Social Science Computer Review

http://ssc.sagepub.com

Creative Uses of Software Errors: Glitches and Cheats

Wilma Alice Bainbridge and William Sims Bainbridge

Social Science Computer Review 2007; 25; 61
DOI: 10.1177/0894439306289510

The online version of this article can be found at:
http://ssc.sagepub.com/cgi/content/abstract/25/1/61

Published by:
SAGE
http://www.sagepublications.com

Social Science Computer Review
Volume 25 Number 1
Spring 2007 61-77

10.1177/0894439306289510
http://ssc.sagepub.com
hosted at
http://online.sagepub.com

Creative Uses of Software Errors

Glitches and Cheats

Wilma Alice Bainbridge
Yale University

William Sims Bainbridge
National Science Foundation

Video games constitute a major sector of computing with distinctive social implications. Analysis of video game programming errors, design limitations, and rule ambiguities suggests a range of positive functions that glitches and cheats may perform. Glitches are software errors, either programming bugs or design flaws. Cheats are tricks for mastering games by circumventing the official rules for play. This article is based on the study of 751 glitches in 155 popular video games plus examination of web sites that disseminate cheats. It compares the standard computer science response to errors, which is to eliminate them, with a common video gamer response, which is to exploit them. The theoretical analysis contrasts games (in which victory is achieving set goals by following given rules) with metagames (in which victory is having the power to define the nature and rules of the game). Several related theoretical concepts are used to describe the complex subculture of video game players, for whom glitches and cheats are socially significant.

Keywords: *computer; error; video game; glitch; cheat*

The video game industry is comparable in size to the motion picture industry, and it has begun to attract considerable scholarly interest ("Chasing the Dream," 2005; El Rhalibi, 2004). Given the complex nature of the gaming subculture, this is an appropriate topic for social-scientific research that may offer insights applicable to other forms of consumer technology or to computing in general. Video game systems are really specialized computers, running software that was programmed often in the same languages (e.g., C++) used for conventional computers, possessing considerable memory and processor speed, and, in their latest incarnations, capable of accessing Internet. The manufacturer of the best-selling console, the PlayStation 2, is the Sony Computer Entertainment Corporation. The console in second sales position, the Xbox, is manufactured by Microsoft, the dominant personal computer software company. Particular games are often available both for video game systems and for personal computers, and a subculture of hackers has developed emulator software to allow games written for the systems to be played on PCs.

Social scientists have long recognized that games can reveal much about fundamental laws of human behavior (Huizinga, 1939/1949; Petersen, 1994; von Neumann & Morgenstern,

Authors' Note: The views expressed in this article do not necessarily represent the views of the National Science Foundation or the United States. Abundant thanks are owed to Constance May Bainbridge, who discovered many glitches and did the artwork for the web site.

1944), and we wish to suggest that the glitches and cheats associated with video games have close analogies across the full range of kinds of social behavior. Glitches are software errors, either programming bugs or design flaws. Cheats are tricks for mastering games by circumventing the official rules for play and they occasionally exploit glitches. On a higher level of abstraction, glitches and cheats are similar because they violate the usual assumptions of novice players and often violate the intentions of the game designers as well. Although computer scientists consider bugs to be bad, they can provide users with new, unexpected, and sometimes very attractive alternatives. In a sense, conventional computer software, such as spreadsheets and database programs, are systems of bureaucratic rules. Thus, errors and naive design assumptions can be cracks in the bureaucratic structure through which human beings may seek freedom.

Research Method

Our research employed a somewhat novel approach, examining the games themselves and the Internet-based subculture oriented toward glitches and cheats. In one sense, we were doing technology assessment, or we were doing contemporary archaeology in deducing social realities from the artifacts created and used by human beings. The methods were often those of ethnography but carried out electronically.

All complex electronic games are likely to contain glitches. Manufacturers invest most of their bug-finding efforts on fixing problems that might cause the software to crash rather than on more benign errors, and they often ship games that still contain known errors because the effort and delay to fix them all would be too costly (Schultz, Bryant, & Langdell, 2005). On average, games programmed for personal computers probably have more glitches than those programmed for the standard video game consoles (Bethke, 2003). Console manufacturers exercise more quality control over the games produced for their systems. Several variants of PC operating systems exist, and graphics display cards thus present extra challenges for programmers, and the fact that software updates can be distributed over the Internet probably encourages manufacturers to ship PC games earlier in the development process.

To gain a broad perspective, we experimented with several different systems: the currently most popular console (PlayStation 2), three different generations of the oldest still-popular console (Super Nintendo, Nintendo 64, and GameCube), two portable systems (Game Boy and Nintendo DS), and Windows-based personal computers including desktop and pocket models. Our equipment also included a video cassette recorder and a video-capture device so we could record episodes of actual game play and port them into a computer. After we had discovered a number of glitches, we launched a web site called The Center for Glitch Studies (which incidentally won an accessibility award) to communicate effectively with the glitch-oriented subculture. There were many web sites dedicated to cheats but few for glitches, and those tended not to be updated. Soon, members of the glitch subculture began sending us information about glitches they had discovered, and whenever possible, we replicated them on our own equipment. By September 14, 2005, our web site offered information about 751 glitches in 155 popular video games, with instructions on how to make each one happen and often a screenshot picture of what it looked like.

As is common in sociological ethnography, we deferred theorizing until we had made a substantial number of observations (Babbie, 2004) but then began categorizing glitches in terms both of the programming or design flaw that caused them and their apparent human consequences. As participant observers, we learned from our own experience and from reading the experiences of the people who sent us information or who posted information on their own web sites. After describing representative glitches and cheats here, we will offer a theoretical analysis that integrates our own observations with some of the more thoughtful relevant literature. Before we can properly discuss innovative conceptions of programming error, we need to document the standard conception that views errors as negatives to be avoided, prevented, and fixed.

Conventional Conceptualizations of Error

To gain a broader perspective on the kinds of error and associated responses found throughout computer science, we did a modest content analysis of information about grants available on the web site of the National Science Foundation (NSF; `www.nsf.gov`). We restricted the search to grants made by the Directorate for Computer and Information Science and Engineering and found fully 940 whose abstracts or titles contained the word *error*. Some of these are collaborative or renewal grants that duplicate the same text. Thus, 103 titles contained the word *error*, of which 85 were distinct. Fully 1,418 sentences in the abstracts contained *error*, including 1,316 different sentences. A sense of the researchers' interest in finding and fixing the errors is revealed by the fact that 133 of these sentences included the word *detect* (as in error detection) and 214 contained *correct* (as in error correction).

Among the most widely studied kinds of errors affecting computers are those that concern corruption of data during transmission, which is the fundamental problem addressed by Shannon's (1948) pioneering paper in information theory. File corruption errors that take place during data storage, perhaps caused by unreliable hardware, are often dealt with by some of the same methods as transmission errors, such as parity checks. Relevant sentences found by the content analysis include: "Error-correcting codes are an essential part of modern communication and storage systems and much of today's technology would not be possible without them," "A small improvement in error-correction can yield large savings in the overall cost of systems," and "Wireless networks provide treacherous environments for digital communication; they are bandwidth-limited and error-prone."

Sentences extracted from the NSF grant database reveal considerable concern about programming errors: "Programming is notoriously error-prone," "Traditional programming language design techniques are very slow and error prone," "Programmers try to eliminate as many errors as possible by reasoning about the program's behavior," "Unfortunately, however, errors are pervasive in end-user software, and the resulting impact is sometimes enormous," "Software is often laced with coding errors which result in loss of life and productivity," and "A single operating system error can crash a machine." But some sentences also provide motivation for our research: "If 'to err is human,' then understanding the origin, nature, detection, and correction of errors should be an important goal for those engaged in the study of human-computer interaction."

Errors can be classified in many ways, such as in terms of when an error might be automatically detected. Compile-time errors are misstatements in source code that are detected by the compiler program that translates the higher-level language in which the program was written into machine code that can be read by the computer or, in the case of Java, by the Java virtual machine. If standard routines are linked from a library into the program separately from compilation, then there can be link-time errors. And, there are also run-time errors that interfere with the running of the program and may be dealt with by error handling routines that may either be built into the programming environment or added by the programmer.

Errors in the design and programming of software are of vast importance, yet not many NSF-supported research projects address them every year simply because new approaches are hard to invent. Numerous methods have been developed to reduce programming errors, including modular programming to allow reuse of well-tested parts of programs, development of software design as a specialty separate from programming of code, formal declaration and strong typing of variables, and mathematical or logical analysis of algorithms or of the large-scale structure of the program.

Despite a half century of research and development, automatic methods have not replaced human testing. Within a software team, this can imply complex divisions of labor, as one NSF abstract notes: "A design assistant will assist a programmer by detecting errors and inconsistencies in his design choices and by automatically making many straightforward implementation decisions." Outside the team, this usually implies beta testing, in which real users try out programs before they are officially released and provide feedback that can improve the final product. Gina Neff and David Stark (2004) have coined the term *permanently beta* to describe the current situation in which the pressures to bring products to the market are so great that software is released long before all the bugs have been found and then constantly updated as users complain about its specific shortcomings.

Glitches

Undoubtedly, the most common video game programming error is a discrepancy between the software's display model and its world model, so common that we included only the most striking examples in our inventory. The display model is the image of the game's environment that appears on the computer or TV screen. The world model is the representation of that environment revealed by the behavior of characters and objects inside the game. The discrepancy is typically noticed when characters and objects in the game collide with each other or with the walls and floor of their environment, and collision detection is one of the more challenging problems for the game programmer (Dalmau, 2003).

For example, Castlevania: The Lament of Innocence, a PlayStation 2 game, depicts a castle with more than 100 rooms through which a knight must battle against monsters and vampires. In the display model, the walls are extremely ornate, decorated with windows, statuary, carvings, tapestries, and fixtures. The world model, in contrast, depicts the walls as flat surfaces that prevent the knight from nearing the plane of the display wall and from touching most of the objects depicted on it. At a few locations, however, the display model reaches closer to the knight than the world model, and he can pass through displayed objects without being harmed: a sword protruding from a statue, a flaming lamp, and a post supporting a staircase.

The world model and display model of contemporary three-dimensional games are created by very different routines in the program, following very different principles, and each makes potentially heavy demands on the speed of the central processing chip of the machine. To simplify the task in writing the program and to reduce the number of calculations the machine must make, characters and free-standing objects are often represented by spheres, cylinders, or cubes or (what amounts to the same thing) by points in space plus a distance within which another object cannot approach. Walls and floors are generally represented as planes. Some of the most ambitious games for the fastest machines follow the standard approach in computer graphics, depicting complex surfaces as a mesh of polygons. The programmers are quite aware of most of these discrepancies between the two models, so they might perhaps be described as design limitations rather than programming bugs, but under either name they are errors.

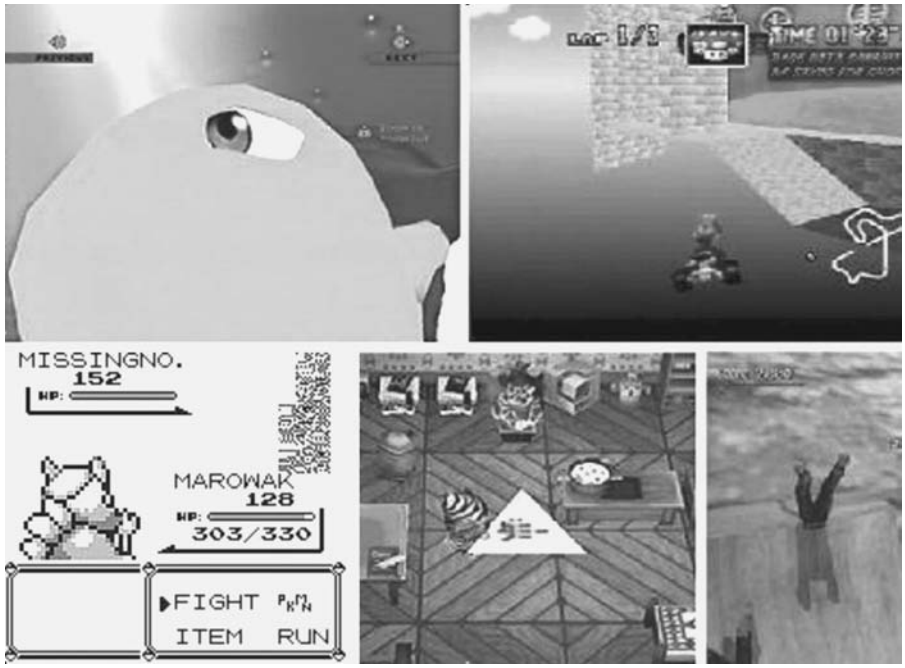
The poet Coleridge (1798/1907) called attention to the willing suspension of disbelief on which art relies. Video games are among the newest art forms, so the concept surely applies to them. Visual discrepancies on the display screen erode that suspension of disbelief, and the inability of the knight to reach the walls of the castle or interact with many of the objects causes at least mild frustration. However, there is a slight magical quality when he leaps through an object, and some players find this entertaining. Thus, one of the many ways in which a player may enjoy a video game is to explore it in search of these discrepancies, relishing the variety of effects they can cause.

Some glitches involve harmless gaps or distortions in the display model of objects. If Mario puts on a tropical shirt and sunglasses, in Super Mario Sunshine, then looks in a mirror, his reflection will have the tropical shirt on, but not the sunglasses. In one region of Super Mario RPG, it is possible to get nonplaying characters to overlap, like Siamese twins, for example two of the Yoshi dinosaurs that Mario rides. In Super Smash Brothers Melee, looking at Daisy from a particular angle reveals a third eye on the back of her head, as shown in Figure 1. In Rocket: Robot On Wheels, the robot main character can grab an object in a tractor beam, which acts like a short piece of rope, pulling the object along. The robot can sling the object on the top surface of a dock, while going under the dock, producing an effect like a magic trick in which the object appears to move itself. This works because the program avoids collision for the robot and the object but not the beam connecting them.

More exciting are cases in which walls that should be in the world model are missing, their absence hidden by the fact that they exist in the display model. By extreme measures, perhaps unanticipated by the programmers, it is possible to reach the top roof of the castle in Super Mario 64, but the roof is not solid and Mario falls down between the walls. Sonic Adventures 2 (adapted from the earlier Sega Dreamcast system to the GameCube) has a spot where it is possible to jump through the sky, transforming the character into a double image and propelling the character's "chao" companions below ground level. Mario Kart 64 allows a go-cart driver, such as Yoshi the friendly dinosaur, to crash over a barrier and wind up outside the world in a featureless void with a pink sky (see Figure 1).

Some glitches involve programmers' failure to clean up after themselves. In Animal Crossing, we discovered that playing a game in an igloo occasionally gains you a useless white triangle with *dummy* written on it in Japanese (see Figure 1). This is a tester item that the programmers forgot to remove from the game (cf. O, 2005a). In the original Pokemon games on the Game Boy, a test pokemon was accidentally left in called "Missigno," which

Figure 1
Five Glitches (Daisy's Eye, Yoshi, Missingno, Dummy Triangle, Tony Hawk)



spawned one of the most popular glitches ever in game history (see Figure 1). Depending on what is done, encounters with this character can result in an almost infinite multiplication of items, permanent destruction of the game, and having pokemon with a higher-than-allowed level.

Among the most exciting glitches is finding one's way into an unfinished part of the game world that was supposed to be sealed off or into an area that is an accidental glitched copy of one of the normal areas. In The Legend of Zelda: Link's Awakening for monochrome Game Boy, opening the map while on a boundary between two areas can move the character through the boundary and, if done in a particular place, will take him to a crazy glitch world with multiple rooms, strange enemies, and invisible walls. A sunken ship in The Legend of Zelda: The Wind Waker cannot be reached by ordinary means, but we discovered a route into it, although there is no escape. The extra ship parts were most likely there for display purposes. It just so happened that to design a realistic three-dimensional ship background that combined with the foreground, parts of it would be solid, but the game designers probably did not intend to make the area accessible. Another good example is the unfinished levels in Super Mario Bros 3 that exist but can be accessed only with hacking devices such as those described below. Some even include enemies that were never used in the finished game.

Glitches in multiplayer games may involve unusual interactions between players. Donkey Kong Country can be played as a cooperative game in which two players (represented by the characters Donkey and Diddy) take turns leading the duo though the challenges. It is possible

to transform the colors and behaviors of objects, at one point, by changing the lead character beside a barrel that shoots the characters into the air. In Super Mario Kart, two players can battle until one is killed, but just the right move can leave the dead loser able to drive his or her go-cart around and shoot weapons at the other, although the weapons do no damage. A better example of glitches caused by complex interaction between characters is in Super Smash Bros where an ice climber can grab another character and freeze it in the air. This happens because the ice climber has two characters controlled by one human, and having the minor character grab another human character freezes it in the air, presumably because game programmers focused on actions by the major character.

Some glitches can halt a game without completely crashing it, whereas others lead to complete corruption of the data. In the skateboarding game Tony Hawk Pro Skater 3, an ollie jump into the correct hillside will leave the character's head irretrievably stuck in the ground, with the legs helplessly thrashing in the air (see Figure 1). In Yoshi's Island, under certain conditions, the game will display a message to the player who can carefully aim Yoshi and a large egg simultaneously at the message block, unpredictably causing major corruption of the data, such as rendering enemies invisible or even crashing the game. We cannot be absolutely sure how such an error happens, without either decompiling the game and studying the code laboriously or getting inside information from the manufacturer, but this appears to be comparable to an out-of-range error in ordinary software. Bytes spill over from one variable to another and at the extreme can corrupt pointer variables that hold the addresses of other variables and cascade unpredictably until the software crashes. Link, the hero of The Legend of Zelda: Majora's Mask, can ride his horse to a particular location where a combination of equipping various items and saving data can either crash the game or produce a host of surreal effects, such as standing in thin air and being able to control the horse without being on it.

We will conclude this section with an initial attempt to categorize glitches in terms of their causes, acknowledging that really confident assignment of causes would be impractical because it would require disassembling the programming code and examining the data that define the objects and spaces in the game. We were able, however, to assign best-guess causes to 580 of the glitches. In addition, we assigned a score to each glitch assessing how exciting it is, a subjective rating of how much the glitch could satisfy a game player's motivation to find glitches if that were the object of play. Table 1 shows data on the basis of 10 categories of glitches.

The first two categories concern discrepancies between the display model and the world model, either a weak region showing minor incongruities or a hole in a boundary that lets a character or object escape the confines of the intended display world. Three categories concern failures of the programmer when writing general parts of the program. Bad code involves simple programming bugs that might be as small as a single line of code, whereas unchecked logical fallacy involves a failure to check the implications between different instructions or conditions. This arises when the player makes a reasonable move but the game fails to anticipate the implications, for example talking to a character without previously getting the item that character is programmed to respond to. Undeleted test code refers to cases where one or more lines of code were in the program to assist the programming process and were inadvertently left behind rather than being removed as intended.

We have separated out glitches having to do with improper behavior on the part of the player. Camera maneuvering means moving the point of view, something that many games

Table 1
Categorization of 580 Video Game Glitches by Cause

	Number of Cases	Mean Excitement
Model discrepancy		
Weak region	93	2.96
Hole in boundary	77	4.52
Programming		
Bad code	24	2.71
Unchecked logical fallacy	141	3.11
Undeleted test code	19	4.50
Unanticipated input		
Camera maneuvering	6	2.42
Unexpected button	110	3.34
Bizarre maneuver	77	4.77
Hardware glitch	25	4.36
Overload of information	8	2.88
Total	580	3.64

do not allow, until the display shows something that should not be seen. The unexpected pressing of buttons is a player input so simple that it should have been caught in the code and involves timing a normal action at the right time or holding a button as the player does something. Bizarre maneuvers require more intentional behavior on the part of the player, involving several steps and going beyond what the game designers expected the player to do. Hardware glitches involve mistreatment of the game system, such as partially removing a game cartridge, using a cheating device (such as the Action Replay), and switching the game off or resetting during an action. Cases of information overload seem to involve too many things happening at once.

We also attempted to characterize the immediate results of the glitches, and there were at least 5 cases each of 23 distinguishable effects. As we have noted, sometimes a character may be able to pass through a supposedly solid surface, gaining unintended access to some area, perhaps entering an incomplete region that was not supposed to be accessible or even departing the world depicted in the game altogether. As in the dummy example above, test objects may inadvertently have been left in the game. Purely visual glitches can change the appearance of the game, alter a computer character, and produce a false reflection or unusual camera angle. A few glitches cause strange sounds. More serious are when objects or characters become invisible yet remain solid, when two objects overlap or when they appear at two or more locations. Characters may perform bizarre actions, float in the air or walk on water, act alive after dying, or violate the laws of logic. At times, the player may gain unusual abilities, or a number such as a score may exceed its set maximum or minimum. At the extreme, a glitch might freeze the game or permanently damage the game play, problems that require the player to restart the game, possibly losing all progress. Depending on whether the player has extra lives saved up, premature death of the player's character can also require a restart.

The excitement scores in Table 1 illustrate the kind of psychological measurement that a subsequent study could do more rigorously, perhaps employing multiple judges and assessing

reliability. Although the scale runs from 0 to 10 and the actual range was from 0.5 to 9.0, the mean is only 3.64, implying that most glitches are bland, and of course we chose not even to count the vast number of very minor discrepancies in game display. Thus, means above 4 are striking for the bizarre maneuver, hole in boundary, and undeleted test code categories. The really interesting glitches offer the player a completely new aspect of game play, affect the game over the long term, or greatly alter multiple facets of the game (e.g., sound, colors, and layering of the sprites that constitute the image components of a scene).

Cheats

Literally millions of players employ cheats at one time or another, and a very complex cheat culture has arisen involving the programmers, game testers, game manufacturers, users, guidebook publishers, and game-related web masters. Here we can only point out a few general principles and highlight glitch-enabled cheats.

Among the most prized glitches are cheat glitches, those that help the player win the game. A famous example is Super Mario 64, a game widely praised by professional game developers but containing two glitches that if used in the right combination can allow one to win after collecting only 16 stars rather than the required 70 out of 120. First, after legitimately winning 15 stars, Mario can enter the basement of the central castle and break through a locked door that usually requires 30 stars to open by arranging to bounce off a rabbit that dwells there, and then the rabbit can be carried out of the basement to get through other locked doors. Doors in the game are not parts of the solid wall. Rather, when Mario gets close enough to a door, the game checks whether the door is locked and uses this information to choose between two movie sequences. In this case a door-locked sequence would be triggered, and Mario would be moved away from the door. However, if Mario can get past the door-checking movie sequence boundary, then he can get through the door even if it is locked. Winning a 16th star gives access to more of the castle, where Mario can do backward long jumps in a manner apparently unanticipated by the programmers to zoom up a staircase that was supposed to be endless until after one had earned 70 stars. Apparently, if Mario accumulates enough speed he can pop through any barriers, including walls and the endlessness barrier. Aggressive use of both glitches brings Mario to the final battle.

The knight of Castlevania: Lament of Innocence must amass a variety of consumable resources, including what gamers call hit points (or life, health) that are used up when the knight is injured and that cause death when they reach zero. During his exploration of the castle, the knight finds multiple copies of items that can be converted into hit points: potion, high potion, HP max up, diamond, small meat, and sushi. In the gamer jargon, these are sometimes called Easter eggs because they are hidden valuables that must be collected. Some Easter eggs are tools the knight keeps (e.g., magical rings, pieces of armor, and whips), but hit point resources are consumed, so the knight must constantly replenish them, often by fighting monsters, which risks the loss of hit points. However, soon after the game was published, a secret spread across Internet: There are a few very precise key locations in the game where the knight can increase the number of each of the items (if he has at least one) by standing at that point while the player presses a particular sequence of control buttons.

Some of the amateur web sites reporting this cheat called it a glitch, but it is certainly not a simple programming bug because it required considerable effort to create and thus could have been made only intentionally. Rather, it is most likely an aid for playtesters, the dozens or even hundreds of beta testers who try out early versions of the game and advise the programmers about bugs and features that should be improved (Fullerton, Swain, & Hoffman, 2004; Schultz et al., 2005). Playtesters need to be able to access all levels of the game conveniently, and one way to make this possible is to provide them with an abundance of resources so they can rush through parts of the game unworried about losing hit points. This Castlevania cheat is a glitch only if the programmers intended to remove the feature and forgot or if one can argue they should have removed it.

How do such cheats become known in the gaming community? One way is for individuals to discover them unaided, then pass the secret on to others. Early games, manufactured before the game cartridges or systems had nonvolatile memories, employed passwords to save the player's progress, useful also for playtesters. For example, The Wizard of Oz for the Super Nintendo is an extremely challenging game with many levels and more opportunities to lose life points than to gain them. After completing a level, the player is given a password, consisting of 24 apparently random letters of the alphabet. The player can write this code down and use it to start the game at the same point next session. With some effort, experimentation, and logic, we were able to crack this code, giving us the power to jump to any level with any resources we wished. For example, the following code gets Dorothy directly into the Emerald City with her companions, the Scarecrow, Tin Woodman, and Cowardly Lion: GCPQRZ-NSTVNX VBSDFG-DLMNPG. Arguably, the programmers should have employed an unbreakable code, and their failure to do so is a design flaw that might be called a glitch.

With many thousands of people playing popular games, able to gain a little fame by posting cheats on the Web, many cheats and glitches are discovered experimentally. Rumors persist that some are leaked by playtesters or programmers working for the manufacturers or by the manufacturers themselves. For at least 20 years, both official and unofficial guidebooks have been published for popular games, and the manufacturers found that exchange of secrets between players could increase sales. They began adding hidden features to games, often using secret codes of button presses, giving information about them to publishers and perhaps leaking the secrets through networks of avid fans. A recent example of a cheat disseminated through a second-party corporation concerns the PlayStation 2 game Destroy All Humans, in which the player takes the role of a telepathic alien invader in the 1950s, trying to destroy the FBI and US Army. The Cheat! program, on the G4 cable television channel that is entirely devoted to video games, revealed a code that increases the alien's parapsychological powers (`www.g4tv.com`).

Just as some glitches can be used as cheats, some cheats lead to very interesting glitches. Star Wars: Shadows of the Empire offers many examples. We will explain how to enter the debugging code for this game on the Nintendo 64 to illustrate how complex these codes can be: Enter your name as "_Wampa__Stompa_" and start the game. Now, push *Start* to pause the game once the level has fully begun. Hold all the *C* buttons, the *L* and *R* buttons, the *Z* button, and the left button on the control pad. Now, carefully tilt the control stick halfway to the left until you hear a sound, and then tilt it to the right until you hear a sound. Keep on tilting it back and forth, not pushing the stick all the way, until some letters appear with cheats you can use.

It takes about 5 tilts. To go through the cheats, use the control pad. To change a number or setting, push up on the control stick. To select a feature from the debug menu, push *A*.

One of these cheats puts the Star Wars enemies to sleep, and shooting them in this condition can make it possible to walk through them, removing them from the world model but not the display model. Another cheat reduces gravity, allowing the character to leap so high he or she passes through the ceiling or even out of the world because the programmers did not bother to put ceilings over many areas of the world model on the assumption that gravity would keep the character from jumping that high. A third cheat allows a rocket plane to fly through the barrier that defines its normal range, which is made of two-dimensional mountains, into a boxlike-area of clouds. If you fly out of the box, the images become distorted, smearing the plane across the sky.

The most extreme cheats employ hardware or software systems intended to hack directly into the game program's code or the console's data registers. A variety of systems circulate among gamers, with names like Game Genie, Game Shark, and Action Replay. These devices are primarily intended as cheats, most commonly giving the player's character unearned resources, as we did in cracking the code for The Wizard of Oz. But they can also open up new opportunities for exploration. The Legend of Zelda: The Wind Waker contains two very interesting hidden levels where programmers practiced working with actions and objects, that can be entered by means of Action Replay (see `perso.wanadoo.fr/spyromedia/nintendo/zelda.html`). Methods such as these can give the player access to the menu that programmers used to control all the variables while creating and debugging a game, even when the program was altered before sales to block ordinary access to it (O, 2005b).

In the summer of 2005, a remarkable case was reported across worldwide news media concerning the extremely popular game Grand Theft Auto: San Andreas ("GTA Sex Scandal," 2005; "Hidden Sex Scenes," 2005; "No More 'Hot Coffee,'" 2005; "Sex Controversy," 2005; "Uproar Grows," 2005). A Dutch member of the amateur game-modifying subculture, Patrick Wildenborg, distributed software that allowed a player to enter otherwise locked areas of the PC version of the game where the game characters could be made to engage in explicit sexual intercourse. One interpretation of the technical situation is that the game programmers had originally planned to include these areas but closed them off to avoid having the game classified for adults only, but there was also some confusion whether Wildenborg's program also altered the game by removing the characters' clothing. One debate centered on whether the sex scenes were really part of the game because they could not be accessed during ordinary play. Within weeks, the game had been reclassified for adults only, removed from many stores on three continents, and then repaired by a software patch distributed by the manufacturer. More recently, the manufacturers of the online game World of Warcraft added a spybot program that checks to see if the user is employing special hardware or software to cheat, thereby protecting the integrity of the game but raising very serious privacy issues for users ("Warcraft Game Maker," 2005).

Theoretical Analysis

The fundamental ambiguity of games is that the rules governing interaction are arbitrary, so one may ask who has the right to set the rules and define winning. This suggests the

Table 2
Categories of Orientations toward a Video Game

Dimensions	Concepts	Meaning in the Context of Video Games.
Goal orientation	Game	Victory is defined as achieving set goals by "legitimate" means.
	Metagame	Victory is having the power to define the nature of the game.
Mode of adaptation	Conformity	The player follows the rules in hope of winning.
	Ritualism	The player follows the rules without any hope of winning.
	Retreatism	The game defeats the player and languishes unplayed.
	Innovation	The player uses cheats to win the game.
	Rebellion	The player sets new goals and rules, such as glitch finding by playing.
Contest	Paidia	Play that often follows rules but does not involve winning and losing.
	Ludus	Game playing in which the key goal is defeating other players.
Motivation	Achievement	The desire to make progress and master the gaming environment.
	Exploration	The desire to explore, discover, and experience within the game.
	Socializing	The desire to interact congenially with other players.
	Conquest	The desire to vanquish other players.
Rules	Explicit	Rules are formally stated and agreed to.
	Implicit	Rules are embedded in what the software permits.
Competition	Synchronous	Multiple players compete in real time.
	Asynchronous	Multiple solo players compare accomplishments when not playing.
Linearity	Linear	There is only one way to win the game, one route to follow.
	Nonlinear	There are many paths to victory, maximizing player autonomy.

possibility of a metagame in which the goal is to redefine victory. Table 2 lists the distinction between game and metagame, along with others that will be introduced momentarily, on the logic that developing category schemes is a good first step toward developing formal theory (Stark & Bainbridge, 1979).

An excellent illustration is the very first video game Easter egg, a single gray pixel in the 1978 game Adventure, programmed by Warren Robinette for the Atari 2600. Although all of its games were programmed by single individuals, the Atari company did not want their names to be known and did not reward them for their pioneering efforts. Unbeknownst to his employers, Robinette programmed into this maze quest a hidden room bannering his name that could be entered only if the player picked up this gray dot and moved it to a particular location. A few avid fans found the gray dot and tried to figure out what it did, then after leaving the company Robinette shared his secret. Recently, he has analyzed this episode:

> For the players, the secret room was the meta-level, the way to truly beat the game and get to the real conclusion. For me, it was the meta-game I was playing with Atari management. They had the power to keep my name off the box, but I had the power to put it on the screen. (Robinette, 2003, p. xvii)

Thus, exploiting glitches and cheats can be moves in a metagame, in which the player rejects the rules that the video game manufacturer established and sets new criteria for winning. Sociologists would immediately recognize the parallel with Robert K. Merton's (1938) influential theory of anomie, that specifically draws the analogy between competition in life and in games. Merton notes that the norms of a culture are rules governing legitimate means

for achieving the goals that are enshrined in the society's values. One may then classify modes of adaptation or patterns of behavior in terms of their differential orientations toward norms and values (means and goals). Conformity is a pattern in which the individual follows legitimate norms in pursuing the established values. Two other patterns of little interest to Merton were ritualism (following the norms but not achieving the values) and retreatism (abandoning both norms and values).

Central to Merton's theory was innovation, the mode of adaptation in which an individual continues to seek the goals valued by the society but employs nonstandard means, thus violating societal norms. Like Merton himself, subsequent sociologists have tended to equate this concept of innovation with crime and delinquency and to assume that people would not do this unless they were frustrated in their attempts to achieve the values through conformity (Cloward & Ohlin, 1960). However, many other theories of norm breaking exist (Hirschi, 1969), and the concept could just as well be applied to innovation in science, engineering, and the arts. In a highly competitive environment, innovation may be required no matter how well prepared one may be to win by conventional means. In the case of video games, winning by deviant means such as cheats may be necessary for poor players but may provide an exhilarating challenge for especially good players.

Merton proposed a fifth category, rebellion, in which a person rejected both norms and values and substituted new ones in their stead. This might describe players who seek glitches not primarily for their possible use as cheats but for the intrinsic challenge of finding them. Glitch hunting, thus, is a metagame that defines victory in terms of discovering unknown software errors. Depending on the orientation of the player, this can be a master's game played against the programmer, or it can be a noncompetitive exploration of exotic territory, even at times having some of the quality of scientific research and discovery.

Many computer games are a contest in which the player strives to win, but not all. For example, the most popular single computer game, The Sims, does not involve winning or losing but allows the player to enjoy building a home and family life inside a computer simulation. Its creator, Will Wright, prefers to call it a software toy, but it is marketed as a game. Gonzalo Frasca (2003; cf. Walther, 2003) has adapted terminology for this distinction from the French philosopher Roger Callois: Paidia is play that may have rules but does not involve winning and losing, whereas the rules of ludus define winners and losers.

Richard Bartle (1996; cf. Bartle, 2004), one of the originators of online "MUD" games—multiple user dungeons or multiple user domains—has proposed his own independent typology that goes beyond the paidia-ludus distinction. He proposed that online game players chiefly enjoy four things: achievement, exploration, socializing, and imposition. (He also calls the last one *killing*, but we prefer the more general term *conquest*.) Achievers want to make progress through the game and amass points, but they are not very conscious of competing with other players. Theirs is a kind of intrinsic achievement motivation. Explorers seek novelty, experiences, innovations. Socializers enjoy the interaction with other players, without feeling driven to defeat them. Conquerors want to grind the other players into the dust and rise supreme above them. Note that both socializers and conquerors are primarily oriented toward the other players, whereas achievers and explorers are not. Rules are a much greater issue for achievers and conquerors than for socializers and explorers.

Rules of traditional games are typically written down, and the players follow them because they have a tacit agreement to do so. Video games, in contrast, chiefly embed the rules in the

game software. Indeed, the instruction booklet that accompanies the game says very little about rules, and part of the player's fun is discovering through experience what works and what does not. In other words, video game rules are largely implicit rather than explicit. This means that it is difficult to argue that it is unethical to exploit a glitch to win because that glitch is part of the software and thus open to empirical discovery like all the other features of the game.

The written instructions for Super Mario 64 tell the player that the object of the game is to collect stars and defeat the villain Bowser to rescue Princess Peach. But the instructions say very little about where the stars are to be found and what must be done to get them. Ironically, the similar instructions for Castlevania: Lament of Innocence turn out to be a lie. The player is told that the knight must rescue Sara from the vampire's castle, but in fact it is impossible to do so. When the knight has battled through five vast realms, vanquishing scores of monsters both great and small, and gains the five orbs necessary to enter the sixth and final realm, he learns that tragically Sara must die. His new mission becomes vengeance against the vampire, but just when he defeats this monster, he discovers that this goal, too, is inaccessible, and the vampire escapes. This double disappointment justifies the subtitle, "lament of innocence," and illustrates again the fundamental ambiguity of video games and perhaps of competition in life as well.

If the rules are implicit in the game, then from one perspective what is permissible becomes merely what is possible. This is especially the case for one-player games, where there is nobody with standing to complain if the player violates any rule. Some games for the consoles allow from two to four people to play simultaneously. Online games may include even several thousand players at once, although usually just a dozen or so in immediate interaction with each other. However, many players of one-player games compare their results with each other, often in what becomes a boasting contest. Thus, we can apply a standard distinction in computer-mediated communication between synchronous games, where players compete in real time, with asynchronous competition, when players establish reputations in the real world for accomplishment in any kind of game.

Glitches and cheats can be significant for synchronous and asynchronous competition. Wright, Boria, and Breidenbach (2002) analyzed player creativity in the online team combat game, Counter-Strike. When a player's character died, he or she could no longer pass information to his or her teammates by the legitimate route, yet he or she might know valuable things such as where an enemy sniper was hiding. Some players discovered that they could still communicate via the voting system that decides the terrain maps the game will be played on, and they used this glitch to help their team win. The game programmers subsequently had to close off this channel for communication with the dead. Some players employ hardware or software cheats to gain an advantage in online games, and other players may not realize they are doing this. Kimppa and Bissett (2005) analyzed such cases philosophically and concluded that exploiting glitches that already exist in the software is a legitimate way of winning, whereas winning by altering the software with other technology is unethical.

According to game designer Richard Rouse (2001), the best video games are nonlinear. With nonlinearity

> the player is not locked into achieving different goals in a specific order or in achieving all of the goals she is presented with. Instead, the player is able to move through the game in a variety of paths and can be successful in a variety of ways. (p. 556)

For example, in Super Mario 64, Mario is supposed to collect 70 of the 120 stars, but which 70 and in what order is largely left to the player. Winning by means of cheats, or redefining winning as glitch discovery, transcends the set of paths intended by the game manufacturer, thereby adding additional dimensions of nonlinearity.

Conclusion

Human beings need a proper balance of liberation and control, and the optimal degree of regulation varies across cultures, age groups, and activities. Without some rules, a game becomes unplayable; yet if the rules are too rigid, the activity is no longer play. Many people enjoy playing conventional games, competing to achieve set goals while slavishly following preordained rules. Yet in a society that values individualism and creativity, metagames may be considered a higher activity, even though they often involve rule violation or the exploitation of game manufacturing flaws. Victory in a metagame involves gaining the power to define the nature and rules of the game, thus maximizing autonomy rather than subservience.

Traditional games can achieve remarkable cultural stability; for example, the rules of standard chess essentially stabilized five centuries ago, with no changes at all in the past two centuries (Eales, 1985; Yalom, 2004). It is not clear, however, how metagames could ever stabilize because the essence of metagaming is transcending the current definition of the game. Thus, metagames have somewhat the same relationship to games as carnivores do to herbivores in the animal kingdom: Metagames cannot exist without games and metagamers are outnumbered by gamers. It is impossible to play an effective metagame unless several other people are playing the game that it transcends. This suggests that the social roles of particular glitches, cheats, and other metagames are likely to be highly dynamic. Social-scientific research on metagames will therefore need to be dynamic, multidimensional, and methodologically varied.

Research on the cognitive, affective, and social implications of glitches and cheats could be especially fruitful, but to this point very little rigorous, theory-based research has been done on the human dimensions of videogaming. The book *21st Century Game Design* (Bateman & Boon, 2006) is a start in this direction, and its cover copy says it describes "different play styles in terms of psychological models, using a cutting-edge audience model." Its primary model is a typology of four personality types, having some similarity to Bartle's model, but unfortunately it is based on the Myers-Briggs personality system. Despite its popularity with human relations consultants to American corporations, Myers-Briggs is poorly supported by research (Druckman & Bjork, 1991; Pettinger, 1993). Better would have been the Big Five model of personality dimensions, which has received extensive support in a wide range of scientific studies (Wiggins, 1996).

Future research could employ online questionnaires or intensive ethnographic methods to study the extended subculture represented by the numerous web sites that offer walk throughs of how to win games, hints about cheats, and occasional information about interesting glitches. It is also possible to directly study players who are interacting with glitches in multiplayer online games using interview, ethnographic, or participant observation methods (Foo & Koivisto, 2004). Research on free and open-source creation of computer games

already replicates the finding from more "serious" open-source software movements that bug finding is a major factor binding many of the participants to the collective activity (Mockus, Fielding, & Herbsleb, 2002; Scacchi, 2004). Thus, glitches and cheats may be a considerable source of social solidarity and status for members of the far-flung gaming community. Avid players of the games who begin hunting glitches and cheats thereby take the first step toward becoming professional programmers. It should not be forgotten that the founders of Apple Computer, Inc. got their start as manufacturers of a cheat device to make long-distance telephone calls, and today's hackers may be tomorrow's computer scientists (Freiberger & Swaine, 1999; cf. El Rhalibi, 2004).

References

Babbie, E. R. (2004). *The practice of social research*. Belmont, CA: Thomson Wadsworth.

Bartle, R. A. (1996). Hearts, clubs, diamonds, spades: players who suit MUDs. *Journal of MUD Research*, *1*(1). Retrieved May 16, 2006, from `http://www.mud.co.uk/richard/hcds.htm`

Bartle, R. A. (2004). MUDs. In W. S. Bainbridge (Ed.), *Encyclopedia of human-computer interaction* (pp. 472-475). Great Barrington, MA: Berkshire.

Bateman, C. M., & Boon, R. (2006). *21st century game design*. Hingham, MA: Charles River Media.

Bethke, E. (2003). *Game development and production*. Plano, TX: Wordware.

Chasing the dream. (2005, August 6). *The Economist*, *376*, 53-55.

Cloward, R. A., & Ohlin, L. E. (1960). *Delinquency and opportunity: A theory of delinquent gangs*. Glencoe, IL: Free Press.

Coleridge, S. T. (1907). *Biographia literaria*. Oxford, UK: Clarendon. (Original work published 1798)

Dalmau, D. S.-C. (2003). *Core techniques and algorithms in game programming*. Indianapolis, IN: New Riders.

Druckman, D., & Bjork, R. A. (Eds.). (1991). *In the mind's eye: enhancing human performance*. Washington, DC: National Academies Press.

Eales, R. G. (1985). *Chess, the history of a game*. New York: Facts on File.

El Rhalibi, A. (2004). Games. In W. S. Bainbridge (Ed.), *Encyclopedia of human-computer interaction* (pp. 269-275). Great Barrington, MA: Berkshire.

Foo, C. Y., & Koivisto, E. M. I. (2004, June). *Defining grief play in MMORPGs: Player and developer perceptions*. Presented at the international conference on advances in computer entertainment technology ACE 2004, Association for Computing Machinery, Singapore.

Frasca, G. (2003). Simulation versus narrative: Introduction to ludology. In M. J. P. Wolf & B. Perron (Eds.), *The video game theory reader* (pp. 221-235). New York: Routledge.

Freiberger, P., & Swaine, M. (1999). *Fire in the valley: The inside story of silicon valley's computer pioneers*. Foster City, CA: IDG Books.

Fullerton, T., Swain, C., & Hoffman, S. (2004). *Game design workshop: Designing, prototyping, and playtesting games*. San Francisco: CMP Books.

GTA sex scandal hits Australia. (2005, July 29). *BBC News*. Retrieved May 16, 2006, from `http://news.bbc.co.uk/2/hi/technology/4728261.stm`

Hidden sex scenes hit GTA rating. (2005, July 21). *BBC News*. Retrieved May 16, 2006, from `http://news.bbc.co.uk/2/hi/technology/4702737.stm`

Hirschi, T. (1969). *Causes of delinquency*. Berkeley: University of California Press.

Huizinga, J. (1949). *Homo ludens: A study of the play-element in culture*. London: Routledge and K. Paul. (Original work published 1939)

Kimppa, K. K., & Bissett, A. (2005). Is cheating in network computer games a question worth raising? In P. Brey, F. Grodzinsky, & L. Introna (Eds.), *Ethics of new information technology* (pp. 259-267). Enschede, the Netherlands: University of Twente.

Merton, R. K. (1938). Social structure and anomie. *American Sociological Review*, *3*, 672-682.

Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*, *11*, 309–346.

Neff, G., & Stark, D. (2004). Permanently beta: Responsive organization in the Internet Era. In P. N. Howard & S. Jones (Eds.), *Society online: The Internet in context* (pp. 173-188). Thousand Oaks, CA: Sage.

No more "hot coffee" sex for GTA. (2005, August 11). *BBC News*. Retrieved May 16, 2006, from http://news.bbc.co.uk/2/hi/technology/4142184.stm

O, J. (2005a). *Off the record, vol. 1—Animal Crossing*. Retrieved May 16, 2006, from http://www.n-philes.com/features.php?id=148

O, J. (2005b). *Off the record, vol. 2—Super Smash Bros. Melee*. Retrieved May 16, 2006, from http://www.n-philes.com/features.php?id=174

Petersen, T. (1994). On the promise of game theory in sociology. *Contemporary Sociology*, *23*, 498-502.

Pettinger, D. J. (1993). The utility of the Myers-Briggs type indicator. *Review of Educational Research*, *63*, 467-488.

Robinette, W. (2003). Foreword. In M. J. P. Wolf & B. Perron. (Eds.), *The video game theory reader* (pp. vii-xix). New York: Routledge.

Rouse, R. (2001). *Game design: Theory and practice*. Plano, TX: Wordware.

Scacchi, W. (2004). Free and open source development practices in the game community. *IEEE Software*, *21*(1), 59-67.

Schultz, C. P., Bryant, R., & Langdell, T. (2005). *Game testing all in one*. Boston: Thomson Course Technology.

Sex controversy over GTA game. (2005, July 11). *BBC News*. Retrieved May 16, 2006, from http://news.bbc.co.uk/2/hi/technology/4671429.stm

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, *27*, 379-423, 623-656.

Stark, R., & Bainbridge, W. S. (1979). Of churches, sects, and cults: Preliminary concepts for a theory of religious movements. *Journal for the Scientific Study of Religion*, *18*, 117-131.

Uproar grows over GTA sex scenes. (2005, July 26). *BBC News*. Retrieved May 16, 2006, from http://news.bbc.co.uk/2/hi/technology/4717139.stm

von Neumann, J., & Morgenstern, O. (1944). *Theory of games and economic behavior*. Princeton, NJ: Princeton University Press.

Walther, B. K. (2003). Playing and gaming: reflections and classifications. *Game Studies*, *3*(1). Retrieved May 16, 2006, from http://www.gamestudies.org/0301/walther/

Warcraft game maker in spying row. (2005, October 31). *BBC News*. Retrieved May 16, 2006, from http://news.bbc.co.uk/2/hi/technology/4385050.stm

Wiggins, J. S. (Ed.). (1996). *The five-factor model of personality*. New York: Guilford.

Wright, T., Boria, E., & Breidenbach, P. (2002). Creative player actions in FPS online video games: Playing Counter-Strike. *Game Studies*, *2*(2). Retrieved May 16, 2006, from http://www.gamestudies.org/0202/wright/

Yalom, M. (2004). *The birth of the chess queen: how her majesty transformed the game*. New York: HarperCollins.

Wilma Alice Bainbridge is a student at Yale University, with special interests in cognitive science and related fields. She has extensive experience programming computers, having started at the age of 9 and having worked on a number of projects in varied programming environments. She may be reached at wilma.bainbridge@yale.edu.

William Sims Bainbridge is a sociologist employed by the computer science directorate of the National Science Foundation. His latest book, *God from the Machine*, is based on computer simulations of religious cognition using a neural network multiagent system. He may be reached at wbainbri@nsf.gov.