USER CENTERED SYSTEM DESIGN

New Perspectives on Human-Computer Interaction

Edited by
Donald A. Norman
Stephen W. Draper

CRC Press
Taylor & Francis Group

Interface as Mimesis

Brenda Laurel

An Initial Assertion

In a recent article entitled "The Importance of Interaction," theorist Robert M. Dunn observes: "When we compound the 14 categories of interaction with the 25 different possible states of discourse in any dialog, we see that there are 350 theoretically distinct forms of interaction. This is the primary reason why building interactive systems is still an art" [Dunn, 1984].

What? All those categories, and it's not science yet?

The design of the human interface to interactive systems is "still an art," and likely to remain one, not because design "science" has not yet come of age, but because an interface is by nature a form of artistic imitation: a *mimesis*. It need not be a dark art, fraught with mysterious rituals and ethereal constructs. If designing interfaces feels like painting on cave walls by flickering torchlight, it is only because we, the designers, have not availed ourselves of better illumination: the science of the mimetic arts, poetics.

Many of us in the computer industry have been struck by the thrashing about of software designers and analysts as they attempt to identify actionable criteria for the design of good interfaces. Since the birth of the consumer software business in the '70's, I have watched software engineers grapple with the issues of engagement and ease of use (designers, as we know, remain a "luxury item" in the consumer end of the industry). Good aesthetic intuitions and crisp logic have marked those engineers who have created successful products. Yet "artistic" concerns—indeed, the very idea of an artistic approach to design problems—has been blushed about as "fluff," while the "harder" disciplines of psychology and computer science are assiduously mined for guiding principles.

As a student of theatre and dramatic criticism, as well as a software designer and researcher, I am impressed again and again by the applicability of dramatic theory to the problems of interface design. I am also impressed by the availability and comprehensiveness of that theory. The dramatic theory of Aristotle, for instance, has been around for 2000 years. It can guide us at levels far deeper than the "fluff" usually associated with art ("What color should the border of my menu be?"). The purpose of this chapter is to advocate an approach to the problems of interface design—both theoretical and practical—which utilizes dramatic theory.

The Purpose of an Interface

According to Aristotle, the *end cause* of a thing is the function that it is intended to serve; that is, what it is supposed to *do*. The end cause of a play, for instance, is "the arousal and catharsis of emotion." By emotional arousal, we mean that the audience becomes emotionally involved in the action of the play. The term *catharsis* is best understood as a kind of closure: all the questions and issues that have been raised are brought to a satisfactory conclusion, so that the audience is not troubled by a bunch of loose ends. For catharsis to occur, the audience's experience must have a rational component; that is, the emotional content of the play is orchestrated so that the events represented are, on some level, both

This article appears in *User Centered System Design: New Perspectives on Human-Computer Interaction*, D. A. Norman and S. Draper, Eds. Hillsdale, NJ: Lawrence Erlbaum Associates, 1986.

probable and believable. Thus the end cause of a play is the emotional and rational *engagement* of its audience. It is Aristotle's premise that such engagement, when successfully achieved, is intrinsically pleasurable.

A specific play may be intended by its author to teach the audience something, to incite the audience to action, or to provide an object for aesthetic appreciation. But these second-order functions can only be fulfilled if the play succeeds in achieving its primary end: the *engagement* afforded by the dramatic form. If the audience loses interest, falls asleep, or cannot follow the action, none of the other desired effects can occur.

Likewise, an interactive computer program may be intended to enable its user to do a variety of different things—find information, compose and format a document, play a game, or explore a virtual world. The user's goals for a given application may be recreational, utilitarian, or some combination of both, but it is only through *engagement* at the level of the interface that those goals can be met. An interface, like a play, must represent a comprehensible world comprehensibly. That representation must have qualities which enable a person to become engaged, rationally and emotionally, in its unique context.[1]

A Question of Form

While the end cause of a thing is that which it is intended to do, the *formal cause* of that thing is the shape it must take in order to do it. Form provides the means whereby the end cause can be realized, through the exercise of its powers to engage, to evoke emotional and rational response, and to provide *catharsis*—emotional and intellectual closure.

To persuade ourselves that it is form, and not content alone, that has these powers, we need only consider the differences between a good play and a "day in a life." During a normal day, a lot of time is wasted with the mundane (scratching, staring out windows, sitting in traffic); the relations between causes and effects are often murky; things rarely build to a climax and then come to a satisfactory conclusion. Selecting incidents from that day and arranging them in a dramatic *form* will create a universe in which our thoughts and feelings make sense and assume a coherent pattern. It is the form of our "day-play" which orchestrates our experience, the form which creates an organic whole from the chaos—a whole in which we can engage with a special kind of pleasure.

The form of a play and that of an interface are similar in a fundamental way: both are *mimetic*. A *mimesis* is a particular kind of representation. It is a made thing, not an accidental or arbitrary one: using a pebble to represent a person is not mimetic; making a doll to represent him is. The object of a mimesis (e.g., that which it is intended to represent) may be a "real" thing or a "virtual" one. A painting, for instance, might be a mimesis of a real landscape or an imaginary one; a play may be a mimesis of events (literally, a series of actions) that are taken from history or that are entirely "made up."

A mimesis is a closed system; it has finite potential and is limited in some way. A play like *Hamlet* can be seen as a closed system in several ways. The most obvious is that it has a clear beginning, middle, and end: when the play is finished, all the potential for action that was present in the beginning has been exhausted. The play is also closed in the sense that new potential cannot be introduced. Laertes cannot threaten Hamlet with a light sabre, for instance, and King Claudius cannot suddenly sprout wings to

[1]Naturally, there is disagreement on this point. The critics point to the all-too-common case in which a user is willing to submit himself to a heinous interface and have a dreadful experience, as long as he is able to get the job done (my own experiences with UNIX Emacs come to mind). So might a drama buff endure a perfectly awful play, to the end of being able to chat about it later. But in seeking design principles for good interfaces, we must, it seems to me, concern ourselves with the best case, and ask, not what the user is willing to endure, but what the ideal user experience might be, and what sort of interface might provide it.

escape his fate (although in other mimetic worlds the potential for such actions exists; e.g., *Star Wars*). An interactive example of the destructive effects of introducing new potential in midstream would be a spreadsheet program that suddenly invites the user to play a few rounds of "Space Invaders".

This is not to say that a "closed" system is not extensible. The world of "Star Trek," for instance, is the host for countless dramas and narratives. But it is precisely the closed nature of the "Star Trek" world that allows new stories and dramas to be created and recognized as part of a larger whole. When new characters, things, or actions are introduced (e.g., the appearance of Kirk's son or the invention of a new star drive), they must conform to the principles and probabilities already established in the closed mimetic world; in other words, such "new" materials do not represent the introduction of new potential, but are rather new formulations of existing potential within the closed mimetic system. The trilogies of ancient Greek drama, as well as contemporary soap operas,[2] are based on mimetic contexts that are "closed" in the same sense.

Because a mimesis is a closed system, it is theoretically completely knowable. Aspects of the mimetic context that are not immediately apparent may be deduced from the principles and probabilities that are built into it. Finally, a mimesis is internally consistent; e.g., a character in a play will not exhibit a radical mood shift without some identifiable and plausible reason (even if that reason is apparent only retrospectively).

In the *Poetics*, Aristotle describes and analyzes those forms of mimesis that have what we have called "pleasurable engagement" as their end cause—the various species of poetry, including narratives and epics, dance, musical performances, and drama. These species are differentiated from one another in terms of the *means*, *manner*, and *object* that each employs in its representations. In this analysis, more similarities between plays and interfaces emerge:

> In both forms, the *means* consist of signs—linguistic, imagistic, etc.—that are arranged in harmonious, or pleasing, ways (the pleasure afforded by such harmony has both emotional and rational components). While other mimetic forms use subsets of these means (i.e., music employs rhythm and harmony but not language), dramatic and interactive forms (at least potentially) utilize all of them.
>
> The *manner* of representation in both plays and interfaces may be termed "enactment"; that is, things are meant to be acted out rather than described. It is worthwhile here to emphasize the distinction between *narrative* and *dramatic* forms. In literature, narrative forms include things like novels and stories, while plays are dramatic in form. The difference lies in the *manner* in which each form presents its representation. Stories and novels are intended to be read, while plays are intended to be acted out. A similar distinction can be observed between certain kinds of computer applications and the human interface to them. When conceptually divorced from its interface, a text-only database or adventure game may indeed be "narrative" in form—it is a descriptive representation. However, the interface to such an application is primarily concerned, not with description, but with *action*. Novels and databases describe things; plays and interfaces act them out. In the process of *enacting* the transactions between user and system, interfaces, like plays, are represented in a medium that includes elements like music and spectacle as well as language.

[2]Michael Lebowitz of Columbia, in his work on an AI system that will generate serial-like narratives, provides operational means for maintaining internal consistency in the creation of new characters and events in his soap opera world, UNIVERSE. See [Lebowitz, 1984].

> The *object* of both kinds of representation (plays and interfaces) is a whole action—or interaction—with a beginning, middle and end.[3] Aristotle emphasizes that plays represent *actions* as their primary object, and that characters are represented only to serve the action. Likewise, *interaction* is the primary object of an interface, with other objects—agents, environmental elements, etc.—represented only as called for by that primary object.

What does all this do for our understanding of a good interface? It provides a basic way of thinking about things. The consideration of means gives us grounds to believe (besides our good intuitions) that good interfaces are ultimately multi-modal. The lawful relations among such "modes" (visual, auditory, lexical, etc.) are described in Aristotle's theory, as well as principles for their orchestration. "Enactment" as the manner of representation steers us away from interface structures that depend upon description and supplication and toward those that focus on enabling the user to *act*. The primacy of action as the object of representation gives us (inter-)action and its component parts as the primary structural elements of the interface, as opposed to objects, menus, "screens," command modes, or other secondary sorts of entities. "The thing itself" emerges as a particular, real-time interaction of a particular shape, with parts and their relations defined by the nature of that interaction—an organic whole.

Mimesis and Interaction

The most important distinction between a play and an interface is that an interface is *interactive*, while a play is not. The audience in a theatre can have no substantive effect upon the action of a play. Their applause, laughter, or inattention may influence individual performers to be louder or slower or to make more faces, but nothing an audience can do will change one word of dialogue or one event in the plot—least of all its outcome. The action of the play is pre-determined. The plot, with its characters and environments, has been painstakingly crafted by the playwright to be an organic whole, an instance of dramatic form, with all its powers and pleasures.

An interface, on the other hand, is literally co-created by its human user every time it is used. We have said that the interface represents a *whole interaction*, just as a play represents a whole action, with beginning, middle, and end. The potential for that interaction exists within the system, as well as within the user as he structures goals in relation to it. Between them, those two fields of potential must contain the predispositions and constraints necessary to guarantee that the interaction, when it occurs, will be an organic whole. That whole is collaboratively formulated in real time by user and system. The user functions as an agent (or character, if you like) within the mimetic context—the context offered by the representation. Just as changing the goals and actions of a character in a play will change its outcome (and thus the shape of the whole action), so the user, through his actions, has a profound effect on the whole interaction.

Few would disagree that the function of the interface is to present a representation—a context—which enables a person to interact coherently with the underlying application. The idea that the user must therefore be engaged in the mimetic context is an obvious corollary. But obvious as it seems, the leading traditions ("metaphors," if you like) in interface design contradict this notion. Consider the following examples:

> *A young woman is playing a popular adventure game on her home computer. She types a command: "Go north." Words appear on the TV screen: "YOU ARE IN A DARK CAVE."*

---

[3]This observation is not meant to preclude interactive representations that take the form of "extended," re-usable environments (as opposed to stand-alone applications). But each interactive encounter or session, like the separate plays in a trilogy or single episodes of a soap opera, must offer sufficient closure (and catharsis) to make the experience pleasurable. Many of those who hated "The Empire Strikes Back," for instance, felt that the piece did not have an adequate ending, even though they knew it to be part of a trilogy (knowing that we would have to wait at least three years for the next installment didn't help).

"Damn," she types. "I DON'T KNOW THAT WORD," replies the screen, unperturbed. "A NASTY TROLL IS APPROACHING FROM THE EAST." "Kill the troll," she types frantically. There is a suspenseful pause. "THE TROLL IS DEAD. YOUR SWORD HAS BEEN BROKEN." She queries the system, "How?" "I DON'T KNOW THAT WORD."

A few hours later, the same young woman is trying to edit her "recipe" database. Having discovered that it is vile, she is attempting to remove the recipe for "Fish Bisque." She consults her user manual to find the command that invokes the file system editor. Then she must select the command for displaying files, then select the file to be deleted from the list of filenames, then return to the command menu and select the "delete" command. "Why can't I just delete Fish Bisque?" she moans.

These two interfaces have one important thing in common: they create a situation in which the user is not doing what she wants to be doing. In the first case, she wants to have an adventure in a fantasy world. In the second, she wants to yank a rotten recipe from her collection. In both cases, however, the interface insists that *what she is doing is using a computer*. This fixation springs from the idea that the computer is a *tool*.

The notion of the interface exemplified above uses programming as its model for human-machine interaction. The logic behind the "tool metaphor" goes like this: regardless of what she *thinks* she is doing (e.g., playing a game, searching a database, or designing a cathedral), the end user is *actually* using the computer as a tool to carry out her commands, just like a programmer. It follows, then, that what the user is interacting with is the computer itself, with outcomes like game-playing and database management as secondary consequences of that interaction.

Of course, that is silly. Just as the behind-the-scenes workings of a theatre are distinct from the action of a play produced in it, so a computer is distinct from any representation that it presents. The end user is not interested in *making* a representation (like a programmer); she wants to move around *inside* one. The context in which she wishes to operate is the *mimetic context*.

It seems to me that the "computer as tool" convention is an artifact of the rather haphazard evolution of interface design, as well as of the evolution of computer usage. The notion of dedicated applications with end users who were not programmers did not spring crisply from the thigh of Zeus. When things were far enough along for somebody to notice the dissonance created for users by the duelling contexts present in most interfaces (namely, the mimetic context and the context of the computer as tool), a cognitive patch was developed to bridge the gap: the convention of the *intermediary*.

The intermediary is an ill-formed presence or persona that belongs wholly to neither context, but attempts to mediate between them for the user. The user tells the intermediary (read "interface") what she wants and the intermediary takes care of making it happen. In the adventure game example, the intermediary seems to act as an agent for both the program in its evaluation of user input ("I DON'T UNDERSTAND THAT WORD") and as an agent for the user in its performance of functions like "inventory" and "look." It also "stands in" for the user: it swings the sword, takes the lumps, and reports what happens. In the file management example, the intermediary takes the form of command menus that are invoked in order to activate processes in the program that will create the desired results. The user does not have the experience of pushing files around, stowing them and grabbing them, or blowing them away. Instead, she has the experience of communicating with the file management intermediary.[4]

[4]"Intermediaries" and I describe them here are fundamentally different from what others [Brennan, 1984] have referred to as "agents." In Brennan's formulation, for instance, an "agent" is an explicitly represented persona that is intended as part of the mimesis that constitutes the interface, while the intermediary is a vague presence that attempts to bridge the gap between a "computer as tool" interface and the mimetic context of the application. An interface agent, when represented mimetically as an interface convention, can successfully assist the user in performing complicated, odious, or overly-detailed tasks *without*

The fundamental problem with both the tool metaphor and the intermediary convention is that they rob the user of the experience of direct agency—the ability to *act*—within the mimetic context. What the user gets to do is persuade the system, via the interface, to take the action she would like to take herself. But isn't it sufficient to enable a user to affect a process or outcome, by whatever means? What is our justification for insisting upon direct agency for the user *within* the mimetic context?

Here we return to that "obvious" assertion we made a while ago. We insist upon full participation in the mimetic context because it is quite simply the *experience* we desire. It is through such direct participation that the full pleasure of the mimetic form is available to us. By becoming *agents* in the mimetic context, our participation is active rather than descriptive, in keeping with the nature of interactive form (the *manner* of the mimesis is enactment, as we have noted, and its *object* is action itself).

Immersion in the mimetic context of a play by an audience member is possible through an act that Samuel Taylor Coleridge called "the willing suspension of disbelief." As an audience member, I know that the people on the stage are actors and that the castle parapets are cardboard, but I choose, in order to have the pleasure of unencumbered emotional and rational participation, to suspend that knowledge for the duration of the play. My experience will be clobbered if I am constantly reminded of what I have chosen to disregard—I do not wish to watch the stage manager pulling the curtain and calling the cues. If I wanted to invent a form of mimesis that was more participatory—more interactive—than a play, why in the world would I choose to make myself even more aware of the suspended facts by becoming the stage manager? On the contrary; I would want to walk into the dramatic world, participate in the action, and change the world by being in it.

One can imagine such an experience as the ultimate adventure game, but can something like it be possible within the humble world of a spreadsheet or a database manager? Aren't we stretching things a little here? It is my position that the principles of mimetic interaction remain the same in both kinds of worlds: the labyrinth of Zork is no more or less real than the terrain of Lotus; both are representations enacted on the same "stage." In neither case do we want to deal with the man behind the curtain; he is extraneous to both the form and the experience.

The conclusion can only be that a particular kind of relationship between the user and the context is appropriate to the nature and form of an interactive mimesis. The term I have chosen to describe it is *first-personness*. The first-person metaphor is grammatical: the personness of pronouns reflects where one stands *in relation to* others and to the world. Most movies and novels, for example, are third-person experiences; the viewer or reader is "outside" the action, and would describe what goes on using third-person pronouns: "First he did this, then they did that." Most instructional documents are second-person affairs: "Place the diskette in drive B"; "Honor your father and mother." Operating a computer program with an intermediary interface is a second-person experience: the user makes imperative statements to the system and asks it questions; the system tells the user what to do and what it has done ("File access denied, please try again"). Walking through the woods is a first-person experience; so is playing cowboys and Indians, writing a letter, or wielding a hammer.

Based on an understanding of interface as mimesis and of the necessary "first-person" relationship between user and context, design principles can be formulated and tested. Such principles are intended to indicate how the materials and structure of a mimetic world can be orchestrated to create the experience we desire.

---

violating the mimetic context. Such an agent must be constructed according to mimetic principles delineated in the section on selection criteria, below.

When we speak of the user as an agent, we mean that the user is an initiator of action. In this context, the opposite of "agent" is "patient" -- one who sits passively by while action is performed by some other entity.

Some Design Principles

Representational Aspects of First-Personness

Personness is affected by the representational aspects of the interface; that is, how the user's choices and actions are introduced into the system, and how the activities of the system are represented to the user. The notion of a first-person interface seems to be more intuitively obvious in applications which, like video games, have the experience itself as the objective of the user.

In the Atari game *Pole Position*, for example, the user "drives" a simulated race car down a track. The user controls the speed of the car by pressure on a pedal that is analogous to an automobile accelerator. The ability of the user to participate in the race-driver fantasy in a first-person way would be significantly lessened if his only means of controlling the speed of the car were to specify speed numerically from a keyboard. Likewise, first-personness would be diminished if the effect of the user's pressure on the pedal were reflected, not by an animated representation of a race car, but by a numerical display of its speed.

In a task-oriented activity like file management, the representational aspect of first-personness can also be explored. In our Fish Bisque example, the functions of file management are represented by a series of command menus. At each level, the selected command tells the system something about the file manipulations desired by the user. But what is really going on? The user wants to get rid of a file. She might, via a touch-sensitive screen, draw a line through the region of text or the name of the file she is trying to remove. She might, via a natural-language interface with speech recognition, simply say, "I want to delete the recipe for Fish Bisque."

The underlying principle here is *mimetic*; that is, first-personness is enhanced by an interface that enables inputs and outputs that are more nearly like their real-world referents, in all relevant sensory modalities. The intuitive correctness of this notion is witnessed by the direction of technical evolution in the areas of simulators and games—toward higher resolution graphics and faster animation, greater sound capabilities, motion platforms, and mimetic input devices like force-feedback controllers. In product-driven applications, new technologies are allowing researchers to replace indirect or symbolic representations and manipulations with direct, concrete ones; e.g., physically pointing or speaking as opposed to typing, spatial and graphical representation of data as opposed to textual representation, etc. Likewise, the evolution of natural-language interfaces is beginning to replace the elaborate conventions of menu- and command-based systems with systems that employ language in ways that are mimetic of real-world activities like conversation and question-and-answer dialogues.

Interactive Aspects of First-Personness

First-personness is affected by the kinds of choices a user may make and the patterns of choice that emerge from the interaction. Three interactive aspects of first-personness are frequency, range, and significance.

The opportunity for a user to take action may occur more or less frequently in an interactive work. *Interactive frequency* is a measure of how often user input is enabled. Near one end of the frequency continuum is a program with only a few clearly delimited interactive nodes. The other extreme is exemplified by action games like *Pole Position*, in which the user makes apparently continuous, real-time input.

We must be careful to distinguish between how often a user actually makes input, and how often he feels he is able to. Especially in product-driven applications, the frequency criterion of first-personness can be achieved simply by making the user aware that he can express himself at any time.

Interactive range describes the range of choices available to a user at a given moment in the interaction. That "interactive moment" may be a distinct node or a slice of "continuous" interaction. A binary choice,

such as the use of a "fire button" in an action game or a yes-or-no question in a file management session, has the narrowest interactive range, even though the *consequences* of that choice may be of great significance to the whole. In natural-language-like interfaces that interpret user input using keywords, the interactive range is determined by the number of words that are both recognized and functional at a given interactive moment. Interactive range is also affected by the user's perception of the choices available to him. When there is a discrepancy between the number of choices actually recognized by the system and those perceived by the user, the user's assessment takes precedence.

Interactive significance is a measure of the impact of the user's choices and actions upon the *whole*. In a product-driven activity, that whole is the whole *outcome*, or result, of an interactive session. In a process-driven activity, the whole refers to the whole *experience*—the result of collaboration between user and system. In an interactive fantasy game, interactive significance measures the user's effect on the whole *plot*. In the CyberVision interactive story program *Rumplestiltskin*, for instance, the user is allowed to choose which of three characters will serve as the queen's messenger. The choice is of relatively little significance, however, in that the actions performed by the messenger in the story remain the same regardless of the user's choice, hence the choice has little impact on the plot.

In the menu-driven approach to file management portrayed in our example, a significant action—deleting a file—is broken into chunks, none of which may be interpreted as a whole (and potentially "dangerous") action. The design will not permit the user to directly express her knowledge of the state of her files or what she wants to do with them. Thus the interface prevents the user from making choices with high interactive significance in the name of protecting her from herself.

The user's experience is enhanced by life-likeness in the realm of agency—the experience of one's ability to *act*. First-personness is most completely realized at the extreme end of each of the interactive variables' continuum: frequency is continuous; range is infinite; significance is maximal.

Designing User Constraints

Constraints are limitations. They may be expressed as anything from gentle suggestions to stringent rules. People are always operating under some set of constraints: the physical limitations of survival (air to breathe, food, and water); the constraints of language on verbal expression; the limitations of social acceptability in public situations (e.g., wearing clothes). The ability to act without any such constraints is the stuff of fantasy and myth—the power of flight, for instance, or the appeal of immortality. Yet even such fantasy powers can be lost by the failure to comply with other, albeit magical, constraints (witness Prometheus). It is difficult to imagine life, even a fantasy life, in the absence of any constraints at all.

The user of an interactive system is subject to some special kinds of constraints. Some constraints arise from the technical capabilities of the system itself: if the system has no speech processing capability, for instance, the user must employ the keyboard for verbal input, and is constrained by its vicissitudes—the "QWERTY" layout, the presence or absence of function keys, etc. Other constraints arise from the nature of the activity that is appropriate to the application. The user of an adventure game cannot perform calculations with his computer while the game is running on it, even though the computer is capable of complex mathematical operations.

Are constraints simply a necessary evil, or do they perform some positive function for users? In idle fantasies about interactive systems, people tend to imagine magical spaces where they can and do whatever they wish—like gods. Even if such a system were technically feasible, the experience of using it might be more like an existential nightmare than a dream of freedom. When a person is asked to "be creative" with no direction or constraints whatever, the result is often a sense of powerlessness or even complete paralysis of the imagination [May, 1975]. Constraints provide the security net that enables people to take imaginative leaps.

Constraints exist to prevent the user from doing things that would cause the system to blow up or resort to second-person intervention. They keep the user's activities within the bounds of the mimetic world. In exchange for his complicity, the user experiences increased potential for effective agency, in a world in which the causal relations among events are not obscured by the randomness and noise characteristic of open systems (like "real life").

How should constraints be determined and expressed? The standard techniques for introducing user constraints—second-person transactions like error messages, or delimiters of interactive frequency and range like explicit nodes with choice menus, for example— are almost always destructive of first-personness.

User constraints can be either explicit or implicit. Explicit constraints, as in the case of menus or command languages, are undisguised and directly available to the user. When the user is in doubt about the "legality" of a certain choice or action, he should be able to find the rules and protocols of the system straightforwardly expressed, either in his manual, or in an on-line "help" facility. Implicit constraints, on the other hand, are inferred by the user from the behavior of the system. In *Zork*, for example, the user is not given a list of words that the language parser understands, but is informed via an error message (`I DON'T UNDERSTAND THAT WORD`) when he uses one that is unfamiliar. Implicit constraints may also be identified by the user when the system fails to allow him to make certain kinds of choices. There is no way, for example, to negotiate with the Zylons in a game of *Star Raiders*, or to get *Word Star* to revise your latest paper.

Explicit constraints can be used without damage to first-personness if they are presented before the action begins. A good example is the determination and expression of rules in child's play, which occurs before play actually begins and creates a contract binding the participants to behave within certain constraints. Once the action has started, however, explicit constraints prove disruptive—an argument about the rules can ruin a perfectly good session of "cowboys and Indians" ("Wait a minute—who says Indians can only be killed with silver bullets?"). Implicit constraints are preferable during the course of the action, simply because the means for expressing them are usually less intrusive than those used for explicit constraints.

Constraints may also be characterized as extrinsic or intrinsic to the mimetic action. Extrinsic constraints have to do, not with the mimetic context, but with the context of the user as operator of the system. Avoiding the "reset" and "escape" keys during play of a game has nothing to do with the game world and everything to do with the behavior of the computer. Frequent breaks in a text editing session to protect your files from power failures and system crashes have nothing to do with the process of composing or editing text (*FinalWord*, an IBM-PC permutation of Emacs, solves this problem by automatically updating a swap file on the program disk every seven seconds).

Extrinsic constraints, when they cannot be handled invisibly (as in the *FinalWord* example), should be expressed in terms of the mimetic context. If the "escape" key is defined as a self-destruct mechanism, for instance, the constraint against pressing it in the course of flying one's mimetic spaceship is intrinsic to the action. The user need not "shift gears" to consider the effect of the key on the computer that is running the game program. Expressing constraints in this manner preserves the contextual aspect of first-personness.

Constraints should be applied without shrinking interactive range or significance as experienced by the user: they should limit, not what the user can do, but what he is likely to think of doing. Context is the most effective medium for presenting such constraints. The user's ability to recognize and comply with implicit, context-based constraints is a common human skill, exercised automatically in most situations, and not requiring concentrated effort or explicit attention. It is the same skill that a person uses to determine what to say and do when he interacts with a group of unfamiliar people—at a cocktail party, for instance. The limitations on his behavior are not likely to be explicitly known or consciously mulled over; they arise naturally from his growing knowledge of the context in which he finds himself. In the *Phone Slave* (an intelligent phone-answering and message-taking system developed at MIT), for example,

users are successfully constrained to provide appropriate and recognizable natural-language input simply by the conversational questions that the system asks [Schmandt and Arons, 1983].[5]

Selection Criteria

The notion of the interactive representation (interface) as an organic whole allows us to articulate some principles for the selection of materials to be included in that representation. Those materials are contributed by both the system and the user during interaction, and are subject to the same criteria. In the *Poetics*, Aristotle provides a general principle for inclusion: "...that which makes no perceptible difference by its presence or absence is no real part of the whole." From this principle he derives selection criteria for the inclusion of materials in dramatic works which can also be applied to interactive representations.

Aristotle describes the plot as the central action of a play—the object of its representation. Incidents which have no direct bearing on the plot should not be represented; e.g., there is no reason to include a scene where Hamlet brushes his teeth. We have all been annoyed by such gratuitous incidents in films or TV shows. My guess is that most of us have been annoyed by the same kinds of incidents in interactive works. A convention in the world of CAI, for instance, is to ask the student to enter his name at the beginning of an interactive session. The most that is usually made of this "incident" is a message that replies, "Hello there, Jimmy," before proceeding with the "meat" of the lesson. The name-entering incident (Jimmy and the computer say "hi") has nothing to do with the plot (Jimmy learns his multiplication tables).

Gratuitous incidents most often occur in the presentation of extraneous information. The appearance of menus, prompts, or "helps" whether you want them or not is one common annoyance; the appearance of "garbage text" is another. The mail facility on most large systems appends a header to displayed messages which contains a cacaphony of information that the user need not clutter his brain (or screen) with. Information about the path that a message took between nodes in a network, for instance, may be of functional use in delivering a response automatically, but its presentation to the user for his perusal is gratuitous in the context of reading and responding to electronic mail.

Another example of gratuitous incidents are bungled efforts to provide "intrinsic motivation" in educational programs (often based upon gross misinterpretations of [Malone, 1980]) by interspersing problem-solving or tutorial segments with pieces of games: if Jimmy solves three arithmetic problems successfully, he gets to spend 20 seconds playing a low-res version of something like *Asteroids*. Either the math problems or the game segments are gratuitous, depending upon Jimmy's point of view. The solution is either to eliminate one of the activities, or to re-shape the context so that it includes both; e.g., a starfighter simulation in which Jimmy solves math problems in order to operate the ship.

A second selection criterion in the world of drama involves the presentation of character traits. A trait should not be included in the representation unless it either eventuates in some action or sets up an important line of probability. We need never know, for instance, that Dr. Frankenstein is a great dancer (unless, perhaps, a dance with the monster shows up later in the plot, as in Gene Wilder's version).

Knowledge about a trait sets up expectations (probabilities) for the subsequent behavior of the possessor of that trait. Traits are rarely asserted explicitly; they are more often inferred from some action or contextual feature. If the user of a text editing system notices that there are commands that will position a cursor at the end of a word, a line, and a sentence, he may infer that a trait of the system is the ability to "go to the end of" various logical units. If he then discovers that there is no command to go to the end of a

[5]Ed Hutchins suggests street theatre as a good example of the process of designing context-based constraints. The improvisational actor in the "interactive" context of street theatre knows enough about his audience to set up a context that will elicit a predictable response. This is essentially the same technique that Schmandt employs with the Phone Slave: "When you ask people questions," says Schmandt, "they tend to give you answers."

paragraph, he must throw out his inference and start over. The trait should never have been implied in the partially consistent behavior of the system. System traits of a similar nature are identified by users on syntactic, visual, and kinesthetic levels [Hulteen, 1984]. In poorly designed systems, they lead users to expect the system to perform inaccessible or nonexistent functions.

The converse is also true: actions should not be represented for which the appropriate traits are not apparent. Peter Pan has traits which make it reasonable for him to fly; Perry Mason does not. Dials and counters that whirl away in the displays of many video games represent actions for which no meaningful traits exist. A game programmer confessed to me recently that he displayed the contents of an address register to "beef up" a video game "control panel." At SIGCHI '84, John Seely Brown proposed a mail system that would surreptitiously insert spelling errors in messages to create a climate of equality and informality in electronic communication. An action so sophisticated is not likely to spring from the humble traits of an electronic mail system (in fact, its users would quickly wonder whose traits were at work in the apparent sabotage).

Previously we have distinguished between a representation and its object. We have noted that the object of an interface is a whole interaction, just as the object of a play is a whole action. In both cases, only those traits which are necessary to represent the object should be included. This seems quite obvious, until one encounters a play in which the author or actor has attempted to create a complete human personality where a dramatic character with a limited number of traits is all that is required. As a student actor, I once auditioned for a character with one line—a maid who said something like "Yes, your lordship." I asked the director at the audition what kind of person the character was. His response was simply, "She is the kind of person who would say 'Yes, your lordship.'" I received a sound drubbing from the director when, in performance, my "maid" delivered the line with such adoring sincerity that the audience whispered conjectures about her love affair with the master of the house. Various dramatic theorists have noted that an audience will find such a character believable in the presence of very few traits [Schwamberger, 1980]. Indeed, the audience is distracted and misled when a dramatic character is too "noisy."

A case in point is the design of an electronic "interface agent" [Brennan, 1984]. One is tempted to design such an agent as a model of a human personality, with human-like knowledge and thought processes. Yet persuasive "computational personalities" like Eliza and Parry have been represented using a remarkably simple set of traits and memory capabilities [Boden, 1977]. The AI dogma of utilizing human-process models tempts us to misunderstand the nature of the design problem and the techniques appropriate to solving it. Creating simulations of human personalities is a blue-sky AI task of awesome proportions, and would probably require years of research and more than a few Crays to carry off. Creating an interface agent, on the other hand, is a problem in artistic representation. The key tasks are specification of the characteristics of the action in which the agent will participate, and identification of the traits necessary to represent that action mimetically.

A Fanciful Example

In my other life as a student of dramatic theory and criticism, I have been exploring ways to employ computer technology to apply certain dramatic principles prescriptively in the design of a system that will create interactive dramas [Laurel, 1983a and 1983b].[6] The notions of first-personness and interactive mimesis presented in this chapter have their origins in that project. The Interactive Fantasy (IF) system is a hypothetical beast that is intended to provide the user with an experience very much like becoming a character in a play, co-creating the plot by making choices and performing actions in a fantasy world. IF

[6]The completed dissertation is "The Design of a Computer-Based Interactive Fantasy System," The Ohio State University, 1986 (published by University Microfilms in Ann Arbor, MI).

sprang from my impatience with "dumb" computer games, a fascination with the idea of "interactive movies," and a perverse desire to become Captain James T. Kirk.

What would it really take to create such a "you-are-there" experience? The system would have to be able to figure out what the user-character was doing as he moved, spoke, and "lived" inside the fantasy world. In order to do that, the system would have to be implemented in a multi-modal interface environment, with user-sensing capabilities like speech recognition, body-tracking, and eye-tracking. It should be able to parse and understand gestures as well as utterances. It would have to know enough about playwriting to be able to weave a good plot on the fly from materials contributed by the user and those generated by the system itself (the behavior of "system characters," for instance). It would have to be able to represent the action in real time in the appropriate sensory modalities—doors that slam and characters that move and speak. As I began to investigate the "state of the art" in all of these areas—user-sensing, understanding and inference, story generation, "intelligent" animation, etc.—I discovered that "the technology" is not the limiting factor in the design of such a system. Most of the necessary capabilities exist, at least in rudimentary form. All that is needed, it seems, is a reason to put it all together—a reason compelling enough to attract the brains and funding for such an enterprise.

Then it occurred to me that there's a way in which the IF system is *all interface*. Such a system could provide interactive representations for worlds as diverse as Hamlet's Denmark and the *Encyclopedia Britannica* entry on ancient Egypt. Right now, we usually define an encounter with an "on-line database" as "searching for information." But what if we defined the activity as *finding* the information, or better still, *experiencing* it? We would no longer be "looking up information about the pyramids"— we would be climbing them, looking around their musty innards, reading heiroglyphs, or reincarnating pharoahs. Our notion of a "database interface" would be radically transformed.

The idea of interface as mimesis is based on the primacy of experience. It requires us to focus, not on what we can deliver within the constraints of current technology and convention, but what kinds of experiences we want to have with interactive representations. It provides a means for analyzing the resources from which such experiences can be built and some principles that can govern the orchestration of those resources. It takes as its model a theory that employs logic and aesthetics to create representations that engage humans in pleasurable ways. That is, after all, what we're trying to do.

References

Aristotle. *Poetics*. Translated by Ingram Bywater. New York: The Modern Library, 1954.

Boden, Margaret. *Artificial Intelligence and Natural Man*. New York: Basic Books, Inc., 1977.

Brennan, Susan. "Interface Agents." April 1984 (publication pending).

Dunn, Robert M. "The Importance of Interaction." *Computer Graphics World*, June 1984, pp. 71-76.

Hulteen, Eric A. "Comprehensibility as a Criterion for Interface Design." September 1984 (publication pending).

Laurel, Brenda K. "Simulation of and Interactive Fantasy System: Some Fanciful Scenarios." Atari Internal Memo # 14, September 1983.

Laurel, Brenda K. "Toward the Design of an Interactive Fantasy System: Description and Functional Requirements." Atari Internal Memo # 12, November 1983.

Lebowitz, Michael. "Creating Characters in a Story-Telling Universe." New York: Columbia University Department of Computer Science, 1984.

May, Rollo. *The Courage to Create*. New York: W. W. Norton and Co., Inc., 1975.

Malone, Thomas Wendell. "What Makes Things Fun to Learn? A Study of Intrinsically Motivating Computer Games." Ph.D. dissertation, Stanford University, 1980.

Schmandt, Christopher M. and B. Arons. "A Conversational Telephone Messaging System." *IEEE Transactions on Consumer Electronics*, Vol. 30, No. 3 (1984), pp. xxi-xxv.

Schwamberger, Jeffrey A. "The Nature of Dramatic Character." Ph.D. dissertation, The Ohio State University, 1980.