Chapter 2
The Eliza Effect

Meeting Eliza

When I was a teenager—in the 1980s—my mother bought a personal computer. It was an impressive machine for the day, decked out with two floppy drives, a dot matrix printer, a Hayes modem, and a monochrome amber display.

At first I only used the machine for some minor programming experiments (in Basic and later Pascal), writing for school (in WordStar), and a few games. But that mysterious modem sat there. Probably intended to let my mother exchange data with the big Digital Equipment Corporation machines she had in her university lab, I knew modems could also be used for other things.

This was about a decade before the Internet began to make its way into homes like ours, and I had no interest in the manicured gardens of services like the Source or CompuServe. Rather than any long-distance journey, I wanted to use the modem to explore the local wilderness, to visit the unruly bulletin board system (BBS) scene sprouting in the dens and basements of my neighbors’ homes.

While largely forgotten today, a BBS was the online destination of choice for 1980s teenagers. Most were run by individuals out of their homes: computer enthusiasts with machines much more powerful than ours, hooked to one or more dedicated phone lines. A user like me could call into a BBS, read messages, leave messages, download and upload files, play text-based games, and (if

1. For more on BBS culture, I highly recommend Jason Scott’s BBS: The Documentary (2005).
Expressive Processing

2. *Eliza* was not the first system to give audiences the impression of meaningful exchange with a computer. Matthew Kirschenbaum’s *Mechanisms* (2008) offers the intriguing example of “Professor RAMAC”—a …

Notes continued at end of this chapter.

the owner of the BBS was at the computer, or if someone called in to one of the other phone lines) have real-time conversations, with total strangers, in text. In other words, the BBS wasn’t just a file repository. It was a window into what has now become obvious: the incredible social potential of combining computers and networks, which has given us email, instant messaging, wikis, blogs, social networking web sites, and much more.

Given the glimpse of this potential, a BBS with multiple lines could feel a little lonely when no one else was on. But then one day I was over at the house of a childhood friend (we no longer went to the same school), and he showed me that on his computer, conversation was always waiting. He showed me a program he’d downloaded from a BBS. He introduced me to *Eliza*.

**Eliza Today**

*Eliza*—or more properly, *Eliza/Doctor*—is a groundbreaking system created by computer science researcher Joseph Weizenbaum at MIT in the mid-1960s. In the two decades between when Weizenbaum created the system and I experienced it at my friend’s house, it had become one of the world’s most famous demonstrations of the potential of computing. First unveiled two years before HAL 9000’s screen debut in *2001: A Space Odyssey*, it seemed that *Eliza* made it possible to have a real conversation with a computer.²

In the computer science literature, under the name *Eliza*, Weizenbaum’s system is a contribution to the field of natural language processing. On the other hand when *Eliza* plays *Doctor* it is a well-known computer character, famous far beyond computer science, often also known
The *Eliza* Effect

by the name Eliza. And *Eliza* has a third common usage in the computer world as well: the *Eliza* effect. This has generally been a term used to describe the not-uncommon illusion that an interactive computer system is more “intelligent” (or substantially more complex and capable) than it actually is. One of my purposes in this chapter is to revisit the *Eliza* effect and give it a further nuance so that it names not only this initial illusion but also the authorial choice that comes with it: severely restricted interaction (on the one hand) or eventual breakdown that takes a form based on the actual underlying processes (on the other).

In the next chapter, with an examination of the *Eliza* effect as background, I will look at the options selected by today’s authors of digital fictions—particularly for computer games. These generally put aside the *Eliza* effect in favor of systems that more clearly communicate their structures to audiences. Yet there are two problems with these that I will consider. Some of them employ processes that, while legible, tend toward a non-*Eliza* form of breakdown. Others, while avoiding breakdown, have low ambitions in their use of computational processes. More ambitious routes will be the focus of the remaining chapters of this book.

But for now I’ll start with the illusion.

The *Eliza* Effect

I don’t remember the specifics of my first conversation with *Eliza*, but here is *Eliza*’s most famous exchange (with the computer’s responses in all capital letters):

> Men are all alike.

IN WHAT WAY
Expressive Processing

They’re always bugging us about something or other.
CAN YOU THINK OF A SPECIFIC EXAMPLE
Well, my boyfriend made me come here.
YOUR BOYFRIEND MADE YOU COME HERE
He says I’m depressed much of the time.
I AM SORRY TO HEAR YOU ARE DEPRESSED
It’s true. I am unhappy.
DO YOU THINK COMING HERE WILL
HELP YOU NOT TO BE UNHAPPY
I need some help, that much seems certain.
WHAT WOULD IT MEAN TO YOU
IF YOU GOT SOME HELP
Perhaps I could learn to get along with my mother.
TELL ME MORE ABOUT YOUR FAMILY
My mother takes care of me.
WHO ELSE IN YOUR FAMILY TAKES CARE OF YOU
My father.
YOUR FATHER
You are like my father in some ways.
WHAT RESEMBLANCE DO YOU SEE
You are not very aggressive but I think
you don’t want me to notice that.
WHAT MAKES YOU THINK I AM
NOT VERY AGGRESSIVE
You don’t argue with me.
WHY DO YOU THINK I DON’T ARGUE WITH YOU
You are afraid of me.
DOES IT PLEASE YOU TO BELIEVE
I AM AFRAID OF YOU
My father is afraid of everybody.
WHAT ELSE COMES TO MIND WHEN
YOU THINK OF YOUR FATHER
Bullies.
DOES THAT HAVE ANYTHING TO DO WITH THE FACT THAT YOUR BOYFRIEND MADE YOU COME HERE (Weizenbaum 1966, 37)

As you might imagine, I was impressed. And I certainly wasn’t the first. Weizenbaum’s employers at MIT had seized on *Eliza* as an accessible demonstration of the potential of computing, exciting to those without the specialized knowledge needed to appreciate much of the ongoing research. The computer world agreed. In fact, when the first demonstrations of the ARPANET (the predecessor of the Internet) were carried out, some of the terminals ran *Eliza* from a remote location (Leonard 1997, 52). Given the excitement produced by *Eliza*, astronomer and science popularizer Carl Sagan suggested that the future might hold “a network of computer psychotherapeutic terminals, something like arrays of large telephone booths” (Weizenbaum 1976, 5).

Originally *Eliza* ran on a computer less powerful than the one in my mobile phone, though at the time it was one of the most advanced at MIT. Rather than being hooked up to a monitor, keyboard, and mouse—or HAL’s microphone, speaker, and camera—*Eliza* was experienced through something like a typewriter, allowing the computer to type to the user and the user to type back. People could type anything they wanted to *Eliza*, and the software would respond in different ways depending on the currently active script.

This last word, “script,” is important. There is nothing magical about *Eliza*—it is simply a bundle of data and processes, and pretty simple processes at that. Each time that *Eliza* runs, it uses a particular script to guide its behavior. The example conversation given was created
Expressive Processing

4. This article is not only a good, clear source of explanation for Eliza’s processes (and the most-cited publication about Eliza in the computer science literature). It also served as the basis for many homegrown versions of Eliza created at …

5. While the original script text is in all capitals, I am regularizing it here. Also, this description focuses on the core processes at work in Eliza to describe them all would make this section as long as Weizenbaum’s paper.

Notes continued at end of this chapter.

using Eliza’s most famous script, Doctor, which causes the software to parody the conversational patterns of a nondirective therapist during an initial visit. All of this is described in Weizenbaum’s 1966 article in Communications of the ACM.¹

How Eliza Works

A session with Eliza can begin with a greeting. Weizenbaum’s Doctor script starts with: “How do you do. Please tell me your problem.” After this point Eliza will not take the initiative again—only respond.

Each time an audience member types something, Eliza examines it, looking for words that have entries in the currently active script. Some of the words are keywords, which I will discuss further below. Some words are marked for simple substitution. For example, when Eliza runs the Doctor script, these substitutions switch all the first-person pronouns for second-person pronouns (“I” becomes “you”) and vice versa (“yourself” becomes “myself”). “Well, you are very helpful,” for instance, would become “Well, I am very helpful.” A word can be both substituted and used as a keyword.

Periods and commas are treated as delimiters. If a period or comma is encountered, Eliza checks to see if a keyword has already been found. If one has, then everything that the audience member typed after the delimiter is discarded. If no keyword has yet been found, everything before the delimiter is discarded. For example, “Well, I am very helpful” would become “I am very helpful.”

Each keyword has a priority level or rank. When the first keyword is found in a text, it is added to a “keystack.” Each time another keyword is found, the rank of the
new keyword is compared with that of the highest-rank keyword yet found. If the new word has a higher rank it is added to the top of the stack; otherwise it is added to the bottom of the stack. The result, at the end of scanning a text for keywords, is that the highest-rank keyword is at the top of the stack.

After keyword scanning, the next step is to find a “decomposition rule” that matches the postsubstitution version of what the audience member typed (minus any parts discarded because of commas or periods). Decomposition rules are associated with keywords, so this search begins by popping the top keyword off the stack. (I'll soon discuss what happens if there is nothing on the stack.) *Eliza* tries out the decomposition rules associated with that keyword, looking for one with an appropriate pattern of words in combination with the wild card “0” (which is equivalent to “any number of words, including zero”). For example, if the keyword is “you,” the first decomposition rule is:

(0 I remind you of 0)

This wouldn't match the sample phrase from above, “I am very helpful.” The first rule would only match a postsubstitution phrase like, “With that last statement I remind you that I are a machine,” or (because 0 can represent zero words), “I remind you of the happier days of your childhood.” The next decomposition rule for the “you” keyword, though, would match the example phrase used earlier:

(0 I are 0)

Once a match is found, the decomposition rule does what it is named for: it decomposes the phrase into parts. This decomposition is based on the number of words and the number of 0s in the rule. For instance, “I am very
helpful” would become:

(1) empty (2) I (3) are (4) very helpful.

All that remains, at this point, is to decide what to say back to the audience. Each decomposition rule can have a set of reassembly rules associated with it. An example reassembly rule (for the decomposition “0 I am 0”) is:

(What makes you think I am 4)

This would result in the response, “What makes you think I am very helpful.” Once they were used during any run of Eliza, the reassembly rules were marked so that all the reassemblies associated with a particular decomposition would be cycled through before any were repeated.

This is the bulk of Eliza’s approach. There are a few refinements, such as techniques that allow scripts to share rules between keywords (e.g., mother and father). But from this the dominant operational logic at work in Eliza should be clear: transformation. Each statement by an Eliza script is the result of a multistep transformation of the most recent audience statement.

Many of these transformations include a verbatim portion of the statement being transformed, as outlined above. Others are more indirect transformations, as when “Perhaps I could learn to get along with my mother” is translated into “Tell me more about your family.” The real transformation difficulty comes, however, when no keywords—a central aspect of Eliza’s transformation logic—are found in the audience’s most recent text. For example, this is the situation when Eliza/Doctor asks, “What else comes to mind when you think of your father,” and receives the response, “Bullies.”

One set of possible transformations, in these cases, can be defined in the script based on a special keyword
called “none.” In Doctor these are content-free phrases such as “Please go on” and “I see.” Yet some of the most surprising moments with Eliza come from another special structure called “memory.” This structure works in combination with a particular keyword. In the case of Doctor it works with the keyword “my” and the decomposition “0 your 0.” When this keyword is selected, a response is generated in the usual way. In addition, another reassembly may be created and stored in a queue for later use when no keyword is found. In Doctor these reassemblies are “Lets discuss further why your 3,” “Earlier you said your 3,” and “Does this have anything to do with the fact that your 3”—the last of these we saw above, with 3 being “boyfriend made you come here.” In other words, the surprising punch line in the above conversation was nothing but a fallback, a default move that Eliza can make when the most recent response has no identifiable keywords.

Here we can see what moved Weizenbaum to write that “once a particular program is unmasked, once its inner workings are explained in language sufficiently plain to induce understanding, its magic crumbles away; it stands revealed as a mere collection of procedures, each quite comprehensible” (1966, 36). But when we interact with a piece of software, we don't necessarily achieve such understanding, we don't get a clear picture of how it actually operates internally. And Dr. Sagan and I were not alone regarding Eliza/Doctor; its audiences often developed mistaken ideas about its internals. They assumed that since the surface appearance of an interaction with the program could resemble something like a coherent dialogue, internally the software must be complex. Some thought it must be
Expressive Processing

something close to the fictional HAL: a computer program intelligent enough to understand and produce arbitrary human language. This happened so frequently, and was so striking, that in computer science circles people began to use a specific term for this kind of misunderstanding, one mentioned earlier in this chapter: the Eliza effect.

Why Eliza Worked
The stories of Eliza/Doctor's effectiveness are many. Weizenbaum himself tells us that “my secretary, who had watched me work on the program for many months and therefore surely knew it to be merely a computer program, started conversing with it. After only a few interchanges with it, she asked me to leave the room” (1976, 6). There is also the oft-repeated tale, which exists in many versions, of the boss thinking he is communicating with a subordinate while actually conversing with Eliza/Doctor.

Vice president: Do you think I can use the computer this morning?
Teletype: WHY DO YOU ASK?
Vice president: I might be able to make some additional telcomp (computer system) sales.
Teletype: WHY AREN'T YOU SURE?
Vice president: My prospective customers have never seen a demonstration of the system.
Teletype: DOES THIS MEAN VERY MUCH TO YOU?
Vice president: Of course it does.
Teletype: YOU SEEM VERY POSITIVE ABOUT THAT.
Vice president: This is ridiculous.
Teletype: WHAT DO YOU MEAN THIS IS RIDICULOUS?
(Murray 1997, 7)

As a high school student, I found that Eliza/Doctor
The Eliza Effect

created an effective initial illusion for the same reasons it worked for the iconic “emotional” secretary and “clueless” boss evoked by the above stories. First, all of us were accustomed to text-only computing and to having conversations with other people within that environment. Second, Eliza/Doctor makes a remarkably good match between process and data. The situation of the initial visit to the therapist, the clever writing in the reassemblies and nonresponses, and the well-chosen keywords do the most possible to leverage the simple linguistic tricks available via Eliza’s transformation processes. (In fact, there were other Eliza scripts created besides Doctor, but none of them became nearly as well known or widespread.) Third, for myself, the secretary, and the boss, this was one of our first experiences with computer characters. But all three of these reasons are only Eliza/Doctor’s specific nuances on a much more general phenomenon: when a system is presented as intelligent and appears to exhibit intelligent behavior, people have a disturbingly strong tendency to regard it as such.8

This phenomenon derailed Weizenbaum’s career. He came to focus his work on the conceptual mismatch that gives the Eliza effect its name and specifically on how it could “induce powerful delusional thinking in quite normal people” (1976, 7). Weizenbaum wrote a book dedicated to demonstrating that the internals of computers aren’t magical, and that we do ourselves a disservice when we assume that human beings are so mechanical that we could or should have our intelligence matched by computational machines. In a sense, he moved from being a computer scientist to being one of the first knowledgeable critics to interrogate the cultures

8. Mark J. Nelson, in the blog-based peer review of this book, urged me to clarify the fact that this isn’t only true of computer systems—as demonstrated by an example later in this chapter. Harold Garfinkel’s yes/no therapy experiment.
of computing and artificial intelligence.

Following Weizenbaum, a number of other authors saw the Eliza effect as important to address in understanding our relationship with computers and our culture more generally. A decade after Weizenbaum’s book, Lucy Suchman published Plans and Situated Actions (1987), in which she sees Eliza/Doctor as an iconic example in human-computer interaction of the broad phenomenon of treating systems as intelligent based on limited evidence. Specifically, she discusses what ethnomethodologist Harold Garfinkel (citing Karl Mannheim) has called the documentary method of interpretation.

Suchman presents one of Garfinkel’s experiments as a demonstration of the idea that people tend to “take appearances as evidence for, or the document of, an ascribed underlying reality, while taking the reality so ascribed as a resource for the interpretation of the appearance” (23). In this experiment student subjects were introduced to a new kind of therapy in which they asked yes/no questions about their personal problems. These were answered by “counselors” who were not visible to the subjects. Unbeknownst to the subjects, the counselors answered each question randomly.

After the experiment, the students were found to have constructed stories that made sense of each string of answers as a coherent exchange and set of advice. This happened even when, as would almost inevitably happen in such a circumstance, the answers given were self-contradictory. The apparent contradictions were explained away by the experimental subjects as revised views based on further information, evidence of a deeper agenda on the part of the counselor or something else.
that fit with the frame of the therapeutic situation.

Yet another decade later, Janet Murray revisited *Eliza/Doctor* in *Hamlet on the Holodeck* (1997). Here she makes a crucial turn for my purposes, seeing the project in terms of media. She views *Eliza/Doctor* as a dramatic character, Weizenbaum as a kind of playwright, and the source of the *Eliza* effect in such cases as “the human propensity to suspend disbelief in the presence of a persuasive dramatic presence” (224). Murray makes visible what wasn’t clear in previous accounts: that people enjoy *Eliza*. Initial interactions are often quite fun and engaging, and create the sense of open-ended possibilities with an interesting character.

But that first impression doesn’t necessarily last.

## Revisiting the *Eliza* Effect

Being a teenager, after my initial interaction with *Eliza/Doctor*, I did the natural thing: I started to play with it. (Software, of course, doesn’t need to be a game to inspire play.) I don’t remember the exact exchange, but I know that I soon discovered the reason Murray doesn’t go far into talking about ongoing interaction with *Eliza/Doctor*, just as Suchman and Weizenbaum don’t. If they did, it would complicate their discussions. This is because, while the initial experience of *Eliza/Doctor* can create the surface impression of an incredibly complex internal system, sustained interaction with the system, the verbal back-and-forth, invites play—and linguistic play with *Eliza/Doctor* quickly begins to destroy the illusion. In other words, precisely the open-ended textual interaction that helped foster the illusion of internal complexity and intelligence enables play that draws attention to the system’s rote
Expressive Processing

10. Turkle comments that “some people embark on an all-out effort to ‘psych out’ the program, to understand its structure in order to trick it and expose it as a ‘mere machine.’ Many more do the opposite. . . . They didn’t ask questions that they knew . . .

11. Suchman argues that Garfinkel’s experiment lends support to Weizenbaum’s view that the feeling of intelligence in conversations with Eliza/Doctor comes from the work of the audience. Further, she contends that . . .

Notes continued at end of this chapter.

simplicity, its distance from human interaction. Before long, many exchanges begin to look like this:

You are going to repeat what I say
in the form of a question

WHAT MAKES YOU THINK I AM GOING TO REPEAT
WHAT YOU SAY IN THE FORM OF A QUESTION

Sherry Turkle, in The Second Self, notes this phenomenon, writing, “As one becomes experienced with the ways of Eliza, one can direct one’s remarks either to ‘help’ the program make seemingly pertinent responses or to provoke nonsense” (1984, 39). Turkle’s book is based on interviews and observations—and she found that some of her subjects worked to keep the illusion going, while others sought to maximize Eliza’s nonsense. Both strategies, though, depended on coming to understand something of the internal processes at work and shaping surface interaction accordingly. Even working to maintain the illusion required a type of seeing past it, something that those who discuss the Eliza effect rarely acknowledge.

The Eliza Breakdown

From my point of view, what Turkle describes points toward a further lesson of Garfinkel’s yes/no therapy experiment. For Suchman, this experiment demonstrates the importance of ethnomethodology and the documentary hypothesis for understanding Eliza/Doctor and human-computer interaction. And certainly it is essential to understand that Eliza/Doctor succeeds, to the extent that it does, because it plays on the interpretative expectations brought to each interaction by audience members. But for my purposes here, Garfinkel’s experiment also serves to illustrate something rather different: the Eliza effect
can be shielded from breakdown by severely restricting interaction. The experiment allowed the subjects to maintain the illusion that something much more complex was going on inside the system (a human considering her problems seriously and answering questions thoughtfully, rather than random yes/no answers) because the scope of possible responses was so limited. If it had been expanded only slightly—say, to random choice between the responses available in a “magic eight ball”—almost any period of sustained interaction would have shattered the illusion through too many inappropriate responses.

When breakdown in the Eliza effect occurs, its shape is often determined by that of the underlying processes. If the output is of a legible form, the audience can then begin to develop a model of the processes. This is what Turkle observes in those interacting with Eliza/Doctor: from the shape of the breakdown they begin to understand something of the processes of the system—and then employ that knowledge to help maintain or further compromise the illusion.

In this context, it is interesting to note that most systems of control that are meant to appear intelligent have extremely restricted methods of interaction. In some cases the reasons for this are quite obvious. If the public were allowed playful interaction with software that identifies possible targets for financial surveillance, the shape of the underlying system would become relatively apparent, making it possible to “game” the system. At the same time, this restricted interaction also serves to maintain the Eliza effect for software that is not nearly as intelligent as the public has been asked to believe.

Further, within a rather different community, this
choice—between severely restricted interaction and the boom/bust of illusion followed by breakdown—presents no good options to those with an interest in creating digital fictions.\textsuperscript{12} So while some have argued that it is best to capitalize on the \textit{Eliza} effect, depending on temporary illusion and the willful suspension of disbelief to carry the day, most digital fiction authors employ a different approach: exposing important elements of the structures of their processes to the audience from the outset. This allows for interaction that matches the process employed, and avoids the \textit{Eliza} illusion and breakdown. Still, as I will discuss next, the most common of these approaches suffer from limitations of their own.

Finally, I should mention that some authors—such as Jeremy Douglass (2007)—assert that breakdown can be an interesting mode for digital fictions. And certainly breakdowns can be fascinating. On a linguistic level, for example, we’re attracted to study every form of breakdown from occasional slips of the tongue to hemmorage-induced aphasia.

What breakdowns can do—in the case of \textit{Eliza/Doctor}, linguistic slips, and neurological problems alike—is give us some insight into the shape of the underlying system processes. This fact points to the reason why I still talk with people online (even if I no longer dial in to a BBS to do so) but I no longer play with \textit{Eliza} in my spare time: a system prone to breakdown is only as interesting as the shape of the processes that the breakdowns partially reveal. And as shown earlier in this chapter, the \textit{Eliza} system processes are mostly a relatively uninteresting set of substitutions. We can do better.
The *Eliza* Effect

Notes

1. For more on BBS culture, I highly recommend Jason Scott’s *BBS: The Documentary* (2005).

2. *Eliza* was not the first system to give audiences the impression of meaningful exchange with a computer. Matthew Kirschenbaum’s *Mechanisms* (2008) offers the intriguing example of “Professor RAMAC”—a four-ton IBM machine anthropomorphized at the 1958 World’s Fair—which used a then-revolutionary hard drive to provide question-answering access to a collection of historical facts.

3. Erik Loyer, in reimplementing the *Eliza* system for a project on which we were collaborating for the digital media journal *Vectors*, noted that this conversation appears to be edited. Unless the *Eliza* system had undocumented features, its response to “You are not very aggressive but I think you don’t want me to notice that” would have been “WHAT MAKES YOU THINK I AM NOT VERY AGGRESSIVE BUT YOU THINK I DON’T WANT YOU TO NOTICE THAT” rather than the snappier “WHAT MAKES YOU THINK I AM NOT VERY AGGRESSIVE.”

4. This article is not only a good, clear source of explanation for *Eliza’s* processes (and the most-cited publication about *Eliza* in the computer science literature). It also served as the basis for many homegrown versions of *Eliza* created at computing centers across the country in the years that followed. This was still decades away from when researchers and hobbyists could easily transport software to one another over computer networks—so paper publications such as *Communications of the ACM* (for researchers) and *Byte* (for hobbyists) often included all the information needed (including, in the case of Weizenbaum’s article, the complete *Doctor* script) for reimplementing the software locally.

5. While the original script text is in all capitals, I am regularizing it here. Also, this description focuses on the core processes at work in *Eliza* to describe them all would make this section as long as Weizenbaum’s paper.

6. Question marks were not used because they were interpreted as “line delete” on the computer system that Weizenbaum was using.

7. This is the version of the story reported in Janet Murray’s *Hamlet on the Holodeck*. In addition to this version, many clearly apocryphal versions circulate, including one in which Weizenbaum is a participant in the events. This version itself cannot be a verbatim conversation with *Eliza*, at least not as the system existed at the time of Weizenbaum’s 1966 paper. That paper’s *Doctor* script, for example, contains the responses “YOU AREN’T SURE” and “WHY THE UNCERTAIN TONE”—but not “WHY AREN’T YOU SURE.” Beyond differences in wording, it is also worth remembering the previous note: the system used by Weizenbaum did not support question marks, which appear throughout this transcript.

8. Mark J. Nelson, in the blog-based peer review of this book, urged me to clarify the fact that this isn’t only true of computer systems—as demonstrated by an example later in this chapter: Harold Garfinkel’s yes/no therapy experiment.

9. To be fair, at the time of Weizenbaum’s initial observations, almost no one could experience ongoing interaction with *Eliza/Doctor*, due to the limited availability of computing resources. As Weizenbaum notes,
Expressive Processing

“Since the subject cannot probe the true limits of Eliza’s capabilities (he has, after all, only a limited time to play with it, and it is constantly getting new material from him), he cannot help but attribute more power to it than it actually has” (1976, 191).

10. Turkle comments that “some people embark on an all-out effort to ‘psych out’ the program, to understand its structure in order to trick it and expose it as a ‘mere machine.’ Many more do the opposite. . . . They didn’t ask questions that they knew would ‘confuse’ the program, that would make it ‘talk nonsense’” (40). Turkle attributes this to a desire to “maintain the illusion that Eliza was able to respond to them.” It is also entirely in line with Murray’s interpretation of Eliza as a media experience, however, with the audience shaping their interaction to help maintain the willful suspension of disbelief.

11. Suchman argues that Garfinkel’s experiment lends support to Weizenbaum’s view that the feeling of intelligence in conversations with Eliza/Doctor comes from the work of the audience. Further, she contends that the strongly situated understandings of the students (they interpreted the random series of yes/no answers based on assumed context) is a challenge not only to the strong structure-oriented assumptions of the social sciences but also those of cognitive science.

12. Except for that limited number of fictions that might want to explore one of these effects.
Chapter 3
Computer Game Fictions

Digital Fictions and the Eliza Effect
Imagine you’ve checked out your books and walked, through the damp twilight, to the bus shelter across the street. Its fluorescent tubes have flickered on—you can read the schedules and advertisements behind the Plexiglas, as well as the stickers scattered over them.

One rectangular sticker catches your eye. It has the name of no band, the number of no locksmith, the logo of no corporation, and no image of Andre the Giant. It's just a block of text. The first words read, “Why bomb libraries?”

The text is a passage from Implementation, a novel written in small chunks formatted to fit on mailing labels that can be fed through a standard laser printer. Implementation is a sticker novel, in one sense, and also a kind of digital novel. Its authors, Nick Montfort and Scott Rettberg (2004), don’t primarily distribute stickers. Instead, they distribute digital files, which others print on stickers and post in provocative locations. Readers sometimes encounter these stickers. But, like most stickers, they tend to be removed pretty quickly from the benches, doors, bathroom stalls, statue plaques, lampposts, bumpers, and other locations where they’re placed. What lasts longer are the photographs that people take of Implementation stickers in interesting positions, and it is through these images that most people experience Implementation, the files for these images dwarfing the other contents of the Implementation web site.

Encountering Implementation can be a mysterious experience. Finding a disconnected sticker of text, especially if well-placed, can border on disconcerting. And the images of placed
stickers can offer up their own mysteries: Where is that? What is that? But, despite the importance of digital technology for *Implementation*, the mystery is never about software. *Implementation* depends on digital processes—but ones we use every day, such that the processes of file downloading, laser printing, and photo sharing no longer attract our attention.

One might even argue that *Implementation* is not appropriate to discuss as a “digital” novel but rather one that is distributed and documented digitally. Regardless, in its use of digital processes, *Implementation* stands as a relatively extreme example of something true of many digital fictions: they avoid the dilemma of the *Eliza* effect by employing processes that are conceptually simple and familiar, and that are clearly exposed to the audience.

Different fictions approach this general strategy in different ways. Some, in a manner relatively close to *Implementation*, embed themselves in familiar digital contexts. An email novel, such as *Blue Company* by Rob Wittig (2001, 2002), depends on processes of email transmission and reading—but, like the processes of laser printing, these have become completely naturalized for many with computer access. Similarly, the familiar link-following functions of web browsers and web servers makes it possible to construct fictions from interconnected networks of web pages, as with *The Unknown* by William Gillespie, Scott Rettberg, Dirk Stratton, and Frank Marquardt (1999). Familiar processes are also the primary digital components of hybrid works such as the alternate reality game by Elan Lee, Sean Stewart, Jim Stewartson, and Jane McGonigal (2004) titled *I Love Bees*, which brought players into a fictional world composed of web
Computer Game Fictions

pages, email messages, phone calls, physical settings and elements, and live performance—but defined its novel processes as rules to be carried out by human participants (rather than by digital computation).

Another approach to digital fiction is more common than this sort of piggybacking on the processes of everyday information life, though. In this more common approach, digital fictions define their own versions of digital media processes in widespread use—or employ versions of these processes defined for use with digital media authoring tools. Such authoring tools range from Adobe’s Flash software (for interactive animations) to game engines (used repeatedly by the same developer or made available for commercial license) to tools identified with particular artistic communities (such as the Storyspace hypertext authoring system).

The most widespread fictions taking this approach are computer games. They tend to avoid the Eliza effect by employing versions of simple processes, familiar to those who play computer games within the same genre, and following conventions to expose the structure and actions of the underlying processes to their audiences. This chapter will look at two particularly well-crafted examples of computer game fiction—Star Wars: Knights of the Old Republic (Falkner, Gilmour, Hudson, et al. 2003), or KatOR, and Prince of Persia: The Sands of Time (Mechner, Mallat, Désilets, et al. 2003), or PoP—as well as two commonly employed operational logics that enable the fictions of computer role-playing games (RPGs). But, to understand all this, it is important to begin with a wider view of RPGs.
Expressive Processing

1. Some games, of course, can hardly be said to engage fiction at all. No one wants a story with their Tetris—and its "fictional world" of falling blocks barely supports the phrase. Given the sometimes-contentious nature of critical discussion ...

2. As tabletop RPGs have come to encompass much broader areas of life than combat, different games (and player groups) have diverged in their treatment of these elements. One movement is toward a generalization of the statistical models ...

Notes continued at end of this chapter.

Role-Playing Games

Games employ fiction in many ways.¹ The most story-ambitious genre of computer games is probably the computer RPG—a form that traces its roots back to a noncomputer form of gaming: the tabletop RPG (see sidebar: Computer and Tabletop Role-Playing Games).

The first tabletop RPG was Dungeons & Dragons, created by Gary Gygax and Dave Arneson and published in 1974 (Mona 2007). It grew out of a tabletop wargaming tradition in which maps were used to represent battlefields and miniatures representing units or individual combatants were placed on the maps. Players would then move figures and engage in battles by following sets of rules and consulting tables of numbers (the process and data of the system) with a random element at times introduced using dice. Dungeons & Dragons departed from this model by suggesting that each player take on a single character, that play sessions connect with one another in an ongoing campaign during which each character would develop, and that the game encompass much more of the characters' lives (and the fictional world in which they live) than raw combat.²

The result was the birth of a genre that is, when well played, undoubtedly the most successful combination of game and fiction now in existence. RPG systems have been created for a wide variety of fictional settings (some inspired by authors ranging from H. P. Lovecraft to William S. Burroughs), are played both as tabletop games and live performances, and embody a diversity of aesthetic goals (the design goals of Greg Costikyan’s games alone range from reproducing the wacky physical comedy of classic cartoons to evoking the alienation of Brechtian
Computer and Tabletop Role-Playing Games

Many computer RPGs trace their origins to tabletop RPGs—and sometimes the distance traced is quite short. Games such as BioWare’s Neverwinter Nights computer RPG (Oster et al. 2002) are licensed versions of the tabletop RPG Dungeons & Dragons (Gygax and Arneson 1974). Further, the more abstract underlying “d20 system” developed for the third edition of Dungeons & Dragons—but then adopted for many tabletop RPGs, from D&D publisher Wizards of the Coast (WotC) and others—has also been used by computer RPGs such as KotOR (Falkner et al. 2003), as pointed out by Jason Rhody in the blog-based peer review of this book. It is also interesting to note that WotC publishes a tabletop RPG based on the d20 system and set in the Star Wars universe, for which there is a campaign guide for KotOR (Thompson et al. 2008)—presenting a rare opportunity to compare two different implementations of the same fictional world in the same basic game system, one computer and one tabletop.

At the same time, it is also worth noting that computer games developed in parallel with tabletop RPGs, so the larger genre of computer RPGs might be seen as a convergence of the two traditions—rather than a translation of the tabletop genre into computer form. In the blog-based peer review for this book, Dennis G. Jerz pointed particularly to early computer games such as Hunt the Wumpus (Yob 1973) and Maze War (Colley and Thompson 1973–1974). Certainly these are important to consider. Still, as Matt Barton recounts in Dungeons & Desktops, “There really is no doubt that D&D played a vital role in the development of the first CRPG. Richard Garriott, creator of Akalabeth and Ultima, was himself a dedicated fan of the game” (2008, 13). That said, Barton also argues that the importance of a number of other influences on the computer RPG should be recognized, including the computer game Adventure (Crowther and Woods 1976), sports simulation games, tabletop war games (the tradition from which D&D emerged), and the writings of J. R. R. Tolkien.

The history we trace by this method is definitely a Western one, however. The history of the flourishing form of the Japanese RPG is rather different. As William Huber writes:

The Japanese reception of role-playing games includes a discontinuity: there is little sign of a culture of tabletop fantasy role-playing games in Japan before the production of computer-based games. Dungeons & Dragons was translated into Japanese in 1985, the same year that Dragon Quest was released: The Ultima and Wizardry games were already in widespread Japanese distribution by this time. . . . Thus the temporal framework for the RPG in Japan has been primarily computer-based from the outset: the evolution of RPG mechanics as a practice performed by a collaborating group of players does not exist in Japan. The “roles” of role-playing are categories of character function, and the player-driven theatrical aspects of pen-and-paper role-playing are instead replaced by story practices that owe as much to cinema and television as they do to military simulation. (2009)
Expressive Processing

3. The most famously Lovecraft-inspired game is *Call of Cthulhu* (Petersen 1981), while *Over the Edge* (Tweet 1992) is a game inspired by both Burroughs and Philip K. Dick. At the same time, as Jose Zagal reminded me in the blog-based peer …

4. Though many massively multiplayer online play groups, including some in which I have participated, include face-to-face interaction between some players. And there are a variety of other hybrid forms that combine elements of in-person and …

Notes continued at end of this chapter.

3. Most retain a role for one of the players as the “game master.” This player undertakes the advance planning for the current campaign, plays the roles of the non-player characters (NPCs), shapes events to ensure the type of story desired, and executes most of the rule processes. This leaves other players free to concentrate on the defining features of RPGs: playing their characters and collaboratively developing the fiction.

Of course, with computer RPGs the situation is somewhat different. Many computer RPGs are single-player experiences. In these cases, if there is a group of characters played in the game, the same player controls them all. Or if multiple players work together (as in massively multiplayer online RPGs such as *World of Warcraft*), the game is structured for players to communicate over a network rather than face-to-face.4 Crucially, no player has the part of the game master. The execution of the rules, presentation of the fictional world, portrayal of the NPCs, and shaping of the story is left to computational processes and data.

When brought to the computer, then, the core experience of story and character in RPGs shifts. In tabletop and live-action RPGs, story and character are experienced primarily through collaborative human performance. But for computer RPGs, especially single-player ones, story and character become media experiences—made interactive via computational processes. In particular, two operational logics have come to prominence in the story and NPC presentations of computer RPGs. These are *quest flags* and *dialogue trees.*
Quest Flags
Modern computer RPGs use many of the same technologies and techniques for representing their fictional worlds as other games. A character (or group) moving through an explorable three-dimensional world—with objects to pick up, NPCs to engage, and combat to resolve—is present in first-person shooters (e.g., the Doom series), platformers (e.g., the Prince of Persia series), and RPGs (e.g., the Knights of the Old Republic series). What changes is the emphasis in play. As the name would suggest, first-person shooters are focused on combat as the main form of play. It becomes both the primary challenge and main motivation for players moving through the world. Platformers, on the other hand, make movement itself (sometimes through a puzzling space of platforms of varying shapes and heights) the main challenge and motivation for players. RPGs, however, often work to motivate players to engage in a variety of types of play (e.g., exploration, combat, and intellectual puzzle solving) via character development set in a larger story. In particular, many RPGs give the sense that the story itself is playable by offering the player freedom to roam across a world infused with quests that operate at many scales, can sometimes be completed in different ways, and are often optional or available for partial completion. As each player chooses which quests to accept—as well as how, whether, and when to complete them—this creates a different story structure for each playing. Some of the player’s nonquest activities may be directly related to this structure (e.g., taking on tasks in the world that earn money, in order to acquire enough to purchase an item needed for a quest), but the structure also provides one context in which even
Expressive Processing

5. I will not attempt to define quests here or present an exhaustive discussion of the types of possible quests. A good overview and proposed definition are offered by Susana Tosca (2003); Jeff Howard’s *Quests* (2008) makes a detailed …

Notes continued at end of this chapter.

world exploration for its own sake can be situated.5

Despite the variety of experiences that players can have with quests, it is commonly observed by both players and authors that there are a limited number of types of quests—perhaps three, or even one. As Lee Sheldon points out in *Character Development and Storytelling for Games*, this observation is correct, as far as it goes, given that “the mechanism for tracking progress by the game engine is virtually identical in every quest” (2004, 227). This mechanism is the setting and checking of a collection of small pieces of data—often called “tokens” or “flags,” but sometimes known by the more formal term “variables”—as the player progresses through the world. These data flags represent the state of the world as it relates to quests of varying scope. They are generally checked and set by relatively simple “scripts” that can be edited by game designers and writers (without recompiling the entire code of the game). The state of quest flags is often explicitly presented to players in the form of a personal “journal” or “notebook” that scripts update with helpful reminder texts about the current state of each quest (at the same time that flags are updated). Other quests are, while frequently signaled equally directly, instead organized around the possession and use of game world objects without messages in the journal.

Overall, this approach is notable for its simplicity of structure. In fact, it is so simple that game companies have worked to open its authoring to players who are not software developers. For example, when game developer BioWare published the game *Neverwinter Nights* (Oster, Holmes, Greig, et al. 2002), it also released the Aurora toolset for creating new spaces, objects, NPCs, and quests.
The goal, in part, was to offer players a rare opportunity in the world of computer RPGs—to play a game master, in a manner somewhat similar to tabletop RPGs, crafting the game experience of other players. As in many other games, the quests of *Neverwinter Nights* are presented to the player in the form of a virtual journal.

Using Aurora, the first step in creating a quest is to begin a new category for entries in player journals, using the special purpose Journal Editor tool (shown in figure 3.1). Each entry in a category is designed to let the player know that the appropriate scripts have been activated to set the flags that are necessary for the next stage of the quest. For example, one could create a journal category with three entries:

- **[0001]** Grandfather had a signet ring that belonged to his own great-grandfather. But apparently no one has seen it since it was passed down to him—and it wasn’t mentioned in his will.
- **[0002]** Grandma would treasure this ring with the rest of Grandpa’s jewelry. But the carved symbol you’ve seen before at your Uncle’s, so he might be interested in it as more than a memento. Of course, it also looks valuable, and you could use the money you’d get from selling it.
- **[0003]** You no longer have the ring.

Different events can trigger scripts that change the current state of the player journal. For instance, the first entry above could be added based on a conversation with the player character’s grandmother, or finding and reading a letter sent to the player character’s father. The same script could set a value of “true” for a flag with a name such as “lookingForRing.” Dialogue trees for NPCs could be different based on this flag, offering the player the option...
Expressive Processing

of asking people if they know anything about the ring, leading to clues as to its location. Once the ring is found, a script fired by that event could move the journal to the next entry as well as set lookingForRing to its earlier “false” value, while a flag such as “learningAboutRing” could be set to “true.”

If the game authors wish to move all players through the quest in the same way, the ring could be in a locked attic room that isn’t accessible until lookingForRing is true. Alternately, the player could have the option of finding the ring before hearing about it in any other way. Quest designs that follow this latter sort of approach (forcing players through as few “gates” as possible) certainly give the player more options for how stories unfold, making the story have some of the same feel of free exploration as the graphical world.

But seeking this feeling of exploration carries a price—one that attention to the shape of processes can lessen. For example, in the case of our hypothetical family ring
quest, starting by finding the ring (jumping straight to the second stage) seems unlikely to produce any problem with the flags or journal. Setting lookingForRing to false when it is already false makes no difference, and starting with the second stage of the quest works with the journal entries as written. But, to determine such things, the game author has to carefully think through every possibility—and the more freedom given to the player (as to when, how, and whether to take on and complete quests), the more the events of different quest strands need to be considered in relation to one another.

This, in turn, brings us back to the idea of simplicity. The quest flag structure is a simple one in a variety of ways. It’s easy for software developers to implement, it doesn’t require much in the way of computational resources (leaving both programming and processing resources free for areas of the game such as graphics, which is something most game companies value), and it can be exposed to the audience quite straightforwardly. Yet given the goal of the audience feeling able to play within the fictional world—not just in areas such as combat but also in terms of the stories being told—the quest flag structure quickly brings game authors to a limit point of complexity, holding back the future development of the form. I’ll consider this further in my discussion below of BioWare’s widely lauded single-player RPG, **KotOR**. First, however, a brief discussion of dialogue trees is in order.

**Dialogue Trees**

Tabletop and live-action RPGs create dialogue between characters by having two human beings improvise it. No computer process can substitute for the immediacy
Expressive Processing

8. As Dominic Arsenault pointed out during the blog-based peer review of this book, dialogue trees can provide a variety of interaction experiences for similar underlying structures. Games such as KotOR present the player with a selection of possible …

Notes continued at end of this chapter.

of such an exchange (or even come close to an ability to understand free-form human language) so computer RPGs have been forced to take a different approach. As with the quest flag approach to story, the computer RPG approach to dialogue is simple both in terms of software implementation and audience comprehension. This approach is the dialogue tree.

Like the *Eliza* system, the dialogue tree model of conversation is founded on turn-taking. But what happens at each turn is utterly different. During the audience’s turn, dialogue trees do not allow for the free-form textual input of *Eliza*. Instead, when interacting with a dialogue tree, the player is generally presented with a small number of options (usually ranging from one to five) for what the player’s character can say. After a selection is made, the system performs the response of the NPC(s) to which the player’s character is speaking. Conversation continues in this manner until the player selects a conversation-ending option (which may or may not be explicitly presented as such).³

Rather than NPC responses being generated by applying rules to the audience’s last statement, as with *Eliza*, each NPC response is selected by traversing a hierarchically organized data structure. This is what gives the dialogue tree its name.

Figure 3.2, an image of Aurora’s Conversation Editor tool for authoring dialogue trees, makes this more specific. It looks quite similar at first glance to the tool for quest journal entries. One difference, however, is that while the journal contained quest stages in an order likely to be encountered by players, the ordering of the text in the conversation editor seems rather odd:
Computer Game Fictions

- Yes child?
- I found grandfather’s ring.
- I’ve been wondering about some things.
- Is there any food in the fridge?
- <FullName>, you’re looking well. And dressed smart. What’s that on your hand?
- The scar?
  - No, the ring.
    - [link to “This ring?”]
- This ring?
  - Oh, I—I almost thought it was your grandfather’s. Haven’t seen it since we were young.
    - I miss him.
      - We all do.
        - [link to “You still think of his ring?”]
  - Excuse me. I have to go to the bathroom.
[End conversation]
- You still think of his ring?
  - I remember because he was so secret with it. There are times I wonder if he got killed for it, and that’s why it wasn’t with his things. If you found it, that would put me at ease. Would you look?
    - Of course.
      - Thank you so much.
  - Soon, I promise. But not right away.
    - I understand. Come back a better time.
  - One of my cousins would be better.
    - Jerry’s a cop.
      - You’re probably right.

This odd ordering is in part a reflection of the “tree” nature of dialogue trees. Each of the two groupings above starts with an initial statement by the grandmother NPC,
9. The "[END DIALOGUE]" at this point in the image is not correct, but an artifact of Aurora. The same is true after "You still think of his ring?"

with possible replies nested at the next level of indentation, each of which leads to a particular NPC reply (and/or the end of the conversation), which may itself have further player character statements nested at a further indentation level below. For this reason, dialogue tree segments can't be read straight through. Dialogue trees are also rarely pure tree structures. Instead, many branches make connections to other parts of the tree—as above, where both of the possible responses to the grandmother’s first question (“What’s that on your hand?”) lead to the same place, because of a link from “No, the ring” to “This ring?” 9

Given this, understanding how players will experience a segment of a dialogue tree requires both traversing the
Computer Game Fictions

hierarchical tree structure and following the links that connect the structure’s branches.

Another type of odd ordering, that of the two primary tree segments, emerges from common authoring practices. Dialogue trees check and set the same types of flags employed by quests—and different sets of dialogue are appropriate at different moments. The expanded segment of dialogue above (that begins “<FullName>, you’re looking well”) is appropriate for the first time that the player character meets the grandmother, which only happens once, so it is placed at the bottom. The most commonly encountered dialogue is placed at the top, as the default, and only altered (or skipped) if particular flags have certain values. If the grandmother were to have only two possible segments in her dialogue tree, this could be accomplished simply by having a flag called, for example, “firstTimeTalked” that is initially set to true. The default dialogue segment would be skipped whenever firstTimeTalked is true, and entry into the second dialogue segment could trigger a script to set firstTimeTalked to false.10

Chris Bateman, in Game Writing: Narrative Skills for Videogames, is generally negative about dialogue trees. He writes:

Despite the name, dialogue trees are seldom true trees but rather converging and diverging chains of conversation. They can be a nightmare to work with, and the benefits they provide are somewhat minimal. Nonetheless, some players greatly appreciate the illusion that they have control over what their character can say, with the consequence that dialogue trees remain important, especially in cRPG games. (2006, 277)
Expressive Processing

My view of dialogue trees is rather different. First, one remarkable thing about dialogue trees is how little illusion they present, especially when compared with systems like *Eliza*. Each NPC will only respond to a limited number of things—and these are plainly presented. Further, because dialogue tree traversal options are “converging and diverging” (often, for default dialogue segments, in a cyclic pattern so that the final options at the ends of branches include a return to the trunk), the fact that dialogues are structured as tree traversals becomes clear even to first-time players.

While this is certainly a (poor) “illusion” of real conversation, it isn’t an illusion of control over what the player character says, as Bateman suggests. Rather, in most RPGs it is an important method of making gameplay decisions—with different dialogue options altering game variables in different ways. The range of possible results from dialogue tree choices is vast, including, in popular
RPGs, either accepting or declining quests (and rewards for those quests), defusing tense situations or initiating battles, learning more about NPCs (and perhaps unlocking quests related to their personal lives), and changing how NPCs feel about the player character in more emotional ways (ranging from feelings of loyalty and honesty to romantic subplots).11

At a technical level, it is correct to say that all these things boil down to menu selections. But successfully traversing these menus can be a tricky business—one made easier for players who draw on their knowledge of the game’s fictional world. In addition, it is one of the most direct means by which players can express the role they imagine for their character(s): agreeing or refusing to do things for others, acting out of kindness or mercenary motivation, telling the truth or spreading falsehood, and so on. Building such consequential choices into dialogue trees is also quite simple for authors. In figure 3.2, the lower portion of the window shows how a few menu selections are sufficient to cause a line of dialogue (“Thank you so much”) to also update the player’s journal—in this case, moving the quest in the Grandpa’s Ring category to entry number one. Figure 3.3 shows this taking place in the game.

But, despite the ease of this basic piece of authoring, Bateman is certainly correct that dialogue trees can be a nightmare to work with—as can the larger method of storytelling that they form in combination with quest flags. Further, this might be one of their smaller problems, as I discuss below in the context of a particular example. First, though, a broader look at the logics of these operations is in order.
Expressive Processing

12. And of course Eliza also makes an occasional fallback on precreated transformations when the most recent audience statement contains no keywords.

**Quest and Dialogue Logics**

The primary operational logic at work in Eliza is transformation—each statement from the audience is immediately transformed into Eliza’s next statement, using a set of processes in a manner specified by the data of the current script. In computer systems parlance, we might say that Eliza conversations are nearly “stateless.” Eliza doesn’t model the ongoing state of the conversation, using its script differently depending on prior interactions, except to avoid excessive repetition of the use of certain rules.12

The combination of quest flags and dialogue trees, on the other hand, is in some ways all about state. The quest flag logic is precisely one of milestone-based progression. Quest flag fictions are ordered sets of discrete units. What matters is where the player is located—at which milestone along the path—and this is precisely what is exposed through the mechanism of the player journal. In some cases the beginning of the progression may be skipped and the end may never be reached, but at each moment of play the fiction is at a particular point, among a small number of predefined points arranged in order.

The logic of the dialogue tree, in contrast, is essentially that of the directed graph. Rather than modeling conversation as a set of discrete exchanges with no context (as in Eliza), the dialogue tree always locates the current conversational state at one particular point, among a set of predetermined points, from which navigation is possible to other points via predetermined links. As with the milestones of quest flags, it’s usually impossible to go backward—the graph is directed toward “progress” in the conversation—but it is also usually possible to loop back
to the main trunk of the currently available conversation, if occasionally rather circuitously.

In other words, somewhat like the graphical logics of games, the logics of dialogue trees and quest flags are about location in a given space. But while the visual spaces of games are often simulated in a manner that supports almost innumerable possible locations, the milestones of quest flags and graphs of dialogue trees mark out all the possible positions (and transitions between them) ahead of time. This mismatch proves problematic.

**An Example: Star Wars: Knights of the Old Republic**

The Game Developers Choice Awards are the Oscars of the game industry—the award with which members of a creative industry recognize achievements of their own. In 2004, game studio BioWare walked away with three awards that are of particular note for this discussion: Game of the Year, Original Game Character of the Year, and Excellence in Writing. All of these were awarded for BioWare’s RPG *Star Wars: Knights of the Old Republic* (KotOR), which also won game of the year awards from a slew of industry publications. While certainly not the most recent major RPG, it provides a good example of the strengths and weaknesses of the quest flag and dialogue tree logics.

**KotOR’s Successes**

*KotOR* uses quest flags and dialogue trees to reward and sustain engagement with its fictional world; establish patterns that, when altered, produce small moments of surprise and pleasure; and direct the audience’s attention to a series of things that must be accomplished through
Expressive Processing

13. This is generally done quite artfully. Even an exceptionally awkward-seeming opening exposition is later given deeper resonances as players progress through the main quest. It is revealed that the characters surrounding the player …

Against this epic backdrop (with its many components, some rather mundane) other, optional quests are presented. Some involve developing the relationships between the main player character and her or his traveling companions. Others involve taking a role in the events happening on particular planets, often with a decidedly interpersonal element (e.g., a wife begs you to find evidence to clear her husband of murder charges, which soon leads to revelations of his infidelity, which you must choose whether to discuss with them). Some are quite small, such as discovering your ship has smuggled cargo aboard, being given the code for the container, and being offered good payment for the goods by a crime syndicate representative. In cases such as this last one, the player need only decide what the characters will do with the one-stage quest—and then enact the decision in the game’s fictional world (travel to the places, interact with the objects, and talk with the appropriate NPCs).
In addition to all these activities, there are also others embedded in the fictional world—ranging from the pleasures of exploration and spatial mastery to “minigames” of card playing and racing. Some of the minigames, in turn, can unlock additional quests and NPC interactions as well as provide resources useful in completing parts of the game to which they are less directly connected.

Given all this, conventional wisdom has it that playing a major RPG takes experienced gamers something like forty hours. But fans of the genre often spend longer, indulging in many optional activities. In addition, many play through the same game multiple times. This is particularly true for games, like _Kotor_, that provide different gameplay options to characters that approach the world with different ethical stances.

What drives players to spend forty, eighty, or more hours with a computer RPG? If we look only at the quest journal, where the operations of quest flag logics are exposed to the player, the appearance is of a massive indulgence in the pleasures of the to-do list. And certainly there is some of that. But, more centrally, _Kotor_ is constantly providing doses of narrative closure and transition—from the small satisfactions of quest stages to the holistic sense of a planet’s stories, characters, geography, and history that can be developed over one or many playings. Jill Walker Rettberg talks about the pleasure found in learning the quest-based fictions of a place, the “network of fragments, most of which are not necessary to experience the game fully, and yet which cumulate into a rich experience of a storied world” (2007, 310).

At the same time, beyond its narrative pleasures, _Kotor_ is also continually providing other reasons to keep going,
to move one part of things along just one more step. A powerful element of this can be found in the rewards given with each bit of quest closure. Some rewards are as simple as cash that can be used to purchase items in the game, bribe recalcitrant NPCs, help those in need, and so on. When brought to fruition, many story elements also deliver experience points (XP), another convention borrowed from tabletop RPGs. As characters accumulate experience, they increase in “level” and become more capable in the game world. Each new level is achieved at a particular number of XP, and the drive to hit the next number is another motivator for the “just one more thing” mind-set that can keep players at KotOR for hours after they’d planned to stop for the night. It is this quantified progression of a primary player character that has led William Huber (2009) to call the RPG genre a “statistical bildungsroman.”

And at forty, eighty, or more hours, the extent of audience engagement with KotOR is certainly more akin to a thick German novel of personal development than, for instance, a film (or even a season of television). But, again, a better analogy is probably with a tabletop RPG campaign—into which players can easily invest a similar amount of time. As in many RPG campaigns, KotOR works to create a sense of flexible story making couched in world exploration and character development. To this end, the player’s character can visit planets in different orders and multiple times; quest items can be found at different points in the quests to which they’re connected; and necessary items and information are often available in multiple ways. Nevertheless, while these things are also true of tabletop RPG campaigns, in KotOR and other computer RPGs
they must be managed through quest flags and dialogue
trees (rather than human memory, improvisation, and
creativity). As mentioned earlier, this creates difficulties for
game authors.

KotOR's Troubles
Much of the narrative power of KotOR comes from the
ways that it makes playable structures out of tried-and-true
narratives. A self-contained example of this can be seen in
one of the optional quests on the planet of Dantooine,
relatively early in the game: the feud between the Sandrale
and Matale families. I encountered this quest while
playing the “Platinum Hits” edition of the Xbox version
of KotOR—and it is worth noting that the experience of
those playing other versions might be different.

After being warned that tensions between the two
prominent families were in danger of overflowing into a
violent civil conflict, my party traveled south to the Sandral
estate and spoke with the patriarch, Nurik Sandral. He told
us that he felt great sorrow over the disappearance of the
young Matale heir, Shen, but knew nothing about it. He
told us that his own son, Casus, had been missing for some
time—and speculated the two might have met similar fates
amid the dangers of Dantooine.

Nurik asked us to show ourselves out. But shortly
after this his daughter, Rahasia, appeared. After I selected
some friendly things for my player character to say to her,
she revealed that her father had in fact kidnapped Shen
Matale. In typically Shakespearean fashion, she and Shen
had fallen in love. She gave my party a key to a side entry
of the Sandral estate, making it possible for us to sneak
in and rescue Shen. Once we reached Shen he refused to
leave without Rahasia—and the result was Shen, Rahasia, and the three members of my KdoR party all coming out the side entrance at once, where we found ourselves confronted by the patriarchs of both families and their battle droids (figure 3.4).

After some tense dialogue-tree discussions (in which I chose statements supportive of the lovers and designed to defuse the conflict), Shen and Rahasia ran off to live in the safety of the Jedi enclave, while their fathers just barely held back from igniting a conflagration. Later, while exploring a portion of the planet further north, my party came on the Matale family compound. The guard droid granted us an audience with the patriarch, Ahlan Matale, who we had last seen as his son ran off to the enclave with Rahasia. Ahlan proceeded to demand at length that something be done to find his son—outlining his suspicions that the Sandral family had kidnapped Shen.\textsuperscript{15} He offered us a reward (which sounded more like a bribe) should his son be found. I suspected this inappropriate dialogue tree segment might be active because of a simple flag of the firstTimeTalked variety, so I took my party away and then returned to the compound. But flag structure was apparently organized in a different way. When we returned, Ahlan Matale came out again to demand an investigation into the possible kidnapping of the son he had already seen rescued from kidnapping. This illustrates one type of problem with the quest flag and dialogue tree approach, a type that results in inappropriate events.

The other major type of problem was illustrated shortly, when my party discovered the body of Casus Sandral. An amateur archaeologist, Casus had apparently been killed by wild animals while undertaking a dig in a
dangerous area. We immediately went to the Sandral estate with Casus’s diary, in order to share it with his worried family. But the estate was shut down entirely, without even the droid out front who had greeted us on the first visit. This second type of problem is visible when the game seemingly arbitrarily shuts off quest possibilities that have the force of narrative drive behind them.

Both types of problems emerge, most commonly, at the juncture between the freely explorable fictional world and the rigid structures of quest flags and dialogue trees. I encountered these problems regularly in my playing of KotOR. Just as the game expected me to visit the Matale estate before the Sandral estate, and produced inappropriate events when I visited in a different order, the same was true on a planetary scale. For example, I visited the home planet of one character who joined my party, Jolee Bindo, later in the game than KotOR’s dialogue tree structure expected. As a result, much of the conversation between him and the main player character consisted of his darkly hinting at truths that had already been revealed in a dramatic fashion. Each conversation with Bindo undermined the sense of KotOR having a consistent fictional world. At the same time, the conversation path with another key character, Carth Onasi, was shut off entirely after not being pursued in the expected manner—despite the fact that there was clearly much to discuss.

These problems do not spring from poor work at BioWare. It would be unreasonable, for instance, to expect lead writer Drew Karpyshyn and his team to have written (and sent for voice acting) different versions of Onasi’s lines for each possible state of the story. As Chris Crawford has observed, such approaches “are always too
Expressive Processing

Figure 3.4a. Selecting what to say during the confrontation outside the Sandral estate.

Figure 3.4b. After defusing the potential battle, a journal entry is added, experience points are gained, light side points are gained, and items are lost.
Figure 3.4c. The updated journal summarizes the quest and its outcome, including the family confrontation and the lovers’ move to the Jedi enclave.

Figure 3.4d. The unexpected conversation that followed.
Expressive Processing

16. Another way of avoiding such problems was employed in the next game on which Karpynshyn was the lead writer: Mass Effect (Hudson et al. 2007). The long-term development of personal relationships with crew members …

Notes continued at end of this chapter.

much work for the designer and not enough meat for the player” (2004, 126). And it is unlikely that the tools at their disposal offered any more elegant (and tractable) method of addressing the situation. Once the sequence of expected timings between quest flags and dialogue sections was broken it might have made sense to allow the player character to ask personal questions and be brushed off, but this is essentially the same as shutting down conversation, as the KatOR team did.16

Which is to say that problems such as these spring from a poor fit between the simple, brittle structures of quest flags and dialogue trees, and the ambitious, flexible game design they are being used to support. Those of us familiar with the logic of milestone-based progression—from business plans, grant proposals, employee evaluations, and so on—know that any detailed set of milestones will generally meet one of two possible fates. First, as the situation evolves, it can become clear that the milestones will be revised for one or more reasons: the steps may not be what one originally thought, may not happen in the order originally thought, or might need to be divided up differently. Second, an alternative fate is that the milestones themselves become a fetish, irrationally driving behavior in a situation that they clearly no longer reflect. Even in a world simulated as partially as that of KatOR, precreated milestones still become an uneasy fit with the evolving situation. Unfortunately, only the second of these two fates is possible for KatOR’s milestones, which cannot be revised by the system.

Given this situation, the prospects for fiction in games may seem grim (perhaps particularly for multi-player online games, see sidebar: Individual Fictions in Shared Worlds).
But there is a widely practiced alternative approach to the combination of game and fiction—one that is more successful in important respects.

**An Alternative:**

**Prince of Persia: The Sands of Time**

At the 2004 Game Developers Choice Awards, *Kat*OR had some competition. Another nominee for Game of the Year, a game that won the awards for Excellence in Game Design and Excellence in Programming, was *Prince of Persia: The Sands of Time* (PoP).

PoP is a strong example of game fiction in its own right—which is no surprise, given that its main writer and designer was Jordan Mechner, a legend in the game design field for his pioneering games *Karateka* (1984), the original *Prince of Persia* (1989), and *The Last Express* (1997). Mechner is credited with bringing cinematic storytelling to computer games, pushing forward techniques from the realistic modeling of human motion to the integration of an overall story into an action game structure. Mechner is also an accomplished independent filmmaker—writing, directing, and editing films such as *Chavez Ravine: A Los Angeles Story* (2005), a documentary about a 1950s community evicted from land they were told would become a public housing project (where they would return to live) only to see Dodger Stadium constructed instead, through a process of greed, hypocrisy, and anticommunist hysteria.¹⁷

Bearing this in mind, it should be no surprise that PoP is one of the strongest examples of an entirely different approach to game fiction from that of *Kat*OR.¹⁸ Rather than attempt to make the game’s story a playable
Expressive Processing

**Individual Fictions in Shared Worlds**

The mismatch between a simulated, explorabale spatial world and fictional progression implemented in inflexible milestones can become even more pronounced in massively multiplayer online games. The typical design for such games requires that objects be present in the world for all players. This means they can be encountered by players who have not yet received quests to give them context—and they must reappear to be available for other players even if they have been removed by one particular player.

I experienced these things from the outset of playing *EverQuest II* (Blakely et al. 2004). I began in the Queen's Colony, the new player area for the city of Qeynos. I received my initial quest and then went exploring, encountering monsters, quest givers, and other players. Often I would kill a number of some type of monster (such as a Morak Devourer) in a particular area and then, a few minutes later, be given a quest to do exactly what I had just done, with no option to reply that I’d already done it.

This helped prepare me for my experience in the Abandoned Village area. There I fought some monsters and found a strange statue. I clicked on the statue, but it wouldn’t respond. In a corner of the area I met Sorcerer Oofala, but he had nothing to say to me. Only later, after my character died, did I end up back in the starting area and talk again to the initial quest giver, Murrar Shar. He sent me to Cleric Mara’Vaen, who also had not been interested in talking when I encountered her earlier. But this time she had a quest, I completed it, and then she sent me to find Oofala (again), who sent me to destroy the strange statues I hadn’t been able to get a response from earlier. At this point I could click to destroy them, if not interrupted by combat during the work.

After I destroyed the last of the statues, an enemy called the Dark Blademaster appeared. I defeated him, took his shield, and spent a few moments shifting things around in my inventory so that I was using the shield. As I did this, the Dark Blademaster reappeared! Already wounded, I decided to run back to Oofala, whereupon the Blademaster stopped following. He had reappeared, of course, so that other players on the same server could also complete the quest that Oofala then gave me: to defeat the Blademaster. The only way to progress was to go back and fight the Blademaster again, this time wearing the shield I’d acquired from the time before last I had fought him.
one (a story that changes shape depending on how play proceeds), PoP has a linear, semcinematic story, with a sense of inevitability artfully fused with a context for player struggle. The struggle, specifically, consists of innovative platforming action and acrobatic combat. For games such as PoP, the question is how to make this kind of combination, between audience-controlled gameplay and linear story, most productive.

The typical structure for such combinations, in the computer game industry, is simple alternation. The audience is given moments and spaces of play that, when completed, yield scripted sequences that tell the story.19 Because this model involves so little connection between story and gameplay, game writers are often employed only at the end of game production. The game’s play mechanics, spaces, and even characters are designed before the writer arrives (Bateman, Boon, Buckley, et al. 2003). The writer’s task is to create a set of scripted sequences that will tie them together and provide an overall context. The results are frequently unsatisfying and can feel, for good reason, quite arbitrary. Mechner’s games—like those of other designer/writers such as Tim Schafer (the creator of Grim Fandango [1998], Psychonauts [2005], and other well-regarded titles)—instead work to conceive the story and gameplay as integrated entities.

The story of PoP begins with a war of aggression fought for reasons of greed. The war’s booty includes a massive hourglass containing the Sands of Time. The possession of these sands, however, turns out to be more curse than blessing.20 When the briefly victorious Persian king seeks to enrich his friend with a present of the Sands, the result instead is to destroy the friend’s palace and turn nearly
Expressive Processing

everyone into sand demons. The prince’s dagger, looted from the same palace as the hourglass, is the key used to unleash the sands. Both of the game’s main NPCs—a woman, Farah, taken prisoner in the looted palace, and the vizier who betrayed the palace’s owner and incited the unleashing of the sands—suggest that the dagger can also be used to improve things. But the vizier has removed the hourglass to the highest tower of the ruined palace.

Mechner describes this as the result of simplifying the game’s story as much as possible. In terms of traditional story, it is certainly efficient, with only three characters: a hero, a sidekick/love interest, and a villain. It is also efficient in terms of gameplay. For platforming, it provides a challenge (moving through a ruined palace) and a motivation (reaching the hourglass). For interacting with most NPCs, it supplies a motivation for combat and an explanation of why other modes of interaction are not available (they’re sand demons). In other words, it focuses attention on what games do well (movement in space and combat) and offers an explanation for avoiding what they don’t do well (interactions with characters) as well as a reason for moving forward. This is already better than most games, but PoP goes further in a number of ways.

Centrally, it creates connections between gameplay and story. The most obvious of these is the dagger. It is both a key item in the story and the focus of what is innovative about the gameplay. The dagger can store the Sands of Time, making it possible for the hero to use powers that make movement and combat easier (e.g., turning back time when mistakes are made). The dagger only holds a limited amount of sand, which is replenished by withdrawing it from vanquished sand demons (a requirement lest they
rise again). Story and gameplay also connect through plot twists. PoP's story employs twists—including one in which the dagger is lost—that also alter the experience of gameplay. Finally, as Mechner (2007) points out in his essay “The Sands of Time: Crafting a Video Game Story,” PoP also works to give as many as possible of the key story moments to the player, in gameplay—as when the prince must face his own father, now a sand demon.

PoP also makes choices at the level of process that are significantly different from games like KotOR. Obviously, given that there is only one, linear story strand, there is no complicated system of quest flags. There are also no dialogue trees. PoP instead uses what is sometimes called an “event-based” dialogue system. In essence, conversation is removed from the playable elements of the game. Instead of players choosing to talk with particular NPCs, certain events trigger conversation and voice-over. In most cases these are not presented as interruptions to gameplay—as happens in many games—but rather layered over it. The prince may think to himself as the player causes him to run down a hallway or swing from a pole. Farah may make comments about the prince’s progress (or failures) in moving through a particularly difficult area, as determined by the player’s actions. Or, in one case, the player may directly elicit a response from Farah—by moving the prince into a first-person camera mode and “staring” at her.

PoP also shares one primary tool for NPC interaction with games like KotOR as well as nearly every type of game that includes NPCs: the finite-state machine (FSM). While on an abstract level all computers could be described as FSMs, in computer game NPC logic FSMs tend to be used a particular way. Each of an NPC’s basic behaviors
is defined as a “state” (e.g., patrol, attack, or retreat) with particular rules for transitioning from the current state to other states. For example, an NPC could be designed to move through a patrol animation (a patrol state) until an enemy is noticed (transition to an attack state) and continue attacking until the enemy is defeated (transition to patrol state) or the NPC’s health is below a defined threshold (transition to a retreat state). There might be no rule allowing for direct transition from patrol to retreat—which would make sense generally, but not if the NPC is already badly wounded from previous combat. As with quest flags and dialogue trees, FSMs are simple to explain, easy to implement in software, low in their use of computational resources, and an authoring nightmare over a certain level of complexity. Luckily, they’re a perfect fit for the exceedingly simple behavior of sand demons, who attack on sight and never retreat, as found in PoP.

The end result, for PoP, is an experience elegantly designed to compensate for the crippling limitations of the processes used to represent story and character in today’s computer games. And this opens up the next question.

### The Game Fiction Dilemma

Authors of game fictions have worked hard—through conventions such as the quest-tracking journal and tree-driven conversations presented as menus—to avoid the *Eliza* effect. Rather than conceal the operations of their processes, game fiction authors seek to expose them to the audience. But, despite this, game fictions still face a dilemma remarkably similar to that outlined at the end of the previous chapter.\(^{21}\)

Both *KatOR* and *PoP* take advantage of what games do
well—in particular, simulated movement through space and combat. The relatively free-form actions allowed to players in these areas might be seen in parallel with the free-form text composition allowed to both those interacting with *Eliza*/*Doctor* and the students involved in Garfinkel’s *yes/no* therapy experiment. The difference, again, is in what changes to the state of the system and influence on future operations can be produced by this interaction.

*PoP*’s fiction, like Garfinkel’s experiment, has an extremely narrow range of possible responses to interaction. Either the player’s actions successfully move the fiction to the next stage (a progression signaled to the player by the triggering of a scripted sequence) or they don’t. The story system is, as players put it, “on rails”—and its structure can be completely exposed to the audience by letting them know when they are departing for the next metaphoric station. Meanwhile, *PoP*’s NPCs are mostly only available for combat (the sand demons). The major exception (Farah) will occasionally offer a linguistic interjection in response to nonlinguist actions in the world, but this is another narrow interaction conduit.

*KaOR*’s fiction, on the other hand, while not allowing the free-form textual input of *Eliza*/*Doctor*, does accept many more actions in the world as input into its system of quest flags and dialogue trees (e.g., whether, when, and how to take on quests, take quest actions, speak with NPCs, move between worlds, etc.). Further, as this chapter has shown, such a system can have many more elements to it than the railroad system of games such as *PoP* (authors can produce huge amounts of data for the quest flag and dialogue tree processes, different subsets and orderings of which can appear with each playing). The result, as with
an *Eliza* conversation, is that the number of potential outcomes is huge. Unfortunately, there is a mismatch between the great variety of situations in which *KotOR* is expected to perform a fiction and the simple model of fiction and character embodied in *KotOR*'s processes—just as there is a massive mismatch between the complexity of human language to which *Eliza* must respond and its extremely simple model of conversation (as a series of transformations). The result, in both cases, is a tendency toward breakdown that takes a shape determined by the underlying processes. And, as with *Eliza*, the processes of *KotOR* are of a basically uninteresting shape.

To put it succinctly, the practices of the mainstream game industry present authors of digital fictions with two bad options for going forward. One is to “design around” breakdown, as *PoP* does, and essentially forfeit the processing power of digital media at the level of the fiction. The other option is to attempt to layer a semiflexible story—organized as a set of ordered milestone progressions—over a much more flexible game world. This creates a space of play that, if embraced by players, leads to unsatisfying breakdown.

In short, the time is ripe for a new approach to game fiction. But just as *Doom* could not be built on the approach to graphics in *Myst*, this will require an approach to fiction and character that is fundamentally different, that is more expressive and flexible than quest flags and dialogue trees. Luckily, this work does not need to begin from scratch. It can start instead by building on a history that has been present as a strand of practice within the artificial intelligence community since the time of *Eliza*. The coming chapters explore this history.
Notes

1. Some games, of course, can hardly be said to engage fiction at all. No one wants a story with their Tetris—and its “fictional world” of falling blocks barely supports the phrase. Given the sometimes-contentious nature of critical discussion around the relationship between games and fiction, I should perhaps also make it clear that I do not believe any games “are” stories or narratives in a classic narratological sense. Rather, following Espen Aarseth’s call for discussion of “quest games” (2004), I propose here to give careful attention to the specific operational logics of quests and dialogue trees in computer RPGs. For readers interested in a more detailed discussion of games and narratology, I suggest chapter 8 of Avatars of Story by Marie-Laure Ryan (2006).

2. As tabletop RPGs have come to encompass much broader areas of life than combat, different games (and player groups) have diverged in their treatment of these elements. One movement is toward a generalization of the statistical models used for war game combat. They have been employed for many additional elements of character progression and play. The other movement is toward an emphasis on creative expression that may resist quantification, or at least be decided by the logics of character and fiction rather than chance, leading to “diceless” RPGs. Players who prefer each emphasis are sometimes called “roll players” and “role players,” respectively.

3. The most famously Lovecraft-inspired game is Call of Cthulhu (Petersen 1981), while Over the Edge (Tweet 1992) is a game inspired by both Burroughs and Philip K. Dick. At the same time, as Jose Zagal reminded me in the blog-based peer review of this book, some RPG systems—such as GURPS (Jackson 1986)—are designed to be “generic,” rather than rooted in a specific fictional world or even a specific type of fictional world. Finally, the games of Costikyan’s that I’m referencing are Toon (1984) and Bestial Acts (1993), respectively. Together with the innovative RPGs The Extraordinary Adventures of Baron Munchausen by James Wallis (1998) and Puppetland by John Tynes (1999), Bestial Acts is reprinted in Second Person (2007), a book I edited with Pat Harrigan.

4. Though many massively multiplayer online play groups, including some in which I have participated, include face-to-face interaction between some players. And there are a variety of other hybrid forms that combine elements of in-person and computer RPG play. For example, during the online peer review of this book, Sarah Toton drew attention to the web sites Obsidian Portal and Gleemax (<http://www.obsidianportal.com> and <http://www.gleemax.com>). The first of these is an independently produced content management system for RPG players that focuses specifically on “fictional” elements, providing a blog for chronicling adventures and a wiki (divided into areas for the party and game master) for keeping track of the campaign world. Gleemax, on the other hand, is a site developed by publisher WotC for online play, social networking, and general reading about games. Toton’s comment reminded me that we should also consider hybrids such as the Living Greyhawk Dungeons & Dragons campaign of the WotC-supported RPG. As the web site explains:

The Living Greyhawk campaign is an immense game played out in regional events throughout the real world. The continent of the Flanaess in the game world is divided into several nations and political states. These nations are mapped onto sections of the real world. Your real world location determines the default home region for you and your characters in the game world. When you travel to different areas of the real world, your character journeys with you to the corresponding locales in the Flanaess. Each region has a special flavor setting it apart from other regions, allowing you to immerse yourself in the intrigues of your home region or join the turmoil in other regions. (Wizards of the Coast 2005)
5. I will not attempt to define quests here or present an exhaustive discussion of the types of possible quests. A good overview and proposed definition are offered by Susana Tosca (2003). Jeff Howard’s Quests (2008) makes a detailed engagement with relevant literary history and theory—and also offers design exercises using the Aurora toolkit discussed in this section.

6. As noted in this chapter’s first sidebar, Neverwinter Nights is a licensed computer version of the tabletop game Dungeons & Dragons—which remains the most popular tabletop RPG, in addition to being the first.

7. In Aurora’s model, the current state of the quest journal is rarely queried by any part of the game system; it’s just a method of exposing information to the player and assiduously avoiding the perils of the Eliza effect.

8. As Dominic Arsenault pointed out during the blog-based peer review of this book, dialogue trees can provide a variety of interaction experiences for similar underlying structures. Games such as KotOR present the player with a selection of possible texts for their character to say (presumably verbatim). When one is selected, the player character does not perform it on-screen; the act of player selection takes the conversational place of that text being uttered by the character. A rather different approach is taken by games such as Indigo Prophecy (Cage et al. 2005) and Mass Effect (Hudson et al. 2007). In these, the player is presented with short indications of the types of things that their character can say. After one is selected, a potentially surprising performance from the character will result—as when, early in Mass Effect, players may be taken aback to find their character striking Dr. Manuel, after choosing less supportive alternatives in the conversation with the shocked scientist. Indigo Prophecy also adds an interesting timed element to the selection of dialogue options, enforcing something more like conversational pacing (in games like Mass Effect characters will await a response indefinitely), and putting some pressure on the strategic decision making involved. A variety of other interface strategies exist, from having players choose keywords for discussion topics (e.g., The Elder Scrolls IV: Oblivion [Rolston et al. 2006]—which also offers literal things to say and actions to take), to the selection of punctuation and icons (e.g., Sam & Max Hit the Road [Purcell et al. 1993]—which is an earlier example of surprising character performances from dialogue tree options, as Darius Kazem pointed out in a comment on Grand Text Auto). An overview of common dialogue system organizations is provided in Gamasutra by Brent Ellison (2008).

9. The “[END DIALOGUE]” at this point in the image is not correct, but an artifact of Aurora. The same is true after “You still think of his ring?”

10. In an actual game that included the Grandfather’s Ring quest, the grandmother conversation would need another flag structure, given that the initial dialogue must be different if the ring has been found before the first conversation with the grandmother.

11. Further, as Benjamin Grandis pointed out during the online peer review of this book, a number of RPGs “experiment with the ability for even your party members to grow to resent you, often for quite complex reasons which tie into the plot and yet are quite avoidable.” In some cases the results of character relationship choices are dramatic. For example, while playing Mass Effect, it is quite easy to select dialogue and quest options that fail to create trust with party member Wrex. This can lead to his death—in some scenarios, at the hand of the main player character.
12. And of course *Eliza* also makes an occasional fallback on precreated transformations when the most recent audience statement contains no keywords.

13. This is generally done quite artfully. Even an exceptionally awkward-seeming opening exposition is later given deeper resonances as players progress through the main quest. It is revealed that the characters surrounding the player character, who at first seem to be involved in laborious exposition, are actually part of an elaborate deception—aimed, of course, at both the player and her character. At the same time, these interactions also serve the necessary purpose of filling in the player about things her character would normally already know. Such strategies are important not only to RPGs but also to digital fictions more generally, and in some cases especially so. For example, Jeremy Douglass (2007) has written insightfully about a strand of interactive fiction work that has turned the discovery of the player character’s identity and situation into the central element of the work.

14. In addition, *KotOR* also provides a quantification, exposed to the audience, of the different ethical approaches available in many RPGs. This is connected to the notion of the “force” that pervades the Star Wars fictional universe. Some quest actions are considered “light side” (those that display traits such as compassion and generosity) while others are considered “dark side” (those that exhibit traits such as selfishness and a taste for needless violence). When these are completed, the player character is awarded a certain number of points that move them along the spectrum between light and dark. Often the dark option is an easier route through the quest objectives, creating a game mechanic for the deep-rooted Star Wars theme of the temptations of the dark side. These choices also alter how the rest of the game is played; for example, some items may only be employed by those on the light or dark ends of the spectrum.

15. Attempting to reproduce this outcome, Colin Wheelock discovered that Ahan Matale’s irrational behavior is only exhibited if the player scores a “light side” point during the confrontation outside the Sandril compound.

16. Another way of avoiding such problems was employed in the next game on which Karpyshyn was the lead writer: *Mass Effect* (Hudson et al. 2007). The long-term development of personal relationships with crew members is almost entirely restricted to conversations that happen while on board the ship, where characters are mostly separated from one another. This means that interaction contexts for each character are much more restricted. In addition to such work to avoid problematic complexity, *Mass Effect* also does more to account for what has already been learned on other planets in the dialogue for each individual planet. The result is a feeling of less temporal coherence in the development of personal relationships but more temporal coherence in the other elements of the plot. While I had been hoping for interactions in *Mass Effect* to be less restricted than *KotOR*s, moving in the opposite direction did lead to an overall increase in logical coherence.

Which is not to say that logical problems don’t crop up, though they are generally at a smaller scale. For example, when exploring the planet Trebin in *Mass Effect* I noticed something on the vehicle’s scanner. It turned out to be a mine entrance. With Kaidan and Tali in my party, I entered the mine and fought a number of Husks. After the battle concluded my party members commented that the researchers (whose disappearance led to our visit) must have found something that turned them into monsters.

Next we exited the mine, got back in our vehicle, and drove over to the research base. Kaidan, who had just been commenting on the fate of the researchers, said, “Where is everybody?” Then we entered the base and looked at one of the computers. Tali said the logs indicated that the team discovered “some kind of alien
Expressive Processing

technology”—the technology she’d just been talking about, the horrible results of which we’d just seen.

17. As of this writing Mechner is also involved in filmic storytelling from another angle—as screenwriter and executive producer of the upcoming Walt Disney Pictures/Jerry Bruckheimer feature film Prince of Persia: The Sands of Time, directed by Mike Newell and starring Jake Gyllenhaal, Gemma Arterton, Sir Ben Kingsley, and Alfred Molina.

18. This is one of the reasons that PoP has received a significant amount of scholarly attention. I am aware of three authors who offer particularly extended engagements. Drew Davidson (2008) provides a detailed, thoughtful reflection on the experience of playing through the game—with particular attention to the relationship between story and play. Barry Atkins (2007) offers a reflection on death, time, and narration (topics central to this game, but important to games in general) in his chapter on PoP for Videogame, Player, Text. Jason Rhody (2005, 2008) has written intriguingly both on the way PoPs interface establishes and controls point of view and, in a response to an essay by Mechner, on how game fictions must be understood dually through interface and involvement in the fictional world, and that successful examples (like PoP) operate in a manner that conjoins narrative and ludic development of the experience.

19. These scripted sequences can be prerendered “cut scenes” (essentially, computer animation files played by the game system), in-engine cut scenes (scripted scenes rendered by the game engine, rather than another animation system, but still removed from player control), or scripted sequences that take place in the engine and during which the player can still control aspects of the game.

20. The parallels with U.S. foreign policy at the time of the game’s production, before and during the 2003 invasion of Iraq, soon began to seem prophetic. They are also unlikely to be accidental, given Mechner’s description of an “anti-war theme” that “underlies the whole tale” (2007, 115). Of course, the sands seized by the second Bush administration yielded a less literal curse.

21. I am using the words game fictions generically, to refer to fictions within games. In More Than a Game, Barry Atkins uses the words game fiction to refer to those games he sees as “having a central narrative impetus, that develop story over time, rather than simply repeat with minimal difference in a move from level to level of increasing excess” (2003, 20). Jason Rhody (forthcoming) develops the term further in his dissertation, writing: What, then, is game fiction? In short, the term is intended to describe a category of game that draws upon and uses narrative strategies to create, maintain, and lead the user through a fictional environment in order to actualize a narrative and ludic goal. . . . [G]ame fictions are competitive, ergodic, progressive (and often episodic), and their primary goal is one of actualization. Game fictions are not limited to a single medium, although a game’s particular materiality—should it include dice mixed with a game board, paper, or even the imagination—often reveals much about the game fiction in question.

Both Atkins and Rhody participated in the blog-based peer review of this book—and both seem fine with me using game fiction generically, while still using the words as a more specific term in their own work.