

12

Game Engines as Open Networks

Robert F. Nideffer

Electronic games have become hugely popular, and are now claimed to be the favorite form of entertainment in the United States. According to a summary of recent studies compiled by the Entertainment Software Association,¹ two times as many people polled preferred playing games to watching television, while three times as many people preferred games to renting movies. The average gamer is 29 years old. The average American youth spends between 7 and 30 hours per week playing. Nearly 50% of players are now reported to be women (IDSA, 2003). Within the video game industry, there is general consensus that a large part of the future of games lies on the Internet. Games will proliferate across devices, touching anything that is able to access the Internet: console systems like the Xbox and Playstation, desktop and notebook computers, Tablet PCs, Smart Displays, PDAs, mobile phones, set-top boxes, and so on (Wolf, 2003). This proliferation is already well under way.

According to a report from market analysts The Themis Group, massively multiplayer online games (MMOGs)²—games that involve tens of thousands of concurrent players and millions of registered players in persistent online worlds—generated roughly \$1.3 billion during 2004 and will increase to an anticipated \$4 billion or more by 2008. The bulk of this will come from subscriptions, but a growing proportion will be generated by the sale of virtual property and in-game items (BBC News, 2004). By 2010 it's anticipated that the installed base of mobile game users alone will reach 2 billion (Hamilton and Stevenson, 2005). In 2000 Internet penetration exceeded 50% in the United States, with more than 53 million households connected. In 2004, according to a new study released by research firm Nielsen Netratings, three out of four Americans, or a total of 204.3 million people, had access to the Internet (Gruener, 2004).

One of the key enablers of this kind of growth has been the development and refinement of the “behind-the-scenes” software that makes games run, commonly referred to as the “game engine.” The term is usually applied to the software infrastructure that renders everything you see and interact with in the game world. Game engines provide the graphics capabilities, the physics models, the collision detection, the networking (when present), and the core functionality the player experiences during game play. Elsewhere I have argued (Nideffer, forthcoming) that one can think about the game engine as a culturally encoded “database interface,” that is, a mechanism through which a predetermined, relatively constrained collection of socially sanctioned procedures and protocols is used to render a world and make it

navigable in context. Along these lines, I have argued that it's important to look at the game engine as a cultural artifact that circulates within a specific social domain, in order to begin thinking about how to make more visible the implicit and often taken-for-granted assumptions operative during software development, as well as to extend the boundaries of what constitutes the game engine. I have done this in an effort to move beyond thinking of the game engine strictly in software engineering terms, and in an effort to also think about it in social engineering and networking terms.

My goal here is to examine the ways in which Internet gaming is beginning to address a fundamental social and technical challenge for digital culture: the relationships between the technical challenges of ubiquitous computing, the new social and creative opportunities available to users, and the persistent commercial pressure to segment online networks into private monopoly domains. I take up the fundamental and newly threatened issue of interoperability and openness within networks. As both work and play become articulated through online systems, these dynamics affect our capacities to reinvent our environments in fundamental ways—to affect not only the *content* of structured interactions but also the infrastructure and *context* in which they play out.

This chapter maps these dynamics through a series of short case studies of the social, technical, and commercial challenges associated with the “engines” that drive networked games and gaming:

- **KALI: the first internet gaming platform**
- **BNEDT V. BLIZZARD ENTERTAINMENT: a legal case that challenges issues of interoperability**
- **MAJESTIC: an ambitious though unsuccessful commercial title released by Electronic Arts in 2001 that incorporated a variety of mixed-media devices**
- **UNEXCEPTIONAL.NET: an open source net-art project that integrates blogs, location-aware mobile phones, and 3D game clients**

kali

Although little acknowledged, computer games have been at the forefront of many core areas in the computer and engineering sciences and have essentially supported multiple players since their inception several decades ago. The earliest and perhaps most prominent example is *Spacewar*, which came out of the Massachusetts Institute of Technology in the early 1960s and was

initially implemented on a giant TX-0, the world's first online time-shared computer. What is seldom reported is the role that the TX-0 played, as an infrastructure platform, in ushering in the next generation of designers and programmers. Early experiments on the TX-0 included not only *Spacewar*, but other game and gamelike programs such as *Bouncing Ball*, *Mouse in the Maze*, and *Tic-Tac-Toe* (Graetz, 1981). Initially, multiplayer meant little more than one user taking turns on the same computer at different times. However, it was not long before multiple players were able to use different input devices on the same machine simultaneously. Eventually, with the advent of personal computing and the ability to both create local networks between multiple machines (using cables), as well as connect to other players remotely (through IPX networks and ultimately via the Internet), people started designing and playing "networked" games in which physical copresence was no longer required.

What is particularly interesting about the cycle of development linking computing, gaming, and networking is how it arose out of a collective desire not just to play, but to play *together*. Games, or for that matter computer-mediated social interaction, had never been integral parts of the early visions of mainframe computing, personal computing, or networked computing. In each case, however, games and social network support became an early and ubiquitous unintended consequence. They were almost always the first direction of unofficial user-driven development of systems and have driven advances in many areas of computing, from graphics software and hardware to input devices, networked operating systems, multimedia delivery, and integrated communication protocols (e.g., those combining chat, email, instant messaging, voice-over Internet protocol, fax, and global positioning systems).

The first robust program facilitating Internet gaming was Kali, a system developed in the mid-1990s by two independent programmers, Jay Cotton and Scott Coleman. Kali was a software solution for connecting multiple machines together, whether located remotely or in physical proximity. It performed a few simple functions: initiate and verify connections with all players in the game and configure it dynamically so that all machines are using the same set of game parameters; support an Internet Relay Chat-like mode whereby players could communicate once connected; and provide an online player directory so that opponents could be selected from a list (Cotton, 1994).

Kali—named after a goddess of death and destruction—was first used to support *Doom*, the groundbreaking, ultraviolet first-person-shooter (FPS) game released in 1993. Kali changed the way *Doom* was played, shifting the dynamic from a person–computer interaction to a much more social experience in which multiple players competed or cooperated in the same online environment. Going far beyond the editable parameters exposed by the designers themselves—difficulty, speed, and so on—Kali opened up a new form of play. It encouraged players to take part in the construction and manipulation of their experience at a deeper and more fundamental level than those enabled through the consumer interface. It traded transparency in the sense favored by game designers of a minimally intrusive interface, for transparency in the sense favored by hackers (and, not incidentally, social scientists)—that of revealing underlying processes.

Although games increasingly include secondary interfaces that make them more editable and modifiable, ranging from player customization of graphics and sound to building new levels, the Kali process exemplified a desire among users to act not only as *content* providers for a game engine, but as *context* providers who can rearticulate the uses of the engine. This is an important distinction that raises questions about the larger computing environment in which games operate and about the freedoms that users can exercise within that environment. Changing the communications infrastructure used by gamers, as Cotton and Coleman did, is contextual in that it changes the forms of sociability through which the game is played. By the late 1990s, as Internet gaming began to take off, commercial game companies increasingly saw control over contextual issues as part of their business models. When, several years later, two programmers undertook a similar programming intervention in the context of battle.net, Blizzard Entertainment's online game space, they found the scope of their freedoms much more constrained.

bnetd vs. blizzard entertainment

Blizzard Entertainment is a successful game development company and publisher located in a corporate research park adjacent to the University of California, Irvine. Founded in 1991 (as Silicon and Synapse), Blizzard shipped its breakthrough hit, *WarCraft*, in 1994. Since *WarCraft*, Blizzard has had a succession of best-selling titles, including *WarCraft II* (1995), *Diablo* (1996), *StarCraft* (1998), *Diablo II* (2000), *WarCraft III* (2003), and most recently its

hugely popular MMOG game *World of Warcraft* (2004). Like many game industry startups, Blizzard passed through several corporate buyouts, ending up as one of the early (1998) media acquisitions of Vivendi Universal, a wide-ranging global media conglomerate. This trajectory was a common one in the electronic game industry in the 1990s, and it maps closely onto the growth of games into the dominant global entertainment media. It also provides a context for the shift in attitudes toward interoperability and other questions of network infrastructure in the course of Blizzard's releases.

For *WarCraft II* and other early games, Blizzard used Kali as its default software for Internet play. It even packaged Kali with its games (Miller, 2002). With the 1998 release of battle.net, Blizzard tried to internalize and centralize the social networking and hosting features enabled by Kali. Battle.net provided an arena for Blizzard customers to chat, challenge opponents, and initiate multiplayer games at no cost to the user beyond the purchase price of the games. According to Blizzard, controlling the portal through which its software titles were played allowed them to do better evaluation of product usage. Arguably more important from a business standpoint, it allowed them to authenticate players using a key method, making it less likely that owners of pirated copies could take advantage of the added value of online play.

In 1998, University of California, San Diego, student Mark Baysinger identified the protocol that the *StarCraft* clients used to connect to battle.net — by most accounts a fairly modest feat. In April Baysinger posted the first version of Starhack, a chat service for *StarCraft* players on battle.net. When Baysinger turned his attention to other projects, he reissued Starhack under the free software general public license (GPL). Other programmers and Blizzard game aficionados soon picked up the project. As Blizzard released new games, Starhack grew into a general battle.net emulator and was eventually rechristened “bnetd.” At its peak, bnetd had 10 listed developers (Miller, 2002). Two of them, Ross Combs and Rob Crittenden, became the bnetd lead developers. A third, Tim Jung, became bnetd's Internet service provider and systems administrator.

Like tens of thousands of others, Combs and Crittenden liked to play Blizzard games on battle.net, but it was notoriously buggy and often slow and frequently crashed due to player volume. It also suffered from a number of “social malfunctions” — notably, wide latitude for players who enjoyed killing other players, and a series of well-known hacks that conferred unfair

advantages in the games. This made the game play experience frustrating for many newcomers and experienced players.

The mature bnetd performed all the functions of battle.net, but it opened them up to the players themselves. Players could download the open source software, install it, modify it if they wished, and run their own bnetd servers for playing Blizzard games (or potentially any number of other games). Combs and Crittenden argued that the motive was to facilitate fun, bug-reduced gaming sessions within friendly communities that would be respectful of one another. With bnetd in wide circulation, gamers no longer had to use the official battle.net site to play Blizzard games. Groups could play *StarCraft* or *WarCraft* independently, using free, locally hosted copies of bnetd as the platform.

Bnetd did not copy battle.net's code; rather, it was a ground-up reinvention of the functions of battle.net. This is a common meaning of "reverse engineering" in software development. It reflects the fact that commercial software vendors sell only the compiled code for their products—the ones and zeros read by computers—not the source code written (and readable) by humans. It further reflects the copyright protection afforded software, which protects code as if it were an original piece of writing, but does not cover its functions. Reverse engineering, in this context, requires close observation of the functions of a piece of software and the creation of a new program that can duplicate them.

The basic method for creating bnetd involved "packet sniffing" and "interception." Packets are the fundamental units of information carriage in modern communication networks. A packet consists of a *header*, which contains the information needed to get the packet from the source to the destination, and a *data area*, which contains the information provided by the creator of the packet. In its simplest form, a packet sniffer captures the data packets that pass through a given network interface. Once sniffed, the information contained within the packet can be analyzed and the software functions that produced it inferred. Bnetd identified these packets and built new, analogous functionality around them.

Although Baysinger had been threatened with cease and desist letters as early as 1998, no action was taken. This changed with the impending release of *WarCraft III* in 2002. Vivendi and Blizzard issued a new cease and desist letter and brought suit against bnetd developers for a series of violations: of the copyright on the software, of the license on the purchase

of software (the end-user licensing agreement, or EULA), and finally of the so-called “anticircumvention” clause of the Digital Millennium Copyright Act (DMCA). Of these, the last two invoked new, largely untested, and potentially far-reaching legal protections afforded software vendors. The EULA issue involved a test of the limits of the rarely read click-through software licenses that accompany most software. Could, for example, a EULA expressly forbid reverse engineering, as the Blizzard license appeared to do? The anticircumvention issue concerned an originally obscure clause in the DMCA that criminalized any attempt to circumvent the technological measures used to restrict access to copyrighted digital works, such as encryption or registration on a central server. The clause targeted practices that were common in many programming settings, ranging from encryption research, to forms of reverse engineering, to personal practices of archiving and backup. The latitude to create software that bridged proprietary systems—thus ensuring interoperability—was, in theory, severely curtailed by the new law. The anticircumvention clause went well beyond a simple additional protection for copyrighted work; it potentially broke the wide range of “fair use” exceptions to copyright for digital media (see Karaganis, Chapter 16 in this volume on modalities of control; see also Firooznia, 2000).

Blizzard won its case on both the EULA and anticircumvention issues in 2004 and again on appeal in September 2005. The consequences of the decision are worrying but far from clear. If interpreted broadly, it may enable any company to create a gated monopoly on network communications, bringing the interoperability of the Internet to an end (Wen, 2002).

majestic

It is a widespread article of faith in the software and computing fields that the future belongs to “ubiquitous computing,” a vision of networked computers embedded in a vast array of devices and appliances. As in the past, game design is likely to pioneer this development. Already, network games are extending into heterogeneous computing environments that combine cell phones, PDAs, desktop and WIFI mobile computers, handheld game devices, and game consoles into continuous and contiguous multiplayer game experiences.

A fundamental challenge in the transition to heterogeneous networked gaming is how to synchronously communicate or represent the actions of players to each other in an effort to ensure that everything

happens at the same time on all systems, regardless of network latency or differences in machine speed. Such synchronization is crucial to creating a consensually coherent reality. Other important pieces of the puzzle involve developing compelling strategies for taking advantage of pervasive technologies like phone, fax, email, and the Web as part of the overall game environment in ways that make use of the “native” capabilities of the devices. It also involves thinking about how to design compelling social interaction through that multiplicity of devices, how to open up authorial control to allow players to have more open-ended and flexible play spaces, how to allow players to modify those play spaces within an existing domain, and how to provide customizable services that enable them to build new play spaces when desired.

Arguably, the most compelling commercially released network-centric game to date was *Majestic*, released by Electronic Arts in summer of 2001. *Majestic*'s marketing hook was that “the game played you.” Conceived as an episodic web-based adventure, players created accounts, logged in, downloaded the custom-built application to run the first installment for free, and then were charged for subsequent installments if they chose to continue the game. In terms of narrative, *Majestic* was framed as an *X-Files*-style government conspiracy and cover-up, delivering its plot twists through a variety of interfaces and client devices.

The innovative conceptual move made by the designers was to have the game take advantage of everyday communication technologies like the Internet, phones, email, instant messaging, and fax. *Majestic* sought to create a narrative experience that blurred the line between lived space and game space. Upon registering, players were able to set the parameters of in-game communication. Depending on these choices, players could be contacted at any point during the day or night by game operatives who would either give them vital pieces of information to aid in moving them to the next stage in the drama, or provide them with misinformation in an effort to send them off track. Part of the developer's goal was to have *Majestic*'s episodic structure appeal to an older generation of gamers who did not have time to sit in front of their computers for hours on end, but wanted to periodically drop in and spend an hour or so to try and decipher a clue or see who had been attempting to contact them.

The result was a costly and high-profile failure. Despite well-financed advertising buzz, the pervasive dimension of the game was very poorly received from the outset: Of 800,000 initial registrants for the free first

installment, only 72,000 completed the process. Only 10,000–15,000 paid for the next installment. Overall, Electronic Arts lost an estimated \$5 to 7 million (Morris, 2003).

Several reasons were offered for *Majestic's* failure. One claim circulated in the gaming community (in part by Electronic Arts) was that the subject matter of the game became too controversial after the World Trade Center bombings. The broader cultural hiatus on sinister portrayals of government in the wake of the attack lends some credibility to this claim. *Majestic's* terrorism themes resulted in the temporary suspension of the game after 9/11. I would argue that another part of the “failed” *Majestic* story had to do with the company’s anxiety over what might happen if players had too little control over the technological interfaces to the game. Several news stories in the gaming press postulated the consequences of family members answering phones or receiving faxes from unsavory characters claiming that their loved ones’ lives were at stake. This anxiety resulted in the developers’ requiring players to grant “access rights” upon registration, effectively destroying the potential “surprise factor” of the game. This also prevented the game makers from pushing more aggressively at the lived space–game space boundary. Hindered by endless disclaimers and legal protections, *Majestic* became “too safe” to realize its full potential.

Although monumental in scope, according to the game’s developer, Neil Young, *Majestic's* ambitions were not particularly complex from a technical standpoint.³ Not surprisingly, implementation became costly and difficult because of the proprietary issues and third-party companies that stood in the way of integration. Voice-over IP, instant messaging, and faxes all had to be worked out with service and technology providers. Negotiating such partnerships was no small task, and required considerable time and financial investment—reportedly in the neighborhood of \$10 million and several years in planning and implementation.

The resulting “experience server,” as the developers called it, was the closest example to date of a massively distributed, pervasive, multimodal gaming environment. The story of *Majestic* provides a clear and still very relevant indicator of the difficulties facing large-scale integration of social and gaming environments across heterogeneous and often proprietary communication networks. *Majestic* did not get far enough to run into player-driven questions about customization, modification, or fundamental retooling of the game environment. Rather, its failure illustrates the extent to which such

ambitions require either much larger corporate synergies than Electronic Arts could dispose of (despite numerous partnerships with AOL, Microsoft, and other media giants), or a much more open network architecture. With the DMCA providing legal leverage for a network model organized into proprietary silos, the odds of successful future developments in this direction lie heavily with the media giants.

unexceptional.net

Games are already the number-one downloaded application on mobile devices, representing upward of 90% of total download requests. According to Datamonitor analysts, in 2005 more than 200 million people in the United States and Western Europe—80% of all wireless phone users—will play online games using wireless devices. Four of the major mobile phone manufacturers—Nokia, Siemens, Ericsson, and Motorola—recently established the Mobile Games Interoperability Forum (now consolidated into the Open Mobile Alliance, or OMA), which aims to define open standards that will let developers create and deploy games across multiple game servers and wireless networks for a variety of mobile devices. The creation of the forum signals an understanding that, in a field as immature as mobile gaming, proprietary standards are more likely to lead to costly balkanization and underdevelopment across the sector than to profitable monopoly positions. The sector is served when all actors participate in the creation of tools (Wrolstad, 2001).

The OMA represents a major positive step toward interoperability in one significant and growing portion of the industry. Unfortunately, the competitive landscape in other sectors leads to other calculations, often by the same corporate actors. The leading game console makers each have proprietary development suites and protocols, as do personal computer manufacturers, handheld device manufacturers, non-OMA mobile phone makers, and so on. Most are jockeying for dominant positions in the delivery of digital media. At the moment, however, the only thing the expanding multitude of devices will share is the need for an IP address. The kind of seamless interoperability between devices described above is not likely to emerge from within this competitive corporate climate anytime soon. It is for this reason that academia may be able to play an important role.

“Anywhere, anytime access” is the mantra of Cal-(IT)², the recently established California Institute for Telecommunications and Information Technology⁴—an initiative between University of California, San Diego

(UCSD), and University of California, Irvine (UCI), dedicated to pushing the technical and social boundaries of IT development. The Game Culture and Technology Lab, which I founded in 1999 at UCI, brings together an interdisciplinary community interested in using game metaphors, design principles, and technologies to develop next-generation multiuser environments for artistic exploration, scientific visualization, and informal science education. As an institutional experiment, the Game Lab is itself an effort to ground the synthesis of perspectives necessary to bridging heterogeneous contents and contexts, both social and technological. A core premise of the lab is that appropriation, misuse, and hacking of technologies are not only legitimate forms of research and development but also can prove central to the process of innovation.

A project that has occupied much of my time lately is “unexceptional.net,” an effort to build an arts-driven game space that can operate across a range of client devices. Unexceptional.net is a mystical-realist journey catalyzed by a series of interconnected events related to sexual infidelity, political conspiracy, and spiritual transformation. The project draws on the traditions of comics, graphic novels, and computer games in order to create an environment that crosses boundaries between pop culture, fine art, and social critique.

The central character of unexceptional.net is “Guy,” a frustrated comic artist, game designer, and hacker who has recently found out that his long-time partner is having an affair. This discovery launches him on a series of quests to gain insight into the nature of his partner’s relationship. Guy’s experience is infused with a disturbingly co-opted and corrupted Eastern philosophy and spirituality that dictates the nature of the quests, and of the ultimate goal, his search for “enlightenment.” To achieve enlightenment you must follow Guy on a series of web- and GPS-based quests to find special key objects that will help unlock and open all seven of his major chakras, the energetic centers of the body according to Buddhist doctrine.

The project involves an extensive database infrastructure for storing and delivering game-state data via the Web, GPS-enabled mobile phones, and a 3-D game client. A blog is used to give the player information about the current game state, player locations, and quest progress. The blog also provides access to an administrative framework enabling game designers to alter the game and have it immediately reflected in the various game clients through a series of user-friendly web pages.

One of the key innovations of unexceptional.net is the way we procedurally generate the game world on the phone. All terrain and structure data used in the game is location specific and sent to the phone from the game server during game play. The game world thus aesthetically represents the physical environment in which it is played. If one is in the desert, one sees an abstracted representation of the desert; if one is in the middle of the ocean, one sees water; and if one is in a city, one sees urban space. This makes the game world extremely extensible, since the small memory footprint and screen size of the phone are no longer a liability in terms of more complex and emergent game play.

In addition to using the screen of the phone to display the game, we also allow players to use their voices to advance quests.⁵ This allows players to receive calls based on where they are in physical space, so that they can continue quests in “voice mode,” better using the native capabilities of the device. Finally, we have incorporated a 3-D client into unexceptional.net using the Torque Game Engine. The initial goal is to make the 3-D client mirror how the GPS phone client works. In other words, we algorithmically generate the 3-D game world and allow content creation and modification to a far greater degree than is currently common in networked 3-D gaming environments.

As game play ensues, the game state is continually fed to the server via the different client interfaces and broadcast back out to those interfaces. If, while playing the mobile phone game, the player logs back in to the blog, the impact of game play via the phone will be reflected. As the player continues to do things via the blog, the phone and the 3-D clients will be affected. These are just several of the innovative methods we have been able to explore by using a combination of free or cheaply available software while exploiting the network protocols that support this type of interoperability. A main goal of unexceptional.net is to push at these boundaries and develop capabilities that can be made freely available to people for their own creative experimentation.

In summary, key objectives of the project include: (1) using unexceptional.net as a test bed for deploying custom-designed and freely distributed software that takes advantage of everyday communication technologies such as blogging, email, 3D gaming, and mobile telephony in order to enable anywhere, anytime access to heterogeneous game worlds; (2) implementing the game infrastructure in such a way that it can be used for alter-

native content development and deployment; (3) facilitating ease of content creation through provision of web-based tools for game modding (using commercial game products to create custom levels in existing titles); (4) sharing the results in the public domain through Internet distribution, formal exhibition in fine-art contexts, professional conferences and events, and publication; and (5) exploring novel forms of individual and community interaction.

in closing

Despite the concentration of ownership associated with Vivendi-like corporate media structures, the current network milieu is still primarily one of fragmentation and divergence, characterized by a proliferation of devices connected through discrete networks. This contrasts with most accounts, which envision devices interacting with one another through a single network infrastructure. Such an infrastructure, or “engine,” would enable social actors to participate in the same online world from whatever device they chose, while maintaining persistent identities and accounts. It would allow them to carry their personas, assets, and social networks between platforms, to scale media in sensible ways, and to have anywhere, anytime access to their shared communities of interest. Mizuko Ito (Chapter 6, this volume), danah boyd (Chapter 8, this volume), and Shay David (Chapter 11, this volume) have all explored aspects of this ambition in their contributions. Ensuring that such an infrastructure remains open and customizable would be a giant step forward in putting both content and context back into the hands of a more diverse, eclectic, and potentially innovative population of players.

There are different levels at which creative work happens. With respect to net-art practices, for example, an important distinction can be made between work that is made using available tools and technologies (e.g., precoded software applications such as image, sound, and video editors, word processors, 3D modeling programs, and the like), and work that is made by retooling the tools, or by changing the infrastructure that the tools run on (like custom-coded web browsers, image processors, sound generators, etc.). This is another way of approaching the distinction between *content* and *context* creation. In the first case, artists and designers provide content for and work with existing infrastructure; in the second, they rework infrastructure in the interest of creating alternative contexts for interaction and experience. There is no clear line between the two modes of practices, and we are not well served intellectually by drawing them. I would argue that we are better

served the more we facilitate that crossover and the less we let that capacity be determined by the size and scope of commercial actors. The range of newly empowered synthetic cultural and technological practices—hacking, cracking, poaching, sampling, mixing, appropriating, misusing, reverse engineering, and others—all partake of and depend on this open dynamic.

With regard to game engines, this means exposing the tools of creation so that they also can become a primary place of play. Practices like modding and machinima (using 3-D game engines as real-time movie-making platforms) signal ways in which gaming has taken important steps in this direction. We have much more social and technical work to do to understand how to promote those strategies in a distributed networking environment. We have much more political work to do to ensure that those technical potentials can both empower users and be widely enjoyed.

In an interesting and timely article from 2000, author and game developer Crosbie Fitch playfully made a pitch for a new protocol (used loosely) that he terms the distributed Internet operating system, or DIOS. From Fitch's perspective, DIOS will facilitate the equitable pooling and exploitation of all information resources around the world. According to Fitch, the most suitable candidates for a DIOS are MMOGs and the engines that support them. The ones that survive, he argues, will be scalable, distributed systems that solve the issues of synchronous presence, diversity of devices, and capacities for expressive behavior. He argues that these goals are much more proximate to the entertainment industry than to financial, military, or other industries. For Fitch, this makes MMOGs not only the future of games or the primary form of entertainment for this century but also the future of the Internet itself (Fitch, 2000). In a nutshell, this is what we believe in the Game Culture and Technology Lab as well, and what, in our own small way, we are collectively working toward.

references

- BBC News. (2004). Online games make serious money. Available online at <http://news.bbc.co.uk/1/hi/technology/3403605.stm>
- Cotton, J. (1994). *History of Kali and iFrag*. Available online at <http://games.igateway.net/kali/history.html>
- Firooznia, H. (2000). *Hackers vs. Hollywood, round 1: A very brief history of DeCSS and the DMCA*. Lodestone.org. Available online at <http://www.lodestone.org/people/hoss/politics/DeCSS-history.html>
- Fitch, C. (2000). Cyberspace in the 21st century: Mapping the future of massive multiplayer games. *Gamasutra*. Available online at www.gamasutra.com/features/20000120/fitch_01.htm
- Graetz, J. M. (1981, August). The origin of *Spacewar*. *Creative Computing*. Available online at <http://www.wheels.org/spacewar/creative/SpacewarOrigin.html>
- Gruener, W. (2004). *Internet penetration in the U.S. passes 75 percent*. Available online at http://www.tomshardware.com/hardnews/20040319_130922.html
- Hamilton, F., and Stevenson, D. (2005). Worldwide growth of the mobile gaming market has exceeded expectations. Available online at <http://www.screendigest.com/reports/05wiregambus/FHAN-6JUD8L/pressRelease.pdf>
- Interactive Digital Software Association. (2003). Media center. Information available at <http://www.idsa.com/pressroom.html>
- Miller, E. (2002). Analysis of BNETD and Blizzard. *LawMeme at Yale Law School*. Available online at <http://research.yale.edu/lawmeme/modules.php?name=New&file=articl&sid=149>
- Morris, C. (2003). Electronic Arts' online folly. Available online at http://money.cnn.com/2003/03/04/commentary/game_over/column_gaming/
- Nideffer, R. F. (forthcoming 2007). Game engines as embedded systems. In V. Vesna (Ed.), *Database aesthetics*. Minneapolis, MN: University of Minnesota Press.
- Wen, H. (2002, April 14). Battle.net goes to war. *Salon.com*. Available online at <http://www.salon.com/tech/feature/2002/04/18/bnetd/>
- Wolf, C. (2003). *Forecast and analysis of the worldwide internet access device market, 2002-2007*. Available online at http://ealibrary.bitpipe.com/detail/RES/1070672414_890.html
- Wrolstad, J. (2001). Forum pushes open standards for next-generation wireless gaming. Available online at <http://www.newsfactor.com/perl/story/11746.html>

- 1 Formerly the Interactive Digital Software Association (IDSA).
- 2 Also referred to as MMORPGs (massively multiplayer online role-playing games).
- 3 When I spoke with Young (a vice president at Electronic Arts and general manager at Maxis) in late Fall of 2003, he was reluctant to even talk about the game, apparently viewing it as a failed project, at least commercially. He was far more eager to discuss the recent *Return of the King* title he had just produced. The entire floor at Electronic Arts was filled with materials associated with the Tolkien saga—life-size characters, wall maps of Middle Earth, architectural models, figurines, and so on. Our conversation died fairly quickly and after I said I believed history would prove *Majestic* a far more significant contribution to game culture and technology than any of the *Lord of the Rings* titles.
- 4 One of four recently established California Institutes for Science and Innovation, and the product of a partnership between the University of California, San Diego, and the University of California, Irvine, campuses.
- 5 This has been done by integrating a freely available telephony software called Asterisk, which incorporates automated call routing. We have hooked a speech-to-text and text-to-speech system called Sphinx (developed at Carnegie-Mellon University) into Asterisk.

