The Driving Technology Behind *Quake III: Arena*

Eric Chan
`ericchan@cs.stanford.edu`

STS 145: The History of Computer Game Design
March 20, 2001

The Driving Technology Behind *Quake III: Arena*

by Eric Chan

The Next Big Game

It came as no surprise to fans of the successful *Quake* franchise when id Software announced that their next big game would be a first-person shooter titled *Quake III: Arena*. Started in 1991 by John Carmack, Adrian Carmack, and John Romero, id Software practically invented the first-person shooter (FPS) genre with the shareware *Wolfenstein 3D* in 1992, then cemented their place in game history with the smash hit *Doom* the following year. *Quake* was the first game that showcased a true 3D environment with polygonal models instead of sprites; *Quake II* achieved new levels of detail and realism by being the first PC game to take advantage of 3D acceleration in hardware. With their newest action title *Quake III: Arena*, id Software continues to push the envelope and to engineer cutting-edge technology. Indeed, while the games themselves have always been fun to play, we shall see that technology is the driving element behind id Software's continued success in the FPS genre.

Quake III was fundamentally different from its predecessors in that it was designed from the beginning to be solely a multiplayer game. Rather than add solo levels with a weak background story, id Software abandoned the single-player paradigm altogether and instead focused on its strength: multiplayer deathmatch. Consequently, the game designers realized from the start that it was essential to provide the best online gaming experience possible.

Graphic Detail

Before settling into a game of deathmatch for the first time, one is usually struck by *Quake III*'s stunningly detailed graphics. The game sports a multipass 3D engine, designed and implemented by lead programmer John Carmack, that is capable of rendering curved surfaces for sleeker and more flexible level architecture. The engine also supports volumetric fog, realistic lighting and shadows, mirrors, and a number of special effects, including lens flares, explosions, dynamic lights, smoke trails, and energy burns. It also brings to life the richly detailed textures, character models and animation created by Adrian Carmack, Kevin Cloud, Kenneth Scott, and Paul Steed. While graphic detail does not affect the mechanics of game play, having superior visuals has always been a strong selling point for FPS titles. Online previews often show in-game screenshots of upcoming games, and a game that can produce eye-catching screenshots attracts more consumers.

The fact that *Quake III* sports cutting-edge graphics may not be so surprising, since fans of the *Doom* and *Quake* franchises have come to expect the best visuals from id's games over the years. John Carmack decided to use OpenGL over Direct3D for the 3D graphics API, a choice that one finds less and less common in PC game studios today. Recently, more game developers have turned to

Direct3D, a part of Microsoft's DirectX API, for 3D graphics work. Carmack's insistence on using OpenGL illustrated that GL is still a solid and effective 3D graphics API, and also encouraged 3D video card manufacturers to improve the quality of the GL software drivers for their products. Indeed, when it became evident in early 1999 that id Software was aiming for a Christmas release, top video card companies ATI, 3dfx, and nVIDIA scrambled to ensure that their latest products would run *Quake III* flawlessly. They all knew that id Software's next installment in the popular *Quake* franchise would be a sure winner in the holiday season. Since the *Quake III* engine is graphics intensive and requires a video card with strong 3D hardware acceleration, the game succeeded in driving video card sales.

Reaching Out To Multiple Platforms

Another solid reason to use OpenGL over Direct3D is that OpenGL implementations exist on all major platforms. From the beginning, id Software intended *Quake III* to be a multi-platform game, specifically for the Microsoft Windows, Apple Macintosh, and Linux operating systems. OpenGL works smoothly on all three of these platforms, whereas Direct3D currently runs only on Windows, not surprisingly. Not only did id Software design *Quake III* to run on these three platforms, but they actually developed and released the game simultaneously for all three. This is unlike most game studios, which usually develop and release a title on a single platform (typically Windows), then hire other companies to port the title to other platforms.

A famous example of this scenario is Epic Games's *Unreal*, which was developed and released for Microsoft Windows first. Epic then hired Westlake Interactive to bring the popular FPS title to the Macintosh, a decision that illustrates two major disadvantages of this general approach. First, the ports for other platforms are usually released much later than the original, due to the additional development time needed. Second, and more important, this delay problem only gets worse when the game is updated or patched. Because of various bugs and issues with the original *Unreal* release, Epic released a number of minor patches. However, they were slow in sending the updated code and communicating the changes to Westlake Interactive, which delayed updates for the Macintosh and alienated the Macintosh community. Furthermore, porting houses are traditionally contracted for the original game port only, not for updates. Hence, Westlake earned zero revenue for the man-months needed to patch *Unreal*, but doing so was necessary in order to keep a solid fan base on the Macintosh, the core of Westlake's business. id Software avoided all of these sticky issues by simultaneously developing *Quake III* for each of the three target platforms. Unfortunately, most development studios don't have the experience or the technical expertise to work out the nuances of multiple operating systems against a tight deadline, and hence choose to focus on a single platform.

The PAK System

Technologically, the simultaneous development and release plan was made possible by the use of the PAK virtual file system. As we shall see, the PAK system not only eases development, but also facilitates game expandability, helps to combat online cheating, and remains instrumental to id Software's technology licensing business. A PAK file is simply a file that encapsulates and compresses a set of files or directories. In the case of *Quake III*, all game data files such as textures, sounds, models, levels, and scripts are stored in a small set of PAK files. The immediate advantage of this approach is that the PAK files work transparently on all platforms. Thus, when id Software shipped *Quake III*, they only needed to include a few files on the game CD: a small game executable (which is compiled separately for each platform), and a set of PAK files containing game data (the same PAK files for all platforms).

The PAK system also serves to keep *Quake III* expandable. Some have criticized the game for offering only a limited number of game play modes (Deathmatch, Team Deathmatch, Capture The Flag), but id has traditionally left the more specialized and fancier game play modes to the mod community. Hence, the design of *Quake III* required that it be easy for fans to create new levels and mods. A mod is essentially an add-on game play mode with different rules. There are relatively simple mods, such as Instagib, in which players compete only with the railgun. There are also highly sophisticated mods such as Urban Terror, which feels much more like a military simulation due to the slower game pace, the urban surroundings, and the fact that all the default weapons have been replaced by real-world military ones. When creating such complex new levels and mods, designers need to include a number of files, often new textures, scripts, and map data. Using tools freely provided by id Software, these can be encapsulated into a single PAK file. There are a number of advantages to this approach. Since all new game content is stored in a single PAK file, players who want a new mod usually need to download only a single file, and since PAK files are compressed, these downloads proceed relatively quickly, even over low bandwidth connections. Most important, since PAK files work on all platforms, a mod or level designer can develop her content on a single platform (typically Windows), and the result will work automatically across all platforms.

Game Engine Architecture

In order to understand how the PAK system is useful to id Software's technology licensing business and how it helps to prevent online cheating, we must first understand the components of the game engine architecture. John Carmack divided the game engine into the following modules:

- `game`, the server-side game
- `cgame`, the client-side game
- `renderer`, the 2D and 3D graphics renderer
- `q3`, the core program

The `game` module defines the rules of the game, such as the amount of damage each bullet does, how fast the players move, the physics, time limit, and the objectives. The `cgame` module defines the client-side version of the game, that is, what the players sees and hears when playing the game. Thus, `cgame` is responsible for sending commands to the engine to draw the scene, play sounds, and handle the interface. The key, however, is that nothing in `cgame` actually affects the *rules* of the game or the actual game play. The `q3` module acts like glue, tying the various engine components together, including low-level details such as input devices, sound mixing, window management, network code, and AI. Lastly, `renderer` is responsible for all the 2D and 3D graphics drawing.

This division may seem arbitrary, or at most intended only to facilitate engineering, but in fact it has a number of important consequences. Traditionally, a developer implements these modules in a programming language like C or C++ and compiles them into a finished executable. However, this was not the case with *Quake III.* Carmack chose to include only the `q3` and `renderer` modules in the final executable. The other two modules, `game` and `cgame`, are included in the PAK files along with the other game data. Furthermore, the source code for `game` and `cgame` are publicly and freely available!

Applications of the Game Engine Architecture

This seems at first like a very bizarre arrangement, but in fact it has proven to be extremely practical. The purpose of making the source code for `game` and `cgame` available is to allow players to make mods. Indeed, notice that these are the modules that define the exact rules of the game and also the client-side effects (pictures, sounds, etc.) experienced by the player – precisely the aspects of the game that mod designers enjoy tinkering with. They can make changes to the code and recompile it into a new PAK file that *Quake III* recognizes as a new game mode. Thus, the decision to include `game` and `cgame` in PAK files instead of the game executable led to a flexible way for third-party developers to change the game play to suit their personal tastes.

The act of storing the `game` and `cgame` modules into the PAK files also helped to combat online cheating. Since id Software was striving to provide the finest online multiplayer experience, it was necessary to have mechanisms to prevent cheating, another factor that led to *Quake III*'s particular game engine design. In past games such as *Quake II*, savvy but unscrupulous players were sometimes able to "hack" the executable, which enabled them to cheat while playing. There were common cheats such as the "aim bot" which gave a player virtually perfect aim, as well as more subtle ones which allowed a player to know the location of other players at any given time. With *Quake III*, many of these issues have been successfully suppressed, since the actual game code that dictates the rules and all client behavior (such as aiming) reside in `game` and `cgame`, respectively, which are stored in PAK files. When a player joins a game server, *Quake III* verifies that the PAK files on her computer are the same as the ones on the server. This ensures that no tampering with the `game` or `cgame` modules has occurred.

Technology Licensing[1]

We have seen how `game` and `cgame` allow the mod community to expand the game in new and interesting ways, and even how these modules help to reduce online cheating. The `q3` and `renderer` modules, on the other hand, contain the actual cutting-edge technology that drives the game: the amazing and flexible 3D engine, the finely-tuned network code, the sound mixer, and the bot AI system, together engineered by Carmack, John Cash, Robert Duffy, and Jim Dose. While the `game` and `cgame` modules were made available for public consumption, the `q3` and `renderer` modules are kept private for engine licensing purposes.

Technology licensing has traditionally been an important part of id Software's business. The company acquires revenue not just from spectacular game sales, but also from the sales of other games based on their engines. Such games include *Heretic*, *Soldier of Fortune*, *Half-Life*, *Daikatana*, *Sin*, *Heavy Metal: Fakk 2*, and *American McGee's Alice*, with the last two in particular using the Quake III engine. Clearly, then, the fact that *Quake III* has a flexible, sophisticated game architecture with cutting-edge graphics is not just a selling point for the consumers, but for potential licensees as well. From the point of a development studio with neither the time nor the resources to develop a state-of-the-art game engine from scratch, licensing a proven and battle-tested engine from an experienced company like id Software is an attractive option. Licensees can even improve and modify the code as appropriate for their game's style and atmosphere.

Not surprisingly, licensing engines has become a popular route for many game development studios in recent years. By licensing proven technology, many studios hope to shorten development time and reduce costs. Instead of spending months implementing and debugging a core game engine, developers can focus their efforts on the game play balance, artwork, and level design. This has worked spectacularly in many cases; witness Valve Software's critically acclaimed *Half-Life*, based on the Quake engine, and Ion Storm's *Deus Ex*, based on the Unreal Tournament engine.

Unfortunately, it has also backfired in a few cases, most notably in John Romero's ill-received *Daikatana*, which took over 4 years to complete and exemplifies the two drawbacks to using licensed technology. One is that the gamers have already seen the technology used in the original game titles (such as *Quake*

[1]for details, see `http://www.idsoftware.com/corporate/idtech/index.html`

and *Unreal Tournament*), and consequently they have no desire to see another company recreate the same look and feel. That is, licensees are expected to raise the bar in terms of game play depth and overall artistic quality. Games such as *Alice* and *Deus Ex* use the technology successfully to create a completely different feel from the games that originally used their respective engines.

The other drawback to using a licensed game engine is that, given the long development cycle of modern computer games, it is likely that by the time the game is released, the engine will feel dated. *Half-Life* has succeeded largely because of its immersive story and single-player design, but its graphics are noticeably less sophisticated than other games released around the same time. On the other hand, *Daikatana*'s original development centered around the Quake engine, but when John Romero saw id Software demonstrating the 3D hardware-accelerated *Quake II* at E3 in 1997, he realized that his upcoming, already-much-delayed title would suffer greatly without using this hot new technology. He decided immediately to license and begin using the Quake II engine, at the great expense of throwing away many months of development effort spent using the original Quake engine. Thus, the technology licensing program established by id Software can be attractive and cost-effective, but brings along a new set of challenges as well.

The Quest for Balanced Game Play

id Software may have spent years researching and implementing the technology capable of dazzling consumers with richly detailed graphics, providing a smooth online gaming experience, and giving flexible options to the mod and level-design community. But for the average consumer looking for a fun and interesting FPS game to play, the real lasting value lies in the game content, not in the backend technology. The majority of players remain blissfully unaware of the PAK system and the organization of project modules; if the game does not provide an enjoyable and rewarding experience, players will simply turn to other games.

Thus, aside from the difficulties in creating cutting-edge technology, the primary challenge for the *Quake III* designers lay in refining the game play details. This was especially important since numerous magazine previews for *Quake III* appeared in early 1999, calling it the "next big title" from id Software. With such successes as *Quake* and *Quake II* in id Software's recent history, *Quake III* was an upcoming title whose name practically marketed itself. Despite the fact that it would likely be a sure hit, id Software's task in making the game fun to play was even more challenging than before, since they had so many fan expectations to live up to. In addition to the established community of *Quake* and *Quake II* players, there were new players that id was hoping to attract. Furthermore, there were also "professional" gamers who competed in leagues and tournaments with large cash prizes.

Since the game was designed from the ground up to be a strictly online multiplayer game in which players combat other players, id Software was faced with a monumental task of balancing game play details. In particular, level designers

Tim Willits, Christian Antkov, and Paul Jaquays needed to focus their efforts in creating maps that were especially tailored for balanced multiplayer combat and team play. Furthermore, the developers needed to strike a delicate balance with regards to the items, weapons, and rules of the game. On the one hand, *Quake III* needed to have simple game rules and physics, keeping the learning curve shallow so that new players could pick up the game without getting frustrated. Furthermore, the professional gaming community cared most about balancing these weapons and items in order to make game play appropriate for competitive tournaments.

On the other hand, the mainstream *Quake* and *Quake II* fans had strong opinions on the weapons they wanted to see in *Quake III*. This was the result of an unexpected division in the *Quake* and *Quake II* community. When *Quake II* was released, it was assumed that the original *Quake* fans would simply move over to the new game. But there were some serious differences between the two games. *Quake II* introduced the railgun, a long-range, instant hit-scan sniper weapon. Though the railgun was undeniably fun to use, some criticized it for slowing down game play significantly as players took cover and sniped at each from across the level. Another frequent complaint about *Quake II* was that the speed of the rockets from id's signature rocket launcher was significantly slower than the original high-velocity *Quake* rocket launcher. Finally, the lightning gun, one of the favorites in *Quake*, was nowhere to be seen in *Quake II*. With these few but significant differences, it soon became evident that the *Quake* community would not vanish quietly, but rather that id had inadvertently split its fan base into two camps. With *Quake III*, they aimed to unite these camps by carefully blending the best of *Quake* and *Quake II* together.

Finding The Right Balance

id Software took a number of interesting approaches to address these issues. The designers first invited a number of prominent players from the *Quake* and *Quake II* community to their office in Mesquite, Texas to discuss early ideas about the game play and to obtain constructive feedback. Not surprisingly, one of these players was renowned *Quake* and *Quake II* champion Dennis "Thresh" Fong, co-founder of Gamers.com. Fong spent many hours with John Carmack discussing game play details, especially in regards to weapons and items. Some suggestions entailed removing certain items and weapons completely, as they seemed to add little to the value of the game, but in most cases they involved making small changes to either the firing rate or the amount of damage caused.[2]

After incorporating the suggestions made by Fong and others, id Software chose an unconventional but highly effective method for obtaining additional feedback from the entire game community. On April 24, 1999, the company released Q3Test freely to the public. Q3Test was a miniature version of *Quake III* that included only two levels and unpolished graphics, but was relatively stable and, most importantly, playable online. This release was not in any

[2] for details, see `http://firingsquad.gamers.com/games/quake3/`

way intended as a demo, but simply as a "test" of the game play and the new technologies in practice. All over the world, people avidly downloaded Q3Test and gave the developers at id a steady stream of feedback. Over the next few months, the developers incorporated small changes into new releases of Q3Test, tweaking the weapons and player movement, occasionally adding artwork and even a new level. The Q3Test was intentionally left unexpandable and did not contain enough levels to be considered a fully functional game. Hence, id Software was not concerned about giving away too much before the full game release in December. On the other hand, it was a brilliant method of receiving constructive feedback and stirring up excitement about this upcoming title.

Game Play Success

When *Quake III* was released in December 1999, the effort spent on fine-tuning the game play balance paid off. The weapons were varied but simple enough that new players learned to play in a matter of minutes. Furthermore, these players had no trouble finding servers online and were able to join in the fun right away. Seasoned *Quake* and *Quake II* players hailed the return of their favorite weapons: a faster rocket launcher, an improved lightning gun from the original *Quake*, and the classic rail gun from *Quake II*. Before long, there was an established *Quake III* community playing the default game modes (Deathmatch, Team Deathmatch, and Capture The Flag) in addition to developing several new levels and mods.

With *Quake III*, id Software succeeded in pleasing fans from both the *Quake* and *Quake II* communities while keeping the game simple enough for new players, but they also included features that made the game suitable for professional competitive play. There is a Spectator mode in which players can watch a game without actually playing, and a Record feature that enables players to save "demos" of their games. These demos can then be distributed to whomever wishes to view the entire match later. These special features, coupled with a balanced set of items and weapons, led to a rapidly growing community of players who were willing to compete at a professional level. Until recently, *Quake III* was the dominant game used in tournaments held by the Cyberathlete Professional League (CPL), founded in 1997, and there are several sites that contain archives of downloadable demos of CPL tournament matches.[3]

Bugs and Controversies

Even though id Software succeeded in refining the game play to suit the tastes of many new and experienced players alike, some players discovered and exploited two "bugs" in the code, leading to unintended styles of game play and sparking controversy and debate in the community. One of these bugs was a numerical rounding error that occurs in the computation of a player's movement. As players move from frame to frame, these rounding errors accumulate and,

[3]one such site is `http://www.challenge-tv.com/demos/`

in certain instances, allow a player to jump slightly higher and farther than is normally possible. This technique is known as strafe jumping. The majority of the players neither notice nor exploit this bug, but the experienced and competitive players use it repeatedly to run faster and jump higher. In particular, some maps contain special powerups in difficult-to-reach areas; with the help of strafe jumping, they can be reached much more easily, which was clearly not the intent of the level designer. In later patches to *Quake III*, id Software attempted to correct this error, but the community response was largely negative, since the changes made overall player movement feel less consistent. Hence strafe jumping remains in the final version of *Quake III* today and has become a standard technique used by advanced players to move more smoothly throughout a level.

The other bug that players discovered was that a rocket explosion radiates through thin surfaces. This meant that if a player was standing on top of a thin ledge, another player below with a rocket launcher could fire at the *bottom* surface, and the rocket explosion would actually hurt the player standing on top. Although this may seem like a minor problem, it allowed for different tactics in some maps. Traditionally, when two players are locked in combat with rocket launchers, the player who is on a higher platform has an advantage, since she can rain rockets down from above, and the explosion that occurs from the rocket hitting the floor inflicts splash damage on her opponent. Thus, players often strive to position themselves strategically above their opponents. With the "through-floors" bug, the opponent on the bottom can mitigate her positional disadvantage by firing at the bottom of the surface supporting her opponent. This is a classic tactic in a popular map (`q3tourney4`) that appears frequently in *Quake III* tournament matches. Recently, however, id Software chose to fix the bug, and blast damage no longer radiates through thin surfaces. Reaction to this decision has been largely mixed, but there is general agreement that `q3tourney4` may no longer be an appropriate tournament level, since game play dynamics are no longer properly balanced. It is incredible how seemingly minor bugs can affect game play and user reaction so much.

Competition and Future Directions

Minor bugs and quirks aside, the next installment in the *Quake* franchise successfully met many fans' expectations, so one may be tempted to conclude that id Software was once again asserting its dominance in the genre it created: the first-person shooter. This was not the case, however. Other FPS titles, in particular Epic Games' *Unreal Tournament*, were also released in the same holiday season. Like *Quake III*, *Unreal Tournament* was designed from the start to provide a solid online multiplayer experience, and hence the two titles competed directly in the PC gaming market. Compared to *Quake III*, however, *Unreal Tournament* offered not only more levels, but also more variety in the environments depicted in these levels. There were also more unusual and interesting weapons, a significantly better game interface, and better support for team play. Consequently, many see *Unreal Tournament* as offering a more complete package, and it is not surprising that it has far outsold *Quake III* in the market.

More recently, a multiplayer game based on *Half-Life* called *Counterstrike* has emerged as the online FPS favorite, with more active servers than *Quake III* and *Unreal Tournament* combined. Thus, even though the developers at id Software designed a successful and enjoyable game, and despite the fact that *Quake III* arguably has the most advanced game engine of any PC game today, "id" is clearly no longer the only big name in its genre.

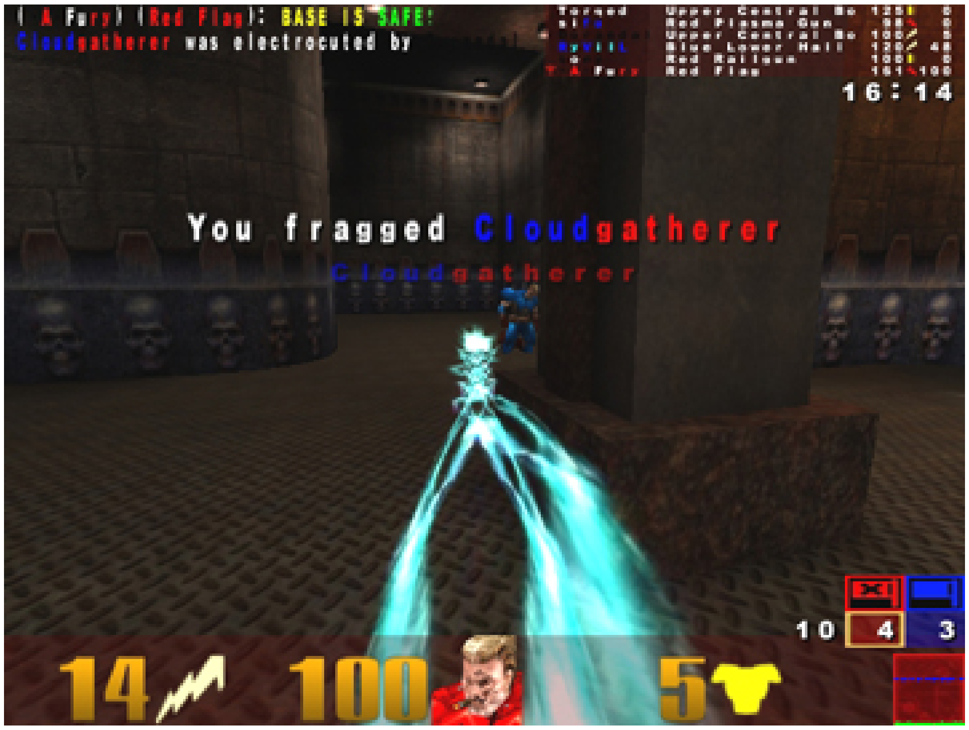
Despite the fact that id Software is no longer the only company creating top-selling first-person shooters, consumers still associate id with cutting edge technology, especially in the rapidly evolving field of computer graphics. The next big title in the works at id Software is *Doom 3*, which will surely contain a multiplayer component, but whose main development effort is currently focused on the single-player experience. At this time, id has not divulged any details about the core game design, but it is clear that the designers will continue to push the technology envelope. On February 21, 2001, John Carmack gave a brief demonstration of the in-development game engine for *Doom 3* at the Macworld Expo in Tokyo[4]. Not only are the models and world architecture more stunningly detailed than in *Quake III*, but in Carmack's words, "we are finally seeing the unification of lighting and shadows in a single environment," meaning that light can be treated the same way across all surfaces, whether they are part of the background world, the characters, or other game objects. This results in a much more realistic and believable world.

When *Quake III* was released in 1999, computers and video cards were not nearly powerful enough to handle such complex lighting techniques in real time, and thus Carmack had to resort to a number of tricks to simulate the effect; even current technology is barely powerful enough to accomplish this smoothly. But rest assured that when *Doom 3* reaches store shelves, the technology required to bring the next level of realism to FPS action titles will be affordable, and id Software will again be producing cutting edge technology. Not only will this appeal to gamers longing to return to the *Doom* franchise, which originally put id Software on the map as a leading game development studio, but the technology will continue to enable the community to expand the default game with new levels and mods. Furthermore, other studios will undoubtedly participate in the *Doom 3* technology licensing program, hoping to take advantage of a state-of-the-art game engine, the entity that clearly continues to drive id Software forward today in the computer game industry.

[4]for keynote in QuickTime format, see `http://www.apple.com/quicktime/qtv/mwtky01/`

Screenshot 1

The author of this paper, a.k.a. **Durandal**, frags an opponent using the lightning gun in *Quake III: Arena*. Many players hailed the return of this weapon, a favorite in the original *Quake*, but absent in *Quake II*.

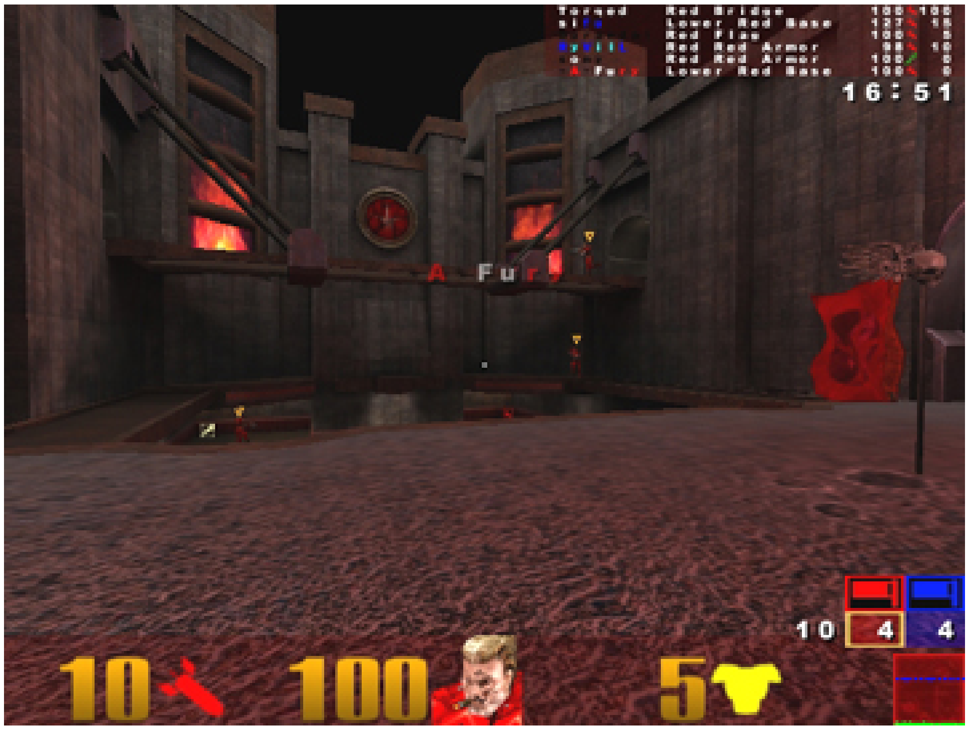


(A Fury) (Red Flag): BASE IS SAFE!
Cloudgatherer was electrocuted by

You fragged Cloudgatherer

Screenshot 2

Several red players in the distance guard the Red Flag in a game of Capture The Flag, one of the team game modes in *Quake III: Arena*. The environment is a custom map titled “Finnegan’s Revenge” made by Threewave (`http://www.threewave.com`).



```
Torqed        Red Bridge        100 100
[illegible]   Lower Red Base    127  10
[illegible]   Red Flag          100   5
[illegible]   Red Red Armor      98  10
[illegible]   Red Red Armor     100   0
A Fury        Lower Red Base    100   0
```

Relevant Official Sites

Quake	`http://www.quake.com/quake/index.html`
Quake II	`http://www.quake.com/quake2/index.html`
Quake III: Arena	`http://www.quake3arena.com`
Unreal Tournament	`http://www.unrealtournament.com`
Counterstrike	`http://www.counter-strike.net`
id Software	`http://www.idsoftware.com`
Epic Games	`http://www.epicgames.com`
Ion Storm	`http://www.ionstorm.com`
Cyberathlete Professional League	`http://www.thecpl.com`