

# interactivity

Every language has its own special words, words that express an idea better than any other word in any other language. There really isn't any other word in the world that quite expresses the idea of the English word "fun." There are plenty of words for "humorous" or "enjoyable" or "playful," but nothing that quite catches the combination of informality, enjoyment, and near-vulgarity that our word "fun" connotes. We have shamelessly stolen such words from other languages. "Taboo," for example, is stolen from a Polynesian word; can you think of any other word that really captures the idea of that word? There are gobs of words we have stolen from European languages: entrepreneur, manana, schadenfreude, presto. But there's one word, a German word, that we haven't yet stolen that should be high on our list of targets: *schwerpunkt*. It means "focal point" or "concentration of effort point" or "central point of attack." It's a beautiful word because it expresses an idea that we just don't have in English: the notion that, in any effort, you have many necessary tasks, but there is one central task that must take first place in your considerations.

Consider, for example, what an army does. An army fights, right? And who does the fighting? Soldiers, right? But wait a minute: The soldier can't fight unless there's a cook who keeps him fed. No cook, no fighting. Ergo, cooks are just as necessary as soldiers. We therefore deduce that the cook is just as important as the soldier, because he's just as necessary. Same thing goes for truck drivers who bring the ammunition to the front, and the clerks who keep track of the food and ammunition, and the guys who dig the latrines, and so on. All of these people are

necessary, and so they are just as important as the soldier. But if you are the soldier, the guy who has to charge through fire and death to face the enemy, would you agree that these other people are every bit as worthy as you are? They may be necessary, and they may be important, but they're not central to the task. The soldier is the whole point and purpose of the effort. He's the *schwerpunkt*. And the soldier's fundamental task is to fight. So the *schwerpunkt* object in an army is the soldier and the *schwerpunkt* action is fighting.

Or consider a computer system. What goes into a working computer system? Well, there's a power supply—you can't have a working computer without a power supply. There's also a plastic box, a motherboard, lots of solder, plenty of wires and cables—all these things are absolutely necessary to a working computer, and I suppose you could argue that they are therefore important. But they're really not the heart of the computer. If you want to get down to the absolute core, it's got to be the CPU. That's the real essence of any computer. When you describe your new computer to someone, do you say, "It's a beige tower system with a 200-watt power supply and an 8 inch by 10 inch motherboard?" No, the very first thing you specify is the CPU and its clock speed: It's a 900MHz G4. That's the real *schwerpunkt* of the computer: its CPU. And what does a CPU do? It *processes*; after all, it is the Central *Processing* Unit. Thus, the *schwerpunkt* object in a computer is the CPU and the *schwerpunkt* action in a computer is processing.

So let us now determine the *schwerpunkt* of games. What is it that is absolutely central to games, the one element that is more than important, more than necessary, but indeed the entire point and purpose of games? The answer we immediately pounce upon is "play"; after all, what else do you do with a game than play it? Unfortunately, although that answer is certainly correct, it's not very useful; like the infamous term *fun factor*, we can never really pin down what elements of game design are useful to support good play. Can we snuffle about and develop a more useful answer?

## History

At this point, let's consider an important historical observation: Computers gave gaming a big boost. Sure, there were plenty of games before computers came along, but with the advent of computers, games suddenly seized a much larger portion of our consciousness. Before computers, gaming was a petty industry, employing a few thousand people at most. In the twenty years since personal computers burst upon the scene, gaming has suddenly exploded into a major industry, rivaling Hollywood in sales and employees. Something about computers made a big, big difference in games. What was that magic element?

If you answered "video," you get twenty lashes with a wet noodle. Video entertainment has been around since 1910! And the same thing goes for sound, music, or even text; all those media have been around for long before the computer made its appearance.

The magic element that computers brought to the party was their processing power, their ability to crunch numbers. That processing power made games so much more compelling. But processing power is an internal trait of computers—the user never directly experiences processing power. Instead, the user experiences processing power through the interactivity that the computer offers.

Consider the most common use of personal computers: word processing. What is it about word processing that is so damned useful? It's not the ability to print out text that's clean and neat—we've had typewriters that could do that for nearly a century before word processing became available. Moreover, word processing made typewriters obsolete in a matter of a few years. Clearly, the ability to get clean, mechanically generated text is not the primary appeal of word processing.

The real power of word processing lies in its interactivity. You type something on your screen, and if you make a mistake, it takes only a second to hit the Delete key and correct the typo. If you misspell a word, your spell checker will flag the error for you and you can fix it. If you don't like

the way a sentence reads, you can rewrite it. If you don't like the order of paragraphs, you can rearrange them with a simple cut and paste. It's the interactivity between user and computer that makes word processing so powerful. Take the interactivity out of a word processor, and all you've got left is a typewriter. Take the interactivity out of a spreadsheet, and all you've got left is a calculator.

Just as the *schwerpunkt* of computers is processing, so too the *schwerpunkt* of all software is interactivity—and this goes double for games. The turbo-charged interactivity that computers brought to gaming transformed the medium from wimp to superhero. Graphics, animation, sound, and music are all necessary to gaming, and they're all important, but they're not the *schwerpunkt*. Interactivity (sometimes called "gameplay") is the real *schwerpunkt* of games.

### Other Attributes of the Computer

There exists within the game design community a school of thought that holds interactivity to be nothing more than a useful component in the design of entertainment software. According to this school, the computer is a wondrous tool offering a variety of capabilities, such as graphics, animation, music, and sound. Interactivity is seen as just another capability to be used or neglected according to the tastes and intentions of the designer.


The one thing that computers can unquestionably do better than anything else is interactivity. And they do it much, *much* better than any other medium. They internalize the rules, carry out calculations, permit immensely more complicated behaviors, and present the results far better than any other medium. Many years ago, before computers were available, somebody published a board air combat game. That's right, a flight sim done on paper. The game was technically successful in that it did, after all, present air combat in a functional manner. But it was a failure in terms of gameplay; it didn't feel at all like flying an airplane.

Let me present the argument in the language of business. Interactivity is the "basis of competitive advantage" of the computer. Sure, the computer can do a lot of things, but the one thing that it does better than any of the competing media is interactivity. And the wise businessman always throws his resources behind his basis of competitive advantage.

You aren't convinced? Okay, how about a military maxim: "Fight on the ground of your own choosing." The wise general offers battle only on the terrain best suited to the strengths of his forces and takes maximum advantage of the weaknesses of his opponent. If his biggest advantage is in cavalry, he will choose to fight on flat open ground where the cavalry can do its job best. Sure, he'll fight with his infantry and artillery, too, but the tactical primacy will go to the cavalry.

So too it must be with interactivity. Yes, the computer can do graphics and animation and sound, but these capabilities are not the primary strength of the computer. Interactivity is the real strength of the computer, and it must be given primacy in our designs.

None of this suggests that graphics, animation, and sound should be eliminated from our designs. These are necessary supporting elements in the overall design. The better we are able to marshal them to heighten interactivity, the more successful our designs will be. But necessity does not convey equality. The ability to use a keyboard is absolutely necessary to a programmer, but typing skill is nowhere near as important to good programming as clear thinking.



Interactivity  
is the  
essence of  
what you are  
selling.

## So What Is Interactivity?

I have written an entire book on the subject of interactivity (*The Art of Interactive Design*, No Starch Press, 2002), and I suggest that you consult that book for more material on interactivity. However, I will present here a quickie synopsis of the core concepts.

There is one common experience we all share that is truly, fundamentally, interactive: a conversation. If you take some time to consider carefully the nature of conversations, you'll come to a clearer understanding of interactivity. A conversation, in its simplest form, starts out with two people. I'll call them Joe and Fred. The conversation begins when Joe expresses something to Fred. At this point, the ball is in Fred's corner. He performs three steps in order to hold up his end of the conversation:

1. Fred listens to what Joe has to say. He expends the energy to pay attention to Joe's words. He gathers in all of Joe's words and assembles them into a coherent whole.
2. Fred thinks about what Joe said. He considers, contemplates, and cogitates. The wheels turn as Fred develops his response to Joe's statement.
3. Fred expresses his response back to Joe. He formulates the words and speaks them.

Now the tables are turned; the ball is in Joe's court. Joe must listen to what Fred says; Joe must think about it and develop a reaction; then Joe must express his reaction back to Fred. This process goes back and forth until the participants terminate it. Thus, a conversation is a cyclic process in which each participant in turn listens, thinks, and speaks.

We can generalize the example of the conversation to get a definition of interactivity:

**A cyclic process in which two active agents alternately (and metaphorically) listen, think, and speak.**

Of course, the task of the game designer is to automate interactivity, to replace one of the participants in the conversation with a machine. We can therefore rephrase the problem of designing interactive entertainment as follows: "How can we program the computer to be an entertaining conversational (metaphorically speaking) partner?"

The overall answer is simple: In order to be a good conversational partner, the computer must perform all three steps in the conversational sequence—and it must perform them all well. It must listen well, giving the player the opportunity to say anything relevant to the situation. It must think well, coming up with interesting and relevant reactions to the player's input. And finally, it must speak well, expressing its reaction clearly. It's not good enough for the computer to perform one or two of the steps well, as compensation for performing a third step poorly. All three steps must be performed well in order for the computer to achieve entertaining interaction.

To substantiate this, I need only refer you to your own experience with conversations. How many times have you had a conversation with somebody who could not perform one of the three steps well? For example, have you ever had a conversation in which the other party did not listen to what you were saying? Perhaps this person could think very well, and was quite articulate in expressing his reactions, but if he didn't listen to what you were saying, was the conversation not a waste of time? And how many times have you had a conversation with a person who listened well, but just couldn't think well—in other words, a dummy? Don't you find conversations with dolts to be a waste of your time? Or how about the conversations with people who just can't express themselves? They stammer and struggle to articulate their ideas, but they do such a poor job of it that the entire conversation isn't worth the effort.

Thus, in order to have a good conversation, both parties must be able to perform all three steps well. This rule can be generalized to all forms of interaction. Thus, if the computer is going to engage in something like a

conversation, then it must perform all three steps. It must listen to the user, think about what he has said, develop an interesting and entertaining reaction to the user's input, and then it must express that reaction back to the user. And it must perform all three steps well.

### **Is More Interactivity Better?**

Now I turn to a more difficult question: If interactivity is accorded primacy in entertainment software design, then is not more interactivity better than less interactivity? Put more baldly, is a more interactive product better than a less interactive product? I think so.

My first argument in favor of greater interactivity relies on the primacy of interactivity. Since interactivity is the basis of competitive advantage of the computer, it is only fit and proper that we should emphasize it. The more we emphasize interactivity in our designs, the more fully we utilize the true strength of the computer as an artistic medium.

Consider the cinema by way of example. Very roughly speaking, and all other factors being equal, the more cinematic a movie is, the better the movie. We can argue long and hard over the precise definition of "cinematic," but surely we can agree that it involves the special capabilities of the camera to capture motion, to pan, zoom, and move, to cut between scenes, and so forth. Surely a movie that fails to use these techniques will be inferior to a movie that does, other factors being equal. Thus, we can say that a more cinematic movie is superior to a less cinematic movie. Of course, there are many exceptions to the general rule: a poorly executed more cinematic movie is not superior, and a movie whose artistic intent requires a less cinematic style could still be a superior work. There are other exceptions as well, but I think that the general rule holds true.

An additional argument arises from the fact that the interactive arts are still in their infancy. We have just scratched the surface of this medium; we do not fully understand interactivity. Our ignorance is reflected in the body of work we have created. In the vast universe of potential computer games, the set of actual games that we have created lies crowded down in a small corner, huddled together in common low interactivity. Taking all of our computer games and entertainments as a group, the amount of interactivity that we offer is a faint and clumsy whisper of what should be possible.

Perhaps a mathematical approach might better illuminate my point. Imagine a scale of interactivity, running from 0 to 100 units of interactivity, with 100 representing interactivity so intense that it lies beyond human comprehension. Imagine all the games in the universe placed on this scale. Now, in an ideal world, these games would yield a bell curve, with a few high-interactivity games, a few low-interactivity games, and a great many games in the middle of the bell curve. But we have not attained this ideal bell curve. Our ignorance denies us the middle and upper portions of the bell curve, constraining us to the lower end of the curve. All of our work lies crowded down there.

This is my second reason why more interactivity is better than less interactivity. We need to move up that bell curve, to explore areas of interactivity that previously have been inaccessible. The low end of the scale is already heavily populated with designs; the middle and upper reaches of the scale are empty. This is why I hold a more interactive game in higher esteem than a less interactive one. This is why I honor a designer who goes where no designer has gone before more than one who hews to more familiar territory.

Of course, when we have populated the bell curve more fully, the day will come when some genius creates a game that is too interactive. But that day is far distant.

## How Do We Measure Interactivity?

What, precisely, do we mean when we talk about “high” interactivity and “low” interactivity? Is an intense, fast-paced action game more interactive than a complex, slow-moving strategy game? I don’t think so. We need merely hearken back to my standard example of interactivity: a conversation. What makes a conversation more or less interactive? Certainly it’s not the speed with which the interlocutors speak. To achieve high conversational interactivity, each person must perform high-quality listening, high-quality thinking, and high-quality speaking. For an intense conversation, you must *really* listen to the other person, really hear what they’re saying. You must think carefully about what they said, gauging their true meaning. And finally, you must choose your words carefully to state your thoughts with perfect clarity. When all these goals are met smoothly, the conversation is intense.

Thus, a highly interactive game would listen very well to its player, think well about the player’s inputs, and produce clear, expressive outputs. We have done a good job with the third step in this process, but what about the first two? Do you really believe that players can say everything that they might want to say during a game? Does the input structure permit them to make the full range of reasonable moves? And does the thinking structure of the game permit the computer to challenge the player as an equal?

So there is our way of measuring interactivity. You can estimate the interactivity level of any game by asking three questions:

- How much of what the player might desire to say does the game permit the player to actually say?
- How well does the game think about the player’s inputs?
- How well does the game express its reactions?

Let's apply this metric to some games. We'll start with the generic fast-paced action game. This game permits the player to say only a limited range of words: move up, down, left, right, and so forth. It does permit the player to say them quickly. Its processing of the player's input is rather simplistic: All it does is move the player around on a map. Its expressiveness is quite complete, within the limited purview of the game. We can conclude that such a game supports a small amount of interaction at a very fast pace; the end result is certainly highly interactive, but the bulk of its interactivity is derived from its speed rather than the brilliance of its design.

So let's jump to the other end of the scale with a ponderous strategy game like **Civilization**. The pace is much slower, but the player has a much wider range of things to say, and the game executes a much more complex set of algorithms. In other words, the listening and thinking are much deeper in this game. Of course, the expressiveness of the game is certainly up to snuff. We conclude that **Civilization** is also a highly interactive game, even though it operates at a slower pace. The slow pace of the game is compensated for by the richer listening and thinking.

Thus, you can achieve a high interactivity game in a variety of ways. A fast pace can certainly help, although it is no guarantee of high interactivity. The important thing is to keep the players active. They can either do a few things rapidly or a great many things slowly—but they must be mentally active at every step.

### Low-Interactivity Entertainment Designs

Let me now turn to low-interactivity products. They have a dismal history. Low-interactivity entertainment is not a new idea. In the first big boom of computer games, from 1981 to 1984, a number of low-interactivity games were attempted. One of these was **Alien Gardens**, published by Epyx. You were a kind of alien bee flitting through a garden of alien flowers, trying to pollenate them. It was a very low-key game, definitely low in

the interactivity department. All you could do was buzz around and, every now and then, try to touch a flower, which might kill you or reward you. Unfortunately, the difference between flowers was arbitrary. The game made no sense and ranks as one of the great turkeys of computer game history.

1985 saw another low-interactivity product: **Little Computer People**, from Activision. This odd product created a small family on your screen, moving around their dollhouse in the course of their daily activities. You the player watched them. The product attracted much attention from the press, but it was not, I believe, much of a commercial success, largely because the player didn't do very much. Much later, Will Wright came up with a much better implementation of the concept with **The Sims**. **The Sims** offered the player more interactivity than **Little Computer People**, and was accordingly a great success.

Epyx roared back in 1988 with more low-interactivity products: its line of VCR games, released with much hype and excitement. Realizing the clumsiness inherent in the serial format of a videotape, the designers rightly limited interaction to the bare minimum, focusing most of their attention on providing interesting footage for the player, who would occasionally fast forward or rewind. Here was the ultimate couch potato game. You didn't need a computer to play it and you didn't have to do much work. All you did was sit back and watch the tape and occasionally push a button on your remote control. Sounds great, right? It sounded great to a number of publishers, who frantically put together their own VCR products. Yet, despite some expensive marketing campaigns, VCR games bombed. They were a total disaster. Mindscape shipped one product and cancelled the second one, even though it was ready to ship, because the first game had failed so completely.

Another experiment in this direction was the CinemaWare line of games. These games were strong on spectacle and weak on interaction. The marketing thrust of the CinemaWare line was that these games were just

like movies, except that you could play with them. Most of the design effort was put into making lots of pretty pictures and animated sequences. The gameplay itself was weak. The first line in the series, **Defender of the Crown**, by Kellyn Beeck, created quite a sensation and sold very well. But after that, it seemed to be all downhill. CinemaWare went bust a few years later.

Another good example is the experience of Cyan, a game developer. Cyan's first game, **The Manhole**, was a low-interactivity adventure game for children. It sold enough copies to keep Cyan alive, but little more. Then they came up with **Cosmic Osmo**, sporting the same low interactivity but better cosmetics. Again, they made enough money to stay alive, but not much more. The big hit came with **Myst**, another adventure game. This time, however, the interactivity was more involved and the cosmetics were sensational. **Myst** sold like hotcakes, Cyan got filthy rich, and all seemed bright. Then they released a sequel to **Myst**, entitled **Riven**. It sold moderately well. A third game in the series, **Exile**, also did reasonably well, but again did not approach **Myst** in sales. Cyan continued in business by developing and licensing the **Myst** brand.

The same pattern shows up over and over. Laserdisc games made a huge sensation, but then faded away almost as quickly as they burst upon the scene. They were low-interactivity games. **The 7<sup>th</sup> Guest**, by Rob Landeros and Graeme Devine, was a huge hit in the early 1990s, sporting low-interactivity puzzles mated with glorious animations. Their sequel, **The 11<sup>th</sup> Hour**, sold reasonably well, but not sensationally well. And there were no further games in that series. Cliff Johnson made a minor splash with a brilliant collection of puzzles, followed it up with a sequel that didn't sell well, and then disappeared.

Low-interactivity games sound like a great idea, such a great idea that people keep going back and doing them over and over. And, in pure Darwinian fashion, the companies that have cast their lot with low-interactivity games have suffered extinction. Epyx, CinemaWare, and

Mindscape were all reduced to ashes; only Cyan broke the curse. But the survivors seem unable to learn from their competitors' failures; low-interactivity games keep popping up like some time-hopping Sisyphusian dodo bird bent on repeating its extinction in as many eras as possible.

Why have low-interactivity games been such a dismal failure? One would think that there should be some small fragment of a market for them. Why is the historical experience so decisively negative in defiance of common sense?

There are two answers, I think. The first is that the available hardware is not up to the task. We have not yet hit the right combination of ingredients to build good low-interactivity games. The VCR gives lots of imagery, but its access times are so slow that even low-interactivity games suffer. The computer itself simply cannot generate or maintain images of enough variety and quality to entertain the player by themselves. The DVD should solve both problems. It offers faster access times than videotape, yet much greater image capacity than the computer. Whether the combination will be fast enough and visually rich enough to overcome the inherent weakness of low interactivity, I cannot say.

The second answer is more pessimistic. I have long maintained that interactivity is the essence of the gaming experience, and that the quality of the interaction determines the quality of the game. If this be true, then the very notion of low-interactivity games is intrinsically wrong-headed, and such products will inevitably fail.

### An Interesting Exception

There is a small group of low-interactivity games that have been somewhat successful. These are the games produced by Cyan (**Manhole** and **Cosmic Osmo**) and Amanda Goodenough (**Inigo Gets Out, Your Faithful Camel**, et al), and a number of products from Broderbund. They are low-interactivity games, really more like vaguely linear stories with some

buttons to press. They have been moderately successful. What is striking is that all of these products are designed for young children. It appears that our industry's Darwinian methods have at last found a suitable habitat for this otherwise less-than-fittest species of game.

Why is it that low-interactivity products are successful with young children when they don't seem to work with older players? I think that the answer can be found by asking another: Why don't high-interactivity products work with young children? Try foisting **SimCity**, **Robotron**, or **Half-Life** on a five-year-old, if you're willing to risk accusations of child abuse. The poor kid will be overwhelmed by such games. He just doesn't have the perspicacity to handle such a game. What's left for him but the low-interactivity games?

### Workload Versus Payoff

Some thinkers have observed that many of the highly interactive games impose a large workload on the player. To master a fast-paced action game, you've got to practice, practice, practice. To make sense of a big strategy game like **Civilization**, you've got to study a heavy manual. Lower interactivity games, they note, impose less work on the player. Taking full advantage of this property of low-interactivity games should yield viable products—or so they argue.

To evaluate this line of reasoning, I suggest that we start with high-interactivity games and then move toward lower interactivity. What do we gain and lose as we move in this direction? As we lose interactivity, we reduce the total quantity of decision-making that the player must perform. This reduces his workload. It also reduces his ability to creatively influence the outcome of the game. In other words, as we reduce the interactivity of the game (its gameplay), the player's degree of participation in the outcome is diminished and he therefore becomes less involved in the outcome, which becomes more predetermined.

But there's a catch: The player's workload is not proportional to the quantity of decision-making. Decision-making consists of two parts: a laborious process of learning the basic parameters for making the decision, and a faster process of applying those parameters. The player must go through the first process whether he makes one decision or a hundred. Thus, his workload is equal to a fixed quantity (learning the rules) plus a variable quantity (playing the game).

An example might help here. Suppose I present you with two games. The first is a truly minimal-interactivity game. You will be asked to make one decision during the entire game. It is a murder mystery game, the climax of which places you in a room with the six main suspects and a gun. You must decide whom to shoot. That's exactly one decision—about as little interactivity as you can get. Yet you will likely ask a great many questions before making your decision. Can I shoot more than once? Can somebody else shoot me? May I choose not to shoot anybody? (True gamesters will note that the problems are trivially solved by playing the game several times, experimenting with each of these options in turn. While entirely possible, this flies in the face of the stated intent of the low-interactivity game.) Note that these questions are really questions about the rules of the game. You will have a considerable workload just learning the context for your single decision, and inasmuch as the outcome of the game rides on your single decision, you had damn well better learn the rules thoroughly.

The second game is a more conventional game with many decisions. Once again you will have the workload of learning the rules of the game, and in addition to that you will have the workload of making all those decisions. Yet the workload of learning the rules is most likely the more substantial of the two. In other words, if you end up making a hundred decisions during the course of this game, your total workload will not be 100 times greater than your workload with the minimal-interactivity game. It might not even be twice as great.

Thus, as we move from the higher-interactivity game to the minimal-interactivity game, two factors are reduced: the player's workload and his ability to influence the outcome of the game. But—and this is the key point—the latter falls faster than the former. Reducing interactivity gains us only small benefits in terms of reducing workload but costs us heavily in terms of the player's ability to creatively influence the outcome of the game.

This, I think, is the real reason why low-interactivity games have been such failures. Diminishing the interactivity just makes the game less fun faster than it makes the game easier. What we gain in terms of reduced workload we more than make up for in terms of diminished fun.

### Weird Ideas

Lastly, there are the blue-sky concepts for low-interactivity games. Most of these center on some form of storytelling. In one approach, the computer tells the player a story, with the player somehow providing cues that the computer uses to adjust the story to suit the player's interests. For example, if the computer mentions an encounter with a beautiful girl, and the player so indicates, the computer could proceed to describe a sexual liaison. If the player is female, it might tell of a friendship developing between the two.

The problem with this lies in the nature of the cues provided by the players. Exactly how do the players communicate their desires to the computer? If we use a series of predetermined branchpoints (a *branchpoint* is a point reached during the play of a game, at which the player must make a decision that will take him down one of several paths), the game has reverted to a conventional adventure game, and the players still must learn the language of expression for the adventure. Proponents of such schemes often fall back on deliberately vague formulations. The computer will "sense" the player's mood, they claim. I find it difficult to imagine just how this sensing will take place, and how the computer will interpret whatever it senses.

A variation on this scheme makes reference to the manner in which performing artists sense the mood of their audiences and adjust their performance accordingly. This, it is asserted, constitutes an advanced form of low interactivity that could be harnessed for new types of games. The problem lies in the input and processing required to accomplish this. The performing artist is analyzing fine shades of voice intonation and subtle nuance of facial expressions. This type of processing is way beyond anything we can process on a personal computer, even assuming that we could equip our games with microphones and television cameras to provide the input.

There is a second and more powerful argument against such schemes. Even if we could implement them, they would still be inappropriate. The performing artist who makes adjustments in response to the audience's feedback does so on a very gross average of the audience feedback. Some people will be screaming "Faster!" as others are yelling "Slower!" The artist can't do both, and so responds to the majority. What's more important is the fact that the audience understands this. We can't all have our way, so we accept the situation. But when I am the only user of a computer game, I am completely justified in expecting that I can have my way. I expect the computer to respond to my wishes. If the computer fails to understand my wishes or is incapable of executing my desires, I will be dissatisfied. So if I grunt or laugh or scowl or drum my fingers and the computer fails to get my message, the product will have failed.

### **Conclusions on Low-Interactivity Designs**

The concept of low-interactivity entertainment is a ghost that we will never exorcise from this industry. The concept just keeps popping up like an annual flu bug. Some naïve fool will come forward with "this great new idea that nobody has ever thought of before." As I discussed, the concept seems sound on first examination, so people will probably give it credence. Who knows, some credulous publisher might be persuaded to part with development dollars to explore the idea.

## Process Intensity Versus Data Intensity

Closely tied to the concept of interactivity is a concept that I describe with the phrase *process intensity versus data intensity*. *Process intensity* is the degree to which a program emphasizes processes instead of data. All programs use a mix of process and data. Process is reflected in algorithms, equations, and branches. Data is reflected in data tables, images, sounds, and text. A process-intensive program spends most of its time crunching numbers; a data-intensive program spends most of its time moving bytes around.

The difference between data and process constitutes a central construct around which the universe is built, and it shows up in every field of human intellectual inquiry. In language, it shows up as nouns and verbs. In economics, it's goods and services. In physics, it's particles and waves. In military science, it's assets and operations. And in computer science, it's bits and cycles. Process is abstract where data is tangible. Data is direct, where process is indirect. The difference between data and process is the difference between numbers and equations, between facts and principles, between events and forces, between knowledge and ideas.

Processing data is the very essence of what a computer does. There are many technologies that can store data: magnetic tape, punched cards, punched tape, paper and ink, microfilm, microfiche, and optical disk, to name just a few. But there is only one technology that can process data: the computer. This is its single source of superiority over the other technologies. Using the computer in a data-intensive mode wastes its greatest strength.

Because process intensity is so close to the essence of "computeriness," it provides us with a useful criterion for evaluating the value of any piece of software. That criterion is a vague quantification of the desirability of process intensity. It uses the ratio of operations per datum, which I call the *crunch per bit ratio*. I intend here that an operation is any process

applied to a datum, such as an arithmetic operation, logical operation, or a simple Boolean inclusion or exclusion. A datum in this scheme can be a bit, a byte, a character, or a floating-point number—it is a small piece of information.

The “process intensity principle” is grand in implications and global in sweep. Like any such all-encompassing notion, it is subject to a variety of minor-league objections and compromising truths.

### **Objection 1: Substitutability**

Experienced programmers know that data can often be substituted for process. Many algorithms can be replaced by tables of data. This is a common trick for expending RAM to speed up processing. Because of this, many programmers see process and data as interchangeable.

This misconception arises from applying low-level considerations to the higher levels of software design. Sure, you can cook up a table of sine values with little trouble—but can you imagine a table specifying every possible behavioral result in a complex game such as

**Balance of Power?**

### **Objection 2: Greater Data Capacity**

A more serious challenge comes from the evolution of personal computing technology. In twenty years, we have moved from an 8-bit 6502 running at 1MHz to 64-bit CPUs running at 1GHz. This represents about a 10,000-fold increase in processing power. At the same time, though, RAM sizes have increased from a typical 4 kilobytes of RAM to perhaps 256 megabytes of RAM—a 64,000-fold increase. Mass storage has increased from cassettes holding, say, 4 kilobytes, to hard disks holding 20 gigabytes—a 5,000,000-fold increase. Thus, data storage capacity is increasing faster than processing capacity. Under these circumstances, we would be foolish *not* to shift some of our emphasis to data intensity. But this consideration, while perfectly valid, is secondary in nature; it is a matter of adjustment rather than fundamental stance.

### Objection 3: "Balance"

Then there is the argument that process and data are both necessary to good computing. Proponents of this school note that an algorithm without data to crunch is useless; they therefore claim that a good program establishes a balance between process and data. While the argument is fundamentally sound, it does not suggest anything about the proper mix between process and data. It merely establishes that some small amount of data is necessary. It does not in any way suggest that data deserves emphasis equal to that accorded to process.

The importance of process intensity does not mean that data has no intrinsic value. Data endows a game with useful color and texture. An excessively process-intensive game will be so devoid of data that it will take on an almost mathematical feel. Consider, for example, this sequence of games: **checkers-chess-Diplomacy-Balance of Power**. As we move along this sequence, the amount of data about the world integrated into the game increases. **Checkers** is very pure, very clean; **chess** adds a little more data in the different capabilities of the pieces. **Diplomacy** brings in more data about the nature of military power and the geographical relationships in Europe. **Balance of Power** throws in a mountain of data about the world. Even though the absolute amount of data increases, the crunch per bit ratio remains high (perhaps it even increases) across this sequence. My point here is that data is not intrinsically evil; the amount of data can be increased if the amount of process is concomitantly raised.

### A Hidden Reason: Difficulty of Abstraction

The most powerful resistance to process intensity, though, is unstated. It is a mental laziness that afflicts all of us. Process intensity is so very hard to implement. Data intensity is easy to put into a program. Just get that artwork into a file and read it onto the screen; store that sound

Eschew  
data-  
intensive  
designs;  
aspire to  
process-  
intensity.

>

effect on the disk and pump it out to the speaker. There's instant gratification in these data-intensive approaches. It looks and sounds great immediately. Process intensity requires all those hours mucking around with equations. Because it's so indirect, you're never certain how it will behave. The results always look so primitive next to the data-intensive stuff. So we follow the path of least resistance right down to data intensity.

Process intensity is a powerful theoretical concept for designing all kinds of software, not just games. It is highly theoretical, and so it is difficult to understand and implement, and there are numerous exceptions and compromising considerations that arise when applying the notion. Nevertheless, it remains a useful theoretical tool in game design.