bears also get mates. Only, instead of dancing to attract their mates, they swat to fight for territory.
Is that something you would like to see?
Student: **Yes**
[Video: "Bears Mauling Each Other"]
Ostriches are not the only animals that dance in order to attract their mates.
For example, crowned cranes also dance to attract their mates.
This is a colorful video. This video also has funny looking birds in it.
Do you want to see that?
Student: **What other videos can I see**
I can show you videos about all of these animals that dance.
Which would you like to see first?
  crowned crane
  whooping crane
  gull
Student: **Crowned crane**
[Video: "Crane Dances"]
Whooping cranes also dance to attract their mates.
I have a cool video about that. This has frantic dancing in it.
Would you like to see that video again?
Student: **What other videos can I see?**
Which would you like to see next?
  whooping crane
  gull
Student: **Whooping crane**
[Video: "Cranes Dance"]
Gulls also dance to attract their mates.
This next video is really funny.
Would you like to see that again?
Student: **No**
So, ostriches dance to attract their mates. Your frog could dance to attract its mate, too. Do you know how frogs currently attract their mates?
  They …
  I don't know.
Student: **I don't know**
Frogs attract their mates by croaking.
So, a frog might dance to attract its mate. Why would you like your frog to dance?
  Attract its mate
  Show me more reasons.
Student: **Show me more reasons**

Well, that's the only reason I know for dancing, but I know about some things that are like dancing. Maybe one of them will give you a reason you'll like for your frog to dance.
Barking is like dancing, because they are both ways that animals send signals.
Hyenas bark. Do you know why hyenas bark?
(I have an interesting video about that. This video also has lions in it.)
  So they can …
  Show me a video.
  Skip this.
Student: **Show me a video**
[Video: "Hyenas and Lion"]
Now, do you know why hyenas bark?
  So they can …
  Show me the video again.
  Tell me.
Student: **Show me the video again**
[Video: "Hyenas and Lion"]
Now, do you know why hyenas bark?
Student: **Show me the video again**
[Video: "Hyena and Lion"]
Now, do you know why hyenas bark?
Student: **Tell me**
Hyenas bark to warn their friends about danger.
…

Resisters

Resisters appeared to either not understand or not accept the basic flow of Creanimate. Characteristic of resisters was frequent use of the "Start over" and "Skip this" buttons and of anthropomorphic changes for their animals. In this transcript, the student resists the system's offer to make one change at a time by asking for two changes at once and then asks for a butterfly that could emulate a popular musician and dancer, MC Hammer.

Student: **A butterfly that can fight**
[Picture: Butterfly]
That's a good idea, is there a reason why you want your butterfly to fight?
Button Press: ***BACKUP***
How would you like to change your butterfly?
  So it can …
  Give it …

So it is ...
Student: **So it can *dance and fight***[10]
[Picture: Butterfly]
How do you want to change your butterfly so it can dance?
Change its ...
Give it something new.
Use its current ...
Give me some suggestions.
Student: **Give it something new**
Ok, what should we give your butterfly to help it dance?
Give it ...
Make it ...
Give me some suggestions
Student: **Give it *MC Hammer legs***[11]
That's a good idea. Sage grouses use their legs to help them dance.
Would you like to see a video about that?
Student: **No**
So, your butterfly could use legs to help it dance.
You can commit to that now, or we can look at some more ways animals dance and you can decide later.
What would you like to give it?
Legs
Show me more ways.
Student: **Legs**
OK, your butterfly will use its new legs to help it dance.
We need a reason for your butterfly to dance. Is there a reason you want your butterfly to dance?
So it can ...
Give me some suggestions.
Student: **So it can do the MC Hammer**
...

---

[10] Because the dialogue manager can only deal with one modification at a time, it ignores "and fight."

[11] Because the parser can not interpret "MC Hammer," it ignores it.

Copyright © 2002 EBSCO Publishing