

Bizarre Creations' PROJECT GOTHAM RACING 2

GARRETT YOUNG, MARIO RODRIGUEZ, AND CHRIS PICKFORD |

Garrett's eight-plus Microsoft years span from testing NBA FULL COURT PRESS to exec. producing the PGR series with Bizarre. Mario began testing games at Microsoft as a 2002 University of Miami grad, recently working on PGR2 and RALLISPORT CHALLENGE. Chris has been designing and testing at Bizarre for four years, shipping the PGR series and FUR FIGHTERS.



GAME DATA

PUBLISHER: Microsoft Game Studios

FULL-TIME CORE TEAM: 40

MAX TEAM SIZE (including test, licensing, localization, and other support resources): 102

LENGTH OF DEVELOPMENT: 2 years

RELEASE DATE: Nov 18, 2003

PLATFORM: Xbox

DEVELOPMENT HARDWARE: Pentium 600MHz–2.4GHz machines with 256–1024MB RAM, GeForce 2–4 series and ATI Radeon 9700 Pro video boards.

DEVELOPMENT SOFTWARE: Microsoft Visual Studio .NET, Microsoft SourceSafe, Alienbrain, built-in 3D Editor, Softimage XSI, Araxis Merge 2001, SoundForge

PROJECT SIZE: 37,174 files, 219,538 lines of code, 41GB of data

PROJECT GOTHAM RACING 2 is the sequel to the best-selling Xbox racing game, developed by Bizarre

Creations (Liverpool, England) with production and publishing support from Microsoft Game Studios.

Our two teams rolled directly into production after finishing international versions of PGR in February 2002. We significantly expanded the scope and quality of the combined team, bringing on new artists, programmers, and testers. Our ultimate goal for this project was to create a AAA PGR title for the 2003 holiday season built upon the fundamental strengths of the PGR franchise and innovate in our use of the Xbox's online system, Xbox Live. Given our on-time delivery and the game's 92.4 percent average score from over 70 reviews (referenced from www.gamerankings.com), we feel that we

achieved our goals.

We owe that success to the strength of our people and the clarity of our challenge. Smart, effective, hard-working people are critical to achieving any worthwhile goal, and our two teams had that in spades. Though there were some disagreements and late changes in tactical direction, everyone on the team was always working toward a clear overall strategic vision for the project. A key ingredient to our ultimate success was the strong relationship between developer and publisher: matching Bizarre's design, technical, and artistic strengths with Microsoft's strengths in testing, licensing, usability, creative writing, and production management. Without an extremely high level of trust, we would not have been able to maximize the efforts of each team, and PGR 2 would not have been as strong.

Our hope is that this postmortem can



provide some insight into how we worked together to build PROJECT GOTHAM RACING 2—the things that worked well, and the things we would do differently if we had to do it all over again. With luck, we hope other teams will be able to apply these lessons to improve their processes and avoid some of our pitfalls.

What Went Right

1. Strong early vision for innovation. In our efforts to build on the market success of PGR, we knew it was critical to stay true to and build on a formula gamers loved. We decided to greatly expand the number and diversity

of cars (from 25 to 100) and cities (from four to 11). Our artists invested significant time in researching routes through cities we felt were interesting, recognizable, and fun to drive. As we broadened the scope of the car list, we added new car categories like classics, muscle cars, supercars, and even SUVs. The designers also expanded the Kudos system to increase the reward for skillful driving, adding rewards for taking a “good line” through a corner, drafting, and navigating track sections cleanly (without hitting walls).

But as a new PGR game, we knew we had to continue to push that spirit of innovation. Though Xbox Live was an unproven and unknown technology during our initial design planning, we committed to pushing the online frontier in

PGR 2. We bet gamers would love racing online against their friends, and we decided to incorporate Xbox Live Scoreboards for each race in our game. These interactive high-score rankings allow every gamer with an Xbox Live account to post a race result, and allow the top 10 racers to post their actual race ghost replay for anyone to download and watch (or race against).

There were major challenges inherent in each decision. To build all the new cars and cities, we virtually doubled the size of the original art team, which also increased the challenges in team management and communication. Relying on unfinished technology from external teams created a large bottleneck in our production schedule, as we awaited their



ABOVE. A Porsche Carrera GT edges out a Saleen S7 on the Sydney waterfront.

LEFT. A classic Porsche Carrera purrs loudly on a bridge in Yokohama.

PREVIOUS PAGES. A Ferrari Challenge Stradale scares off tourists at the Duomo in Florence.

deliverables. We chose to accept these challenges head-on, given our vision to expand the scope of the game beyond the original title.

2. Reducing worldwide management.

The Bizarre Creations team included all of our core developers, artists, game and sound designers, and production staff. The Microsoft team in Redmond, Washington included many production and design support staff, licensing managers, a full test team, and the marketing team. Given the importance of our international release, we had localization staff working full-time in Japan, Korea, Taiwan, and Ireland. We also employed 3D art vendors in Australia and England, and translation

vendors in France.

During production, members of the team visited locations all around the world to gather research, reference material, and recordings. We shot thousands of photos and hundreds of hours of video in each city. We recorded real DJs in Moscow and Tokyo. We even made a special trip to the Promised Land, visiting the Ferrari plant in Maranello, Italy to photograph and record engine audio of the Ferrari Enzo before it was released to the public. To borrow an old British saying, the sun never set on the PGR 2 team.

Managing such a global team created many problems in communication and schedule management. Our approach to solving these problems was to actually reduce, rather than expand the amount of management. We built up strong communication channels between all members of the team, removing the communication bottleneck that can occur at the producer level on game projects. All members of the production support staff at Microsoft were empowered to interact directly with all of their peers and other members of the Bizarre team. The Redmond testers and Liverpool developers interacted directly through the bug

database, e-mail, and phone calls. The Liverpool art team worked closely with the Redmond licensing managers on approvals and change requests from vehicle manufacturers and other external licensors. All members of our international localization teams interacted directly with our UI developer.

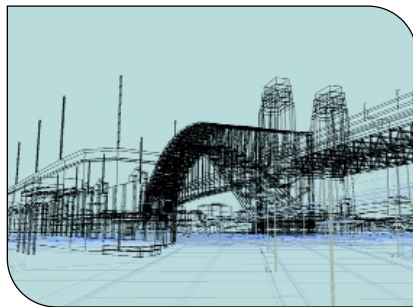
As most teams do, we also planned goals and deliverables for each milestone over the life of the production schedule. However, it was the people we had in place and our open communication channel team-wide that were the greatest contributors to our ability to resolve issues quickly and hit our aggressive holiday release schedule.

3. Proving stable online game-play early.

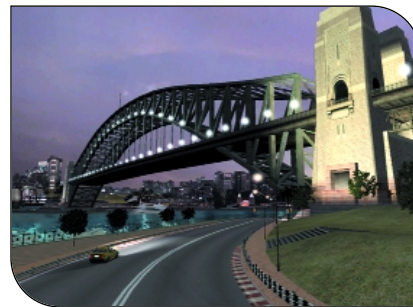
Online multiplayer is frequently the highest risk area for any game, since multiplayer features can be the hardest to implement and require significant optimization and tuning. We addressed this problem early by implementing the bulk of our network code, physics optimizations for interpolation of car speed and trajectory across all boxes, and support for all Xbox Live features by early April, 2003, five months before release. Our overall multiplayer execu-



LEFT. A photograph of a bridge in Sydney.



MIDDLE. A wireframe version of that bridge.



RIGHT. An in-game screenshot of the bridge.

tion was relatively smooth, with only performance and voice issues to resolve during our final code optimization phase.

We were able to achieve positive results by hiring a strong network programmer, working closely with the Xbox Live team to fully understand the new technologies as they were being built, and synchronizing our implementation and test processes. Implementing a major set of Xbox Live features to the requirements of the Xbox certification team required educating our team immediately. This investment paid off later by allowing us to quickly implement new APIs as Xbox Live 2.0 features such as stats with attachments (to support upload of ghost replays) were completed late in our project schedule.

Testing's early involvement quickly identified lag and interpolation problems, and allowed us to troubleshoot new Xbox Live features such as friends, voice, and Scoreboards during initial implementation, which allowed the development team to stabilize new features earlier.

As a result of proving multiplayer stability early on, we were able to eliminate a major risk to our production schedule and focus our efforts on other gameplay, tuning, and polishing issues during the endgame.

4. Driving user feedback into product design. Designing a game under tremendous time pressure can often create a myopic approach to interface design and play balance. The team is so close to the game during development that objective evaluation of the inherent challenge and usability becomes difficult if not impossible. Schedule requirements limit valuable iteration time, and members of the team cannot represent the diversity of skill across

the spectrum of all end users. The theory “a user is only a ‘new user’ once” presumes everyone in your audience will be willing to struggle through a confusing interface and unbalanced difficulty levels to experience and enjoy your game vision.

We had success in addressing these challenges on PGR 2, but not without serious investment of people and iteration time over the last few months before release. We spent literally hundreds of hours hand-tuning each car, to balance realistic handling with ease of use. We used the Microsoft usability and consumer playtest labs extensively to seek out feedback from racing gamers, and see first-hand their initial experience and difficulty navigating our interface, understanding the Kudos system, and achieving the micro-goals of each race mode.

Aside from finding and fixing all functionality and content bugs, our greatest efforts during the endgame were put towards balancing our gameplay difficulty. Almost everyone across the teams in both Liverpool and Redmond—and many others not already on the PGR 2 team—spent time providing play-balance feedback on specific challenges, each of the five difficulty levels, and the overall rate for unlocking cars. Considering the scope of the game, the size of the audience, and the concerns we had about exposing “golden paths” (shortcut cheats to high scores) in the scores and ghosts posted to our Live Scoreboards, we were very happy with the results achieved during our intense and collaborative tuning period.

5. Effective licensing management. Real-world authenticity is a core characteristic of the PGR franchise: real cars, real cities, real radio stations with real DJs, and real music from real bands. Unlike movie makers, game mak-

ers are required to get contractual approval with the owners of each logo and likeness before releasing it in a game.

With 11 cities (including a real-world race track), over 100 cars, unique DJs for each of our 33 radio stations, and over 300 songs, this was a monstrous task. We employed a team of five licensing managers over the course of the entire project to own and execute on this task, establishing contacts and maintaining relationships with all appropriate parties, facilitating review of all in-game assets, and working closely with legal counsel to close down each contract.

The licensing team's contributions were vital. Not only did they enable our artists to fully realize the authenticity of each city, they also secured the appropriate rights, without which we would not have been able to include Ferraris, Porsches, BMWs, or any of the other real cars in PGR 2.

What Went Wrong

1. Synchronizing production deliverables worldwide. As mentioned previously, managing the contributions of our worldwide team was a great challenge, and we feel that process went well overall. However, as with any huge challenge, there were major problems that surfaced in some core areas of production.

Our original plan for addressing the problem of recording, processing, and implementing source material from our worldwide car list was to distribute the workload and take advantage of our global resources. This ended up causing more problems than it solved. Crucial implementation and tuning time for each car was sacrificed, as all cars were not



A Porsche 911 GT1 burns rubber on the streets of Chicago.

recorded to identical specifications and many cars were recorded very late. Additionally, an essential member of the sound design team moved to California and attempted to continue to fill the role part-time, adding friction to an already weak process. A related problem also occurred with the DJ scripts we recorded in each local city, as long contract and recording schedules delayed implementation beyond our desired dates.

This failure taught us two lessons. First, we needed to better synchronize content creation and implementation in the future. Second, we were reminded of a challenge all Microsoft-published projects face: synchronization of the implementation and test teams.

Though all the developers, artists, and designers at Bizarre were working in real time within the same bug database as our Redmond testers, licensing managers, and international localization teams, we did experience many setbacks. At 4GB, our transfer time for builds was substantial, and we were forced to change file transfer tools three months before release. Some obvious bugs in Redmond were difficult to reproduce in Liverpool, as the testers verified builds that were at least a full day

behind development. The 5,000-mile distance between teams greatly increased the risk associated with any last-minute file changes at the end of the project.

Though we made some great strides to better synchronize implementation and testing, our challenge moving forward will be to hit our functionality and content deadlines more effectively, increase the stability of the build process, and to increase the scope of smoke tests in Liverpool before builds are sent over to the Redmond test team.

2. Design took too long.

Incomplete design had the single largest impact on the slide of deliverables throughout our project schedule. Though we all understood and agreed upon the fundamental vision for the franchise, we started our PGR 2 design document very late, and we were still updating this document after feature freeze and E3 as new ideas cemented. We ended up overhauling many elements of our design several times, such as the user interface, Kudos reward values, and overall game structure.

There were many critical factors that extended our design phase. Our designers were spread too thin early on, and we

pursued many different paths and game modes before hitting on a concrete plan. Martyn Chudley, the creator of the PGR franchise and head of Bizarre Creations, played a critical part in stepping in to narrow the focus of the overall game design in early 2003.

Iteration on the handling characteristics of each vehicle proved incredibly challenging, and was completed very late in the schedule. As with any simulation-quality racing game, each change in vehicle handling has a significant knock-on effect on each car's usability, the performance of the AI, the class and competitive categorization of each car, and the difficulty-level setting for each race.

Quadrupling the number of cars from PGR to PGR 2 more than quadrupled our vehicle-tuning time, as the broader scope of vehicle content required far more diligence, testing, and tuning between individual vehicles and classes. We underestimated the initial scope of this effort.

To combat these problems we expanded the size of the test team, pulled in additional designers from other projects, and hired a small team to execute specifically on the play-balancing task. Moving forward onto future projects we will seek out additional



LEFT. The Bizarre Creations PGR 2 development team. **RIGHT.** The Microsoft Game Studios PGR 2 production and publishing team.

solutions, such as fleshing out key sections of the design document earlier, prototyping new gameplay elements in offline builds, and proactively scheduling our extensive play-balance efforts. Iteration is a natural part of game development and we're pleased with the results, but getting there was no easy task.

3 • Relying on new technologies from external teams. PGR 2's design called for online features to be integrated within almost every area of the game. These features required APIs that were still in development by the Xbox Live team as our schedule progressed, raising many significant obstacles and time constraints.

While our Scoreboards were an extension of the system shown in the 2002 Xbox Live starter kit's MotoGP demo, we utilized them to a far greater extent than on other titles. Coupled with new Xbox Live features, such as stats with attachments (such as ghosts), and the high volume of Live users we anticipated, we knew that we had placed a big bet on bringing these emerging technologies to a usable and stable state. If any piece failed, the system would have failed. Implementing the new attachments APIs for uploadable and downloadable ghosts presented us with unpredictable problems, as we were pioneers in this space. The Xbox ATG team was extremely supportive, but they were also pushing hard to complete features for their own deadline.

One example of a problem we should have been able to avoid was optimization of our interaction with the Live Scoreboards. During research done by the Xbox Live team late in our schedule, they found our code was making far too many calls to their Scoreboard servers, a capacity problem their servers would not have been able to handle after release. Though we fixed this problem, it raised significant production fears at the end of our schedule.

In relying on critical technology from external teams, we have learned the importance of allocating an adequate schedule buffer to accommodate unforeseen problems, maintaining strong communication with all dependent parties, and gaining a deep understanding of the technology.

4 • Stability of the ghost replay system. In supporting a feature where any Xbox Live user worldwide could

upload a ghost replay, we needed to be able to guarantee each ghost would be a perfect replica of the actual race result. The critical nature of this feature caused us to dedicate significant attention from our test team.

The replay subsystem served as the underlying framework for recording the ghost data. This legacy system was both complex and difficult to consistently debug. One late night two days before RTC (release to certification), our testers were in heated competition, challenging each other's high score ghosts on one



A BMW M3 shines through a rainy night in Edinburgh.

of the arcade cone challenges, with several lead changes over the course of a few hours. As one score became very difficult to beat, one of the testers noticed the total Kudos shown in the ghost replay did not match the value on the Scoreboard. Despite all the months of non-stop testing, a strong game only hours from shipping still had a very critical bug somewhere in the replay system.

Competitive gameplay is a very valuable part of the testing process during the endgame—this was how gamers were going to be playing our game! Although we fixed it, we could have found that bug earlier, and in the future we will also push to create more automation for critical areas such as this, including hooks for specific test scripts, boundary, and stress conditions. We'll never be able to perfectly emulate the gameplay of millions of gamers, but by prioritizing our focus, expanding our automation suite, and increasing the size and scope of our endgame “bug bash” efforts to broader internal groups and teams, we'll be better armed to find and fix all show-stopping bugs before release.

5. Build process and source control. We had a substantial number of assets to manage during content creation at Bizarre Creations, including over 100 cars with 3D model and dynamics/handling files, 11 constantly evolving

city models, and over 8,000 audio content files. This caused tremendous confusion at the end of the project, as our processes were not originally planned to handle this scope of code and content.

At the beginning of the project we had multiple teams uploading to one game image. This led to significant content incompatibility problems in the build, such as cars using incorrect engine audio, city tracks without track-side barriers, and old bugs re-appearing as old content over-written as new. We planned to solve this by splitting up the original game image into separate images for city data, audio and radio content, car content, and all source code. We also planned to create a unique test image, where all content would be copied before release to the Redmond test team.

However, splitting the process this way ended up causing more harm than good. During the endgame, as the Bizarre team was doing builds every few days, the test image had to be manually updated frequently. To ensure consistency and speed, the team created a step-by-step process and set of batch files, to be run in a specific order each time. With a game so large and server space at a premium, we could not use Alienbrain or another file management package. We were forced to dedicate one person to this manual drag-and-drop process for creating the test image.

Disasters began to strike as the build

process began to take longer. Everyone was working very long hours, late checks were made after build smoke tests, and additional steps were added in managing retrieval of assets from multiple game images, complicating the build process. The test image would often be pulled together and posted to the secure FTP site, only for the Redmond test team to find the build would not run when they came in the next morning, losing a day of testing on the latest bits.

Moving forward, the Bizarre team will stick to two game images—one image for the team to post all code and content, and another image for all tested, shippable content. With less moving parts, we expect the process to go more smoothly on future projects.

Final Lap

In the end, we are all very proud of the results we were able to achieve in PGR 2, and we hope gamers are too. As a team, we were able to deliver on our vision and critical priorities for the game. We were able to increase our quality bar by maintaining a strong balance between building upon the core fundamentals of PGR gameplay and breaking new ground in online multiplayer and scoreboards. We were also able to deliver the game to gamers on time.

However, no project is perfect, and we certainly had our share of hurdles to overcome, many self-imposed. We grew a lot as a combined Bizarre Creations and Microsoft Game Studios team between PGR and PGR 2. Our challenge will be to continue that growth in the future, to learn from the success and failures of our past, and to work together to overcome future problems as they arise.

Looking back, the key to our success was the team involved in bringing the game to life. PGR 2 was built by smart, hard-working people working together effectively with a high degree of trust, open communication channels, and a clear vision and goals. Easy things to say, but the magic was in the execution, as it will likely continue to be in the foreseeable future. 🚗