GAMASUTRA
The Art & Business of Making Games

PRINT

Postmortem: Presto's Whacked!

By Michael Saladino

Presto Studio's first and last attempt at multiplayer mayhem.

It's not often in the game industry that a developer is given a chance to create something truly unique that they've been dreaming about for years. When most publishers are filling their portfolios with sequels and movie licenses, Microsoft gave Presto Studios a great opportunity to create an original project for their Xbox console. I have recently come off the two-year project that was *Whacked*, a cartoon-derived, highly frenetic multiplayer shooter. Originally working under the code phrase "Big Toe", Presto began discussing the idea back in the mid 1990s when the company was making the move from pre-rendered adventures to real-time action. It was a long road to find a home for the idea but I was lucky enough to be there when Microsoft gave us the go.

The concept of *Whacked* was pretty straightforward. We wanted to create a multiplayer game that would be addicting to the same degree as *Mario Kart* was on the Nintendo 64. We wanted something simple that anyone could just pick up and play. It needed to have strong rubber-banding so that the game could turn around, rapidly dropping the leader into last in a viscerally spectacular way. And finally we all decided that the genre for this game would be a shooter. Everything else was left up in the air. There were ideas such as a "big red button" in the levels that when pushed would somehow change the global rules of the game but we decided not to write anything in stone until we started playing a basic prototype.

The art team was an interesting mix of talent, spanning from rookies to veterans, but was overall a group with significant pre-rendered experience. This was immediately identified as a red flag since we were creating a real-time game but early focus on this issue kept it from becoming a serious problem and actually led to some interesting design choices. The technology group was headed by Max Elliott as CTO and I as lead programmer. Together the two of us were the drive behind Presto's Sprint Engine, which supported *Whacked*. We kept the total number of programmers small so picking dedicated and experienced talent was critical. Overseeing the entire project was a team of producers who, like the art team, also varied from grizzled veterans to relative beginners. Overall, the project's success would come down to the interaction between our experience and our lack of it.

What Went Right

1. Try before you buy. Presto had been burned on numerous projects by the classic game development blunder, designing too much on paper and not enough in game. Our previous real-time games such as Beneath and *Star Trek - Hidden Evil* suffered immensely from this problem. If you're trying to create something unique, no matter how much experience you have, you can never be sure if that your idea will actually be fun once it's converted into a game. Early gameplay prototyping is critical for unique ideas. If you're simply adding a sniper rifle to a first-person shooter, you can be relatively sure that you'll be able to find the fun in such a weapon. However, if you're adding a freezing weapon that encases the target in a block of ice, is that fun? Is it fun not only to freeze someone but is it fun (in a desperate, panicked sort of way) to be frozen?

A good weapon should be fun for both sides. This was the mantra throughout the development of *Whacked*. Our central tenet from the beginning was to make sure something was fun before we performed full production on the idea. We would start with a simple idea such as a slow moving homing missile that could follow targets around corners. These objects would begin with programmer art, which was part of our plan; if something ugly was fun, imagine how great it would be once our artists got to dress it. So I would release new builds daily, often hourly, with new tweaks to the weapons including speed of projectile, reload time, damage inflicted, and so on. We had a core team of game testers that would play every day and build real experience as to what works and what does not. Everything put into the game worked this way whether it was control schemes, character motion, or level layouts. We created over fifty undressed levels of which only the best thirteen found their way into the final product. Dozens of extra weapons, power-ups, and global effects were created before being thrown away as being "not fun." But for the ideas that developed successfully, we would hand the details of the object over to the art staff who did an amazing job retrofitting design concepts within our gameplay constraints.

Another way that this mantra helped us was with our multiplayer experience. We initially developed the game as a four player split screen experience on a PC. We did not begin designing or implementing any AI for the game until nearly a year after we started the project. Too many times, I have seen a project start AI early in the cycle because the game testers want to play against something. After creating games both ways, I believe that starting with human opponents makes the most sense. When a game is being created, its rules should be allowed to

change rapidly based on human experience. Other humans will be able to adjust to these changes far faster than any AI programmer trying to make his creatures keep up. Also, these high-speed changes to code will quickly create a mess of the AI system. So by using split screen or networking, you can avoid AI until much later in the process when the basic gameplay mechanics will be laid out allowing your AI programmer to see much more clearly the correct path to having his minions behave in the best way possible.

2. On-time deliverables. As anyone that has worked with me in the last four or five years will attest, I have become rabid about not missing milestones. It's simply not done. Sure, maybe a day here, three days there, but the norm in our industry seems to be that slipping by weeks and even months is acceptable. I fight to make sure everyone on my team understands that this is not acceptable. The producers need to be on top of things well before it leads to changing launch dates and missing Christmas sales. When dates are missed, it's not just your team that has to deal with the shift. Marketing dollars disappear when they suddenly don't have their product to advertise. Sometimes penalty payments are made from publishers to retailers due to the losses from empty shelf space. If the game production economy continues to grow the way it has for the last decade, then losing money due to missed milestones will not be tolerated.

With my strong opinion on this issue noted, *Whacked* shipped not just on time, but early by three weeks. How did this happen? The product was solid enough to make it through release certification faster than was originally planned. This is especially surprising when you consider that it was the first online game for Xbox's Live network to pass release certification. Microsoft decided to go ahead and release the product early instead of holding onto it for an extra three weeks until its original Halloween launch date.

This dedication to hitting milestones was more than just our launch date but it covered all of our six-week milestones. A couple of the thirteen milestones ended up slipping by a matter of days but for the rest the builds would ftp to Microsoft the day required. This often required long hours and the classic "all-nighter" to make everything come together but that was part of our desire to have Microsoft see us in the best light. This was our first action game. This was our first console game. This was our first online game. This was our first Microsoft game. With so many firsts, we wanted them to know that we were up to the task, which would hopefully then lead to talks of other games. As some of you may know, those other projects never gelled for reasons that go far beyond this article, which resulted in the closing of Presto Studios. However, for all the reasons that may have caused the company to close, a lack of faith in our abilities by Microsoft was never a reason that was expressed to me.

3. A focused technology effort. So imagine this following scenario: Presto Studios just shipped *Star Trek - Hidden Evil* and has been given *Myst III*. Our artists are obviously going to be busy for quite some time trying to finish *Myst III* in record time. The company managers have decided to hire an external programmer to handle the lead position on *Myst III* because they have worked with him on previous graphic adventures. This leaves Max Elliott and I with some time to spare outside of our occasional rendering engine work on *Myst III* (MMX bilinearly filtered software renderer, rippling water effect, etc…). This spare time gave us the chance to think about the next big thing.

We spent the first six months developing what we called Sprint 2.0, the latest version of our internal engine. This rebuilding of the engine reached into nearly every subsystem including our tool path, lightmapper, rendering core, physics, geometry handling, sound, and so on. We were able to move to DirectX 8.0 and remove legacy systems such as Glide so that they would not constrain our API design. With our collective drive to create, we reveled in the daily discussions about APIs that would best balance and organize general purpose with game specific. Focused on next generation ideas like vertex buffers and the far off "shader", we designed for the future of Presto technology, which in turn outlined future projects.

Once we started getting information about the Xbox, we immediately thought it would be the perfect path to move Presto away from PCs and onto consoles. We had previously determined that a move to the Sony Playstation 2 would cost Presto far more money then we had. The Xbox, by comparison, could be ported to in about a month. (Once we got a development kit, we ported our engine to the Xbox in a couple of days.) This savings was huge for a small company like us. We were able to immediately begin working on cool new features like vertex and pixel shaders instead of spending up to a full man-year porting code to the Playstation's unusual architecture. This gave us another six months of nothing to do but explore the Xbox and the gameplay dynamics of our next game, *Whacked*.

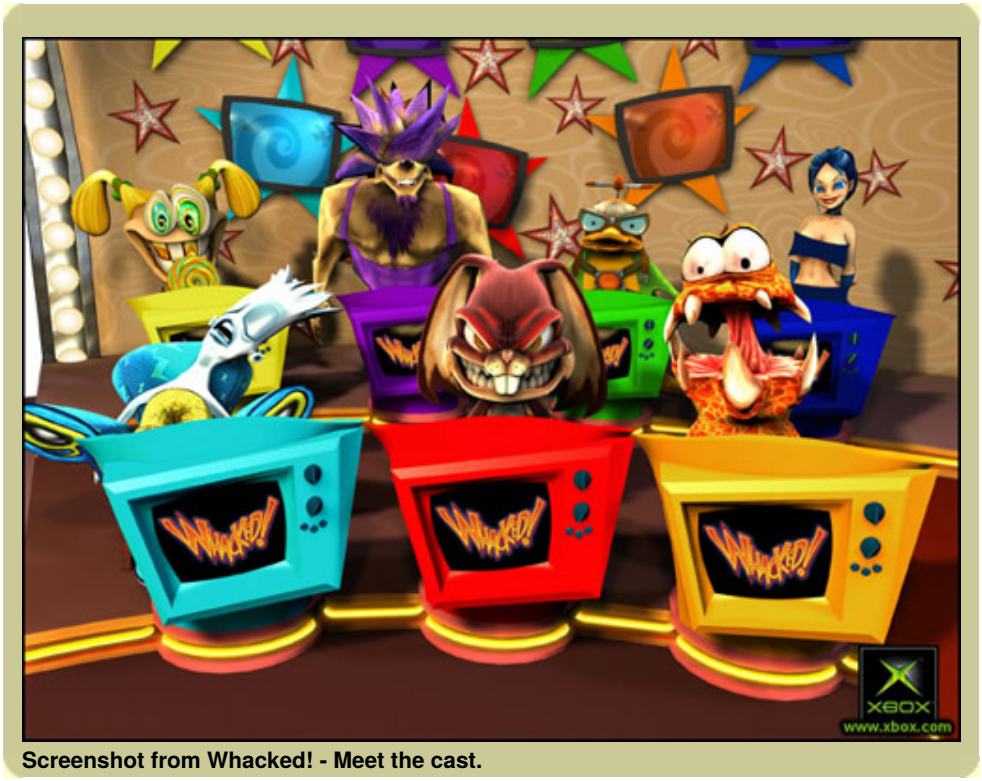
With this full year of technology and gameplay development before art production, it gave the artists a very stable platform in which to build their product. How many times have you known an artist that had to rebuild something because the underlying technology or tool changed? Our goal in the technology department was to keep this to a minimum. We focused on reducing dependencies of the artists on our tools. By giving them technology that was constant, it saved countless man months in production as they could do things right the first time.

4. Extraordinary visual designs. Presto was known as a great pre-rendered art house which really was a double-edged sword on *Whacked*. We had an amazing collection of modelers, animators, and texture artists, most of who had worked on *Myst III*. Obviously, their proficiencies as artists were well proven with that product. However, the differences between prerendered art and real-time art are massive and can be a daunting challenge to overcome for even the most talented artists. We suffered for it, but also ended up opening interesting new avenues for the conceptual direction of the game.

I personally believe that *Whacked* looks unlike any other game on the market. It lacks the classic wrapping texture style that most have come to expect from console games. We decided at an early design stage that our levels were going to be small arenas because they were more fun than large sprawling layouts. (We made both kinds of levels during our gameplay testing stage and small levels were normally the most fun because of the close proximity with other players at all times.) So when we merged the massive texture space of the Xbox along with our artist's abilities to rapidly create large volumes of custom textures, we knew that we wanted custom textures covering our levels. One building would not look like the next. One sign would not be the same as the last. We would create large custom texture sets for all sections of our

arenas giving them a richer sense of space.

I want to mention the work of Dan Paladin specifically because I believe his conceptual work and in-game cut scenes were essential to *Whacked* visually standing out. For those that haven't seen*Whacked*, Dan created Flash animated cut scene "commercials" to be played after games in the single player experience. These commercials are funny, strange, and sometimes complete non-sequitors. However, his understanding of funny makes them all work on multiple levels. Dan doesn't just climb out of the box when creating his art. He climbs out of the box and proceeds to light the box on fire. His commercials alone are almost worth the price of admission to this product. And in addition to Dan, Mike Brown as lead animator and character modeler and Ron Lemen as lead conceptual artist created an amazingly fresh and original product.



Screenshot from Whacked! - Meet the cast.

5. Load times: good code at work. Whenever I write a postmortem, I always have to include one technically specific thing that went right. For *Whacked*, I'm selecting our work to improve the load times of resources from the front-end interface to the back end levels. As late as six months before release, we suffered from painfully slow load times across the game. The Sprint engine was designed for shipping PC games, which are notoriously accepting of long load times. We took advantage of this by making an engine that loaded completely generalized data sets across thousands of files. Every texture, sound, animation, and object existed in its own file. Deep directory hierarchies were used to help organize the data for humans but this in turn slows down file access. Multiple file types were allowed such as TIFs, JPGs, and BMPs, which slowed access as the bitmap loader checked for each type. We had developed the resource system to be extremely flexible for the humans making the game but not fast for the end user that has to suffer through the load times.

Our first strike at the problem was the creation of what we (and many others) call packfiles. This is when you create one large file that in turn contains numerous small files. Our solution included a directory structure at the top of the file to handle the offset jumps required to get to the start of the subfile. This saves file open times which off a DVD and even hard drives can be quite large. We had a global sound directory that contained over a thousand streaming sound files that could play at any time during a game. Without packfiles, each playback required a search down into the correct directory from the root, followed by a directory search through those 1000 files looking for the correct one, and then finally it was opened. With packfiles, we opened one file at the beginning of the game with 1000 sounds in it. Each time a sound was needed, we could simply seek to the correct spot in the already open file and start playback. Noticeable pauses in the game whenever one of these streaming sounds would play completely disappeared after implementing packfiles. Essentially, packfiles are an engine's attempt to handle file access faster than the operating system because it has more knowledge of the file uses.

However, these packfiles weren't our largest savings. The problem with packfiles is that they were still just individual files placed end to end. When we load a level, we would open files during the processing of other files. For instance, say a level contains object A, we would transition to loading object A when it was visited during the level file load. Now say that object A, during its creation, attempts to load a sound and then a couple of textures for the object. We had multiple files opening inside other files, which would still bounce the read head around during the load.

Our solution was what we called a stream file. A stream file was simply a stream of bytes as loaded by the engine. We knew that when you loaded one level in our engine that it would load the same data in the same order every time. If for some reason, it stopped reading from one file and started reading from another, it would do this every time at the same point as long as the data remained consistent between loads. So we created a piece of code in our file system that could be flipped on to dump every byte loaded back into a new stream file. On the next load, you could draw the data from the single stream file instead of the dozens or hundreds of original files needed to define the level. We

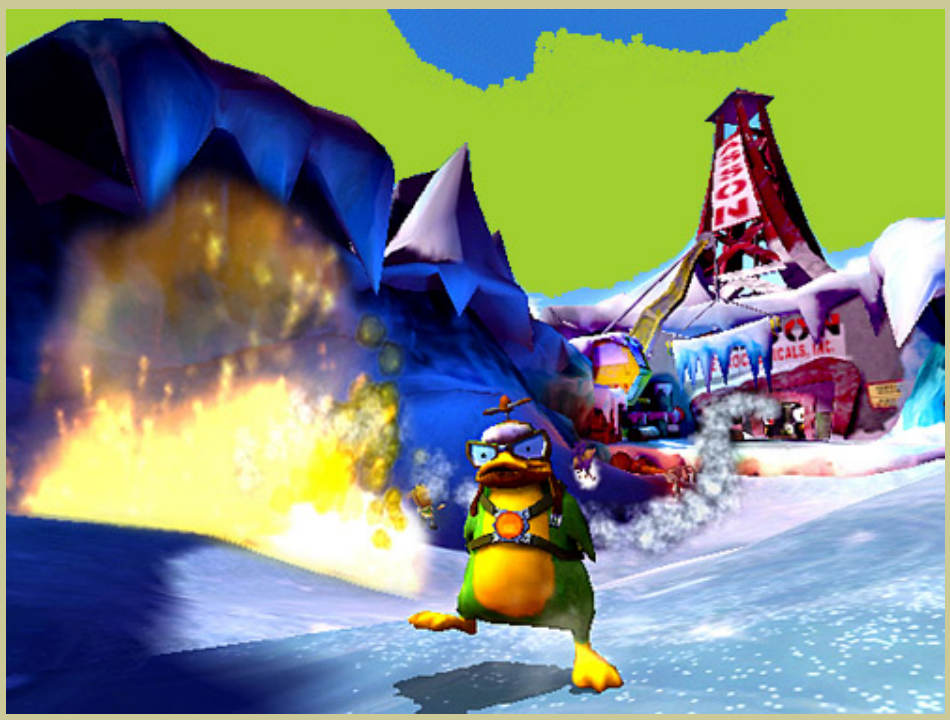
could then do the same thing with our characters and global object data by creating separate stream files for each of them. Now when we go to load a level, we simply read a contiguous block of 30 MB of data as fast as the DVD can stream it. Wrap this code in a multi-threaded IO reader and this dropped our load times from 30 to 40 seconds down to 5 to 10 seconds, which in turn satisfied the Microsoft certification maximum load time.

What Went Wrong

1. Loss of talent during production. As many of you already know, Presto Studios is no more. The exact details of which can be found in the December *Game Developer* postmortem on Presto, but obviously the closing indicates that there were internal issues. One of those issues manifested itself in the loss of two of our three creative designers half way through the project. Their loss to another San Diego company was quite painful for Presto and the *Whacked* team in general. The only remaining designer was the creative director of the company, Phil Saunders. Phil is an amazing designer at the top of his craft; however, the work that remained was far more than he should have been asked to produce. This put us in a difficult situation. On one hand, the design team had always been in control over the production of resources, but now the sole designer did not have time to oversee it all. We could have handed more autonomous control to the production staff but that would often go against Phil's artistic vision of the game. I completely appreciate his passion because I shared it for the Sprint engine; however, in this case it caused bottlenecks at the highest level of production, which in turn would slam the bottom level at milestone deliveries.

We attempted to alleviate this problem by hiring two new conceptual artists to help out Phil. Both new hires, while showing promise, were rookies in the game industry, which meant their ramp up time was going to be long. Also, the mandatory mistakes were going to be made in the form of level redesigns that the original experienced designers would have nailed on the first try. But this is simply the way it is with young talent. With our tight time constraints, hiring more experienced talent from inside the industry would have been significantly more successful.

Since two new hires could not be expected to fill the void left by two amazing conceptual artists, the bottlenecks continued until the end of the level production cycle. One level got so out of control that it fell three months behind schedule. Considering that a level's schedule was three months, this made it take twice as long as any other level. With our total drop in experience across the design team, it was allowed to slip through the cracks. So with little direction, a level builder who had never built a real-time level before was given the reigns. The disaster seemed to drag on forever with it finally being handed over to another level builder to fix in his spare time. A level team completely run by rookies is never a good idea. Always match up a rookie with an experienced person to keep the level development on track.



Screenshot from *Whacked*! - Fire & Ice

2. Imaginary art scheduling. The scheduling of resources was an endless cause of problems. Even ignoring the loss of two conceptual artists, a fundamental problem existed in the scheduling of resource delivery. The design department was always given permission to slip well beyond their internal deliverables because we had an "it's done when it's done" attitude for design -- "You cannot force good ideas."For instance, we needed eight characters for *Whacked* and in order to help design direction we chose to use the seven deadly sins as the foundation for the characters, with the eighth being pure evil. For the sins that obviously worked into a computer game such as lust and wrath, the ideas flowed freely and we quickly found our favorites. However, after months of character design we were pretty stuck on pride and most of all envy. How do you create a likable character whose primary characteristic is that he wants to be someone else? I understand the difficulties of this especially since we were trying to create a group of licensable characters. (We had dreams of *Whacked Kart* and *Toof's Big Day*.) However, "it's done when it's done" is a luxury that we really could not afford. In the end, we finished on time with eight great characters but at what cost to other departments? How many all-nighters in texturing could have been saved if we had delivered an idea sooner to modeling? I know some people don't want to think you can schedule original ideas but in the end that's exactly what we need to do. What makes an artist working for a company different than an artist working for himself out of his personal studio? … Deadlines. Originality on-time and on-budget -- that's the skill a lead conceptual design is paid for.

The scheduling nightmare was probably the worst with our pre-rendered cut scenes. This is different from the Dan Paladin Flash animations. These cut scenes were pre-rendered in 3D Studio Max and totaled nearly an hour of animations. They were done in about four months by two animators and a hand full of prop builders that were simultaneously trying to finish their levels. As any animator knows, that time frame is insane. I honestly do not know why the powers that be desired that much cut scene other than guessing that it came from our adventure game past. This desire led to a bizarre schedule that often resembled a Mobius strip with animators being told to work on something that hadn't even been designed yet. It's a testament to Mike Brown's ability to lead the animation effort that allowed the project to be completed. Further his skill at pruning needless extras and leaving only what is truly necessary saved many of our milestones.

3. Networking. Yes… no… maybe. The first pitch of *Whacked* to Microsoft included a Winsock networking system that I wrote in about two weeks. It was a basic LAN implementation that showed networked *Whacked* could be done by Presto. Microsoft decided that they did not want it included in the final product. I was disappointed but they were the publisher and it was their choice. However, they returned in November 2001 to ask for not just LAN but full online support. This was about six months before our original ship date in the spring of 2002. Max and I saw this as not just an opportunity for Presto to ship a launch Live title but also a personal test of ourselves. We snapped up the chance despite the difficulty the extension would cause. Our launch date would shift to October 31 and we would be held to that like no other launch date we had ever experienced. As our product manager Thomas Zucotti put it, "We broke ourselves into jail."

When we reactivated the networking layer, it was of course broken since it had been allowed to lay fallow for over a year. Not only was the low level transport layer completely PC based Winsock but the game layer had mostly disappeared during the last year. The front-end interface had already been started without networking. And perhaps the most serious issue, all the gameplay interactivity had been designed without network latency in mind. We were going to need help to deliver on our network promise.

We brought in a new programmer with extensive Winsock networking experience to take the networking workload off Max and I, which partially worked. The low-level transport layer he created was amazing along with the dead reckoning system he developed to handle bad network conditions. However, he was what I refer to as an "absolute 40" meaning he did not believe in the unpaid overtime that was apart of the Presto experience. I can accept that some people want to work this way but in our situation it was exactly what we did not need. Max and I were left to work many nights after he left for the day in order to get the game level network code working. In the long run, I'd guess that 60 percent of the total network code was game level, which kept a serious load on us as we tried to finish the game. Eighty-hour weeks were the minimum for months. I personally spent eight months at this high burn rate from December 2001 till August 2002. The addition of the new programmer was critical, but I had hoped for so much more. However, in the end it was Max and I that made the agreement to Microsoft, I can understand if the new guy was not willing to sacrifice himself for the team. However, it was my mistake as lead programmer that I signed off on hiring him without ever asking about his willingness to work hours anywhere close to the other three members of the programming team.

4. UI - Undesigned Interface. The interface in *Whacked* was unloved. No one really wanted to design it, build it, or program it. Perhaps it was because we were focusing on what we considered more important details like networking or level construction but in the end, it was a colossal mess. The first most serious mistake was that no one seemed to believe me when I told them how much work it was going to take. I can still remember vividly the chaos on *Descent: Freespace* caused by the interface system. And when *Whacked* got its online extension, I pushed to hire an interface programmer to take the workload but I was repeatedly told that we could not afford it. So I took the programming of the main interface on my shoulders. Thankfully, our AI programmer Andy Schatz was kind enough to take on the larger online interface.

The design of the interface was complete as you go. It was redesigned multiple times visually and functionally, usually in the game instead of in a more flexible environment such as a web page design tool. The online component suffered because we were aiming at a moving target as Xbox Live continued to evolve underneath us. The actual creation of assets moved from one artist to another; none were particularly good at creating and keeping naming conventions. And as a personal pet peeve, many of the resources were created on a Macintosh and saved in Mac formatted TIFFs, which our engine could not read resulting in missing bitmaps. This was the single worst time suck of the entire project and I sadly have to admit that instead of screaming louder, I just went along and programmed the beast. I should have screamed louder.

5. External Audio Development. This was my second experience with having audio created outside of the company and I have come to the same conclusions. First of all, external music usually works fine especially if it's just a looping WAV file as ours was. However, sound effects are extremely difficult to get right if the audio person is not in-house experiencing the creation of the title firsthand. Most of the sound effects we received from our external source were not correct for one reason or another. Whether it was something simple (yet still sloppy) such as the wrong format or something more serious like a sound having the wrong length for its use, the problems seemed endless. The lack of communication was the main cause as he would receive incomplete instructions for a given sound and then just fill in the gaps himself instead of asking for clarifications. It was a breakdown on both sides. Our contact producer should have made him come in at least once or twice a week so that he knew the levels and objects he was creating effects for. And the audio designer should have been able to identify these shortcomings in the instructions and request better details. If the producer asks for a sound to go along with the swooshing pendulum blades, you should ask how long the duration of one pendulum swing is.

Beyond these communication mistakes, having an external sound designer also created a resource management issue that was never addressed. All of our art resources existed in source control except audio. Audio revision control was usually handled in the form of filenames such as "PendulumSwing3.WAV" which causes a problem when the sound emitters in your level file are referencing "PendulumSwing.WAV" specifically. Without any revision control, we would misplace sounds and be constantly unsure of current versions. Furthermore, the level soundscapes were often completely left up to level builders and myself to sculpt because the audio expert simply was not around. Sounds would come in to be placed with no information about volume levels and attenuation ranges. It was left up to me to perform the final pass of audio leveling, and while I may have some amateur music experience, I'm most certainly not the person that should have been doing it. While the sound designer took considerable pride in his scoring abilities, he never seemed to care about the special effects side of his job and the game suffered.

A Long Road

And so we come to the end of the road of *Whacked* and Presto. It was a rough one but I think the final product speaks for itself. It's strange, funny, and just plain fun to play. Two years later and nearly the entire company still loves bashing each other in four-player free-for-alls. Considering the project was initially supposed to be small, it sure grew into something big with its forty weapons, thirteen levels, Xbox Live play, and nearly an hour of pre-rendered cinematics. And someone at Microsoft really saw something special, because its demo has been included in the Xbox Live launch package, so quite a few eyeballs should get a look.

I hope this look at *Whacked* gives you a little insight in your own projects: past, present, and future. And on behalf of all the production team at Presto, I want to thank the Microsoft Game Studios' Life Studio for taking a chance on our small production house. Extra special thanks to Thomas Zucotti for championing us at MGS and also to Harris Thurmond our test lead. Thanks for finding all my bugs; sorry I put them in. And for every Presto fan that loved *The Journeymen Projects*, *Myst III*, and now *Whacked*, thanks for the love.

Return to the full version of this article