GAMASUTRA
The Art & Business of Making Games

PRINT

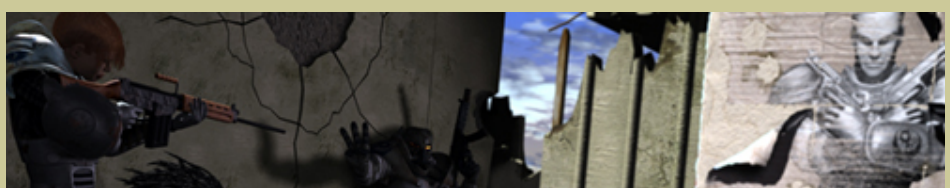
# Postmortem: Micro Forte' s *Fallout Tactics*

By tony oakden

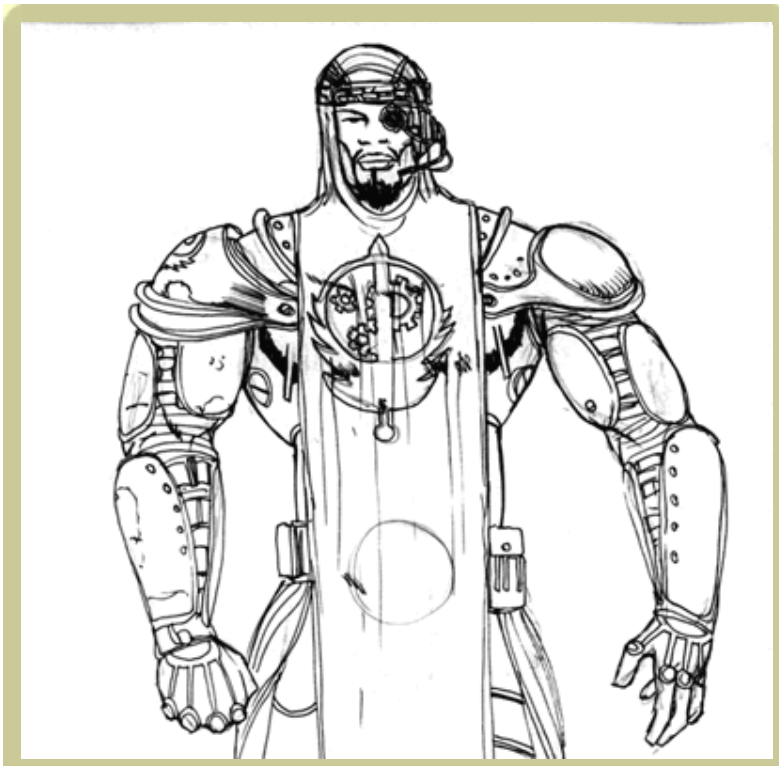
In May 1999, John De Margheriti, our CEO, visited Interplay Productions with a pitch for an isometric scrolling shooter called *Chimera*. Phil Adams at Interplay was impressed with the quality of the demo, but was not interested in the game itself. However Brian Christian, who was the head of Interplay's 14 Degrees East studio, saw enough potential in the demo and in particularly the Phoenix engine to propose the idea of doing a tactical combat game set in Black Isle's postapocalyptic *Fallout* world. The new game would continue with the themes explored in earlier *Fallout* games but with an emphasis on tactical-style combat rather than RPG gameplay.

After a rough design document was put together by Ed Orman, our designer, a design meeting was set up and key members from Micro Forte visited the Interplay offices for five days to thrash out the initial design specification. Much was discussed, including doing the game as turn-based only, and where the game would fit in the *Fallout* timeline. We would stick to the original prerendered appearance of *Fallout*, as we felt that current 3D technology would not give us the detail that we wanted to show in the environments. The game would run in much higher resolutions than other prerendered games, and our engine would make extensive use of alpharing for antialiasing and other effects. Brian Fargo, Interplay's chairman and CEO, stipulated early on in the design process that one of the major components of the game would be the inclusion of a multiplayer game. A ship date was fixed so that the game would be on the shelves for Christmas 2000, and a series of milestones created. The first major demo called for us to show a fully playable multiplayer game at E3 2000, and this became our initial focus.

Over the next 15 months we worked tirelessly until the game finally went gold on March 5, 2001 -- approximately four months late. We ended up creating 21 core missions (some of which were so large that a vehicle was required to get around the map), 30 unique encounters in the flavor of the original *Fallout* game, hundreds of different combinations of randomly generated missions, five highly detailed bunkers for the player to visit when re-equipping the squad, five different types of vehicles, and a whole new race system. On top of this, with the help of Dan Levin at Interplay, we produced an interesting and involving story that progressed through the use of prerendered movies, hand-drawn illustrations, and in-game cutscenes. The game includes over three and a half hours of in-game voiceover plus 30 minutes of prerendered movies. There are more than 26,000 separate tiles and more than 300,000 separate frames of animation. Our animations are some of the best I have ever seen in the industry. We included real-time lighting and a novel shadow-casting routine, which really helped to disguise the tiled aspect of the game.




**Bos soldiers ready to ambush unsuspecting raiders.**



**A concept sketch for a BoS scribe.**

It was a rocky ride. The first few months were spent bringing the team together and getting a clear idea of what the game was going to be. Because the lead programmer from the original design left, we had to work extra hard with 14 Degrees East to convince them that the project was still viable. They, in turn, had internal problems convincing the powers that be that this was going to be a successful game and that we would deliver it on time.

Along the way we got into arguments with the *Fallout* fans about what the game should be and often felt that we would never be able to produce a decent product. We got through it, and we all learned a hell of a lot. There are a lot of things I would do differently in retrospect, but changing the people involved is not one of them.

---

**What Went Right**

It always annoys me when I read a game Postmortem and the author sings the praises of the team and game. I think we made a lot of mistakes, but we had an incredibly difficult job to do and we got it done -- eventually! We must have done a lot of things right; the following are just a few of the things we did well.

**1. Game-creation tools**. Right at the start, we recognized the importance of developing decent in-house tools and invested a substantial amount of time very early in the project to producing them. By January 2000 we had a level editor in place, and the artists could start placing tiles. This became the cornerstone of our level-production process. Over a period of a few months we produced a suite of tools, including a level editor, tile editor, and sprite editor, that we continued to develop right up until the end of the project. Because we did this early, the artists were able to start producing content for use in the game right from the start.


**Level Editor. This was the main development tool used for creating the game. Tiles could selected using the file browser on the left of the screen and dragged into a scene before being positioned in 3D space.**

An unexpected benefit of this was that towards the end of the project, when the artists had finished creating the tiles for the game, they were also experienced in using the tools for placing them into the levels. At that point we needed to generate a substantial amount of content very quickly and achieved this relatively easily because we had good tools and a large team of people, all of whom could help with creating levels. The situation was further improved because it was easy to cut and paste within maps and between maps. We could have put more effort into our entity-editing tools, but on the whole I was very pleased with the tools we created compared to tools I've seen and used at other companies.

**2. Working closely with the publisher.** I spent a lot of time dealing with Brian Christian at Interplay. If he was happy, I was happy -- if he was pissed off, I did not sleep. I visited Interplay about once every three months for a one-week design and planning session. My reception at the first meeting was very cold, and I got the distinct impression they were not happy with the way the project was going. I went through the progress we had made and spent a lot of time reassuring them that we were addressing all of their concerns. Eventually, we agreed to have regular meetings either over the phone or with me traveling from Australia to L.A. During this time we were able to recap on what had gone right and wrong. Near the very end of the project, I went to Interplay and worked there for the last week. This way, I was able to see first-hand the problems their QA department were having and interpret them. I was able to fix minor problems with the dialogue and insert new data into the game on-site to save a lot of time. I think our relationship with Interplay was excellent, and I enjoyed working with them very much.

**3. In-house QA.** I was initially skeptical about the benefits of having a formal in-house QA process, but by the time the project was completed I was converted. The programmers were able to release versions to our QA first, and they in turn would identify key issues and request fixes before Interplay got hold of the versions. Because our QA was in the same office, there was a very healthy dialogue between QA and the programmers and mission scripters.

Toward the end of the project, the in-house QA really saved the project. The time difference between Micro Forte in Australia and Interplay in L.A. meant that the turnaround between Interplay's QA reporting bugs and receiving new versions was too long. Because our team was checking the bugs in-house and talking to the team directly, we saved literally days of QA time. We were lucky to have an experienced QA manager in the form of Jason Sampson with us for nearly the entire project, and he established most of the procedures which we would later rely on to get the product out.

**4. Excellent team structure and discipline.** Initially the studio lacked a strong structure. I came in as a lead programmer and Parrish was the lead artist. John De Margheriti felt that I should be promoted to producer. Karl was promoted to the position of lead programmer, and Ed was the lead game designer from the beginning. Paul McInnes, lead designer from our other studio in Sydney, made some very strong contributions to the overall level designs near the end of the project. The leads took complete control for what happened within their teams, and I only dealt with the team indirectly through the leads. This worked extremely well once we had got over a few initial teething pains.

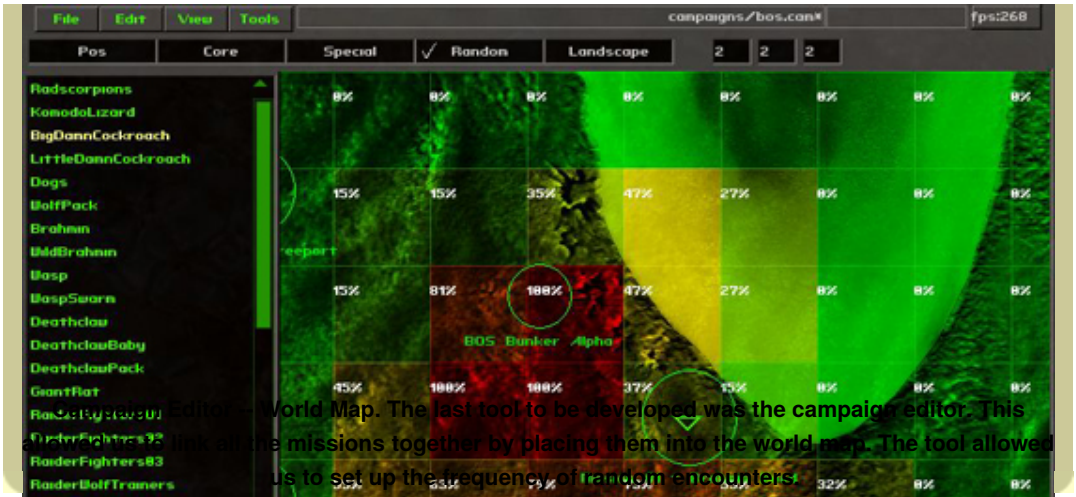
We had a large art team. This could have caused problems, but Parrish was excellent at maintaining the schedule. By the time we finished, the team had grown to 13 artists. We were very lucky in that Micro Forte is allied with the Academy of Interactive Arts, a training facility in Canberra that specializes in training 3D artists. From there we were able to recruit some of the most talented artists I've ever worked with.

The programming team, led by Karl, did a fine job under difficult circumstances. We had the good fortune of having a well-designed code base. A decision was made before I joined the company to largely rewrite the existing Chimera engine, and ultimately it paid off. The programmers were able to spend time putting a solid C++ architecture in place on which they were able to comfortably build new features. I was not involved in the design of the engine, which is one of the major reasons why I was never conformable in the role of lead programmer, but I know from the very small number of crash bugs reported by QA that the strategy adopted by the programming team worked. By writing from scratch they were able to get everything in place pretty much how they wanted it. I'm not a great fan of rewriting stuff for the sake of rewriting it but, I have to admit it worked in this case.

Game design was our only real weakness, but I think a lot of our problems came from muddled design goals that may have originated largely from outside the studio. There were no major personality clashes during the project, and everybody remained very motivated right to the end. I did ban Counter-Strike in the last three months of the project, but by that point we all knew what had to be done to get finished.

For the final few months there were a lot of long hours involved, but for the bulk of the project we managed to keep the weekends free and only had to stay late when a milestone was due to be released. Compared to other projects I've worked on, our working hours were very reasonable. I think the fact that we were still able to deliver in 15 months is living proof that working ridiculously long hours for a sustained period is not the way to get projects out on time.

**5. Well-defined milestones and demos.** Having a short development cycle and clearly defined milestones certainly helped us to focus, to say the least. E3 was our first deadline, and we were due to demonstrate a working multiplayer version of the game. This was a major achievement for us. It also provided a great boost to everyone's morale. After E3, though, we had a couple of rather weak milestones. During this time we did drift a bit until we were due to deliver our first single-player demo. It was obvious at this point that, in a space of about three months, things had gone wrong, and we concluded that we would not be able to get the game finished by October.




Editor -- World Map. The last tool to be developed was the campaign editor. This ink all the missions together by placing them into the world map. The tool allowed us to set up the frequency of random encounters.

After this we reass ject and re-mapped our milestones. Once we had established attainable milestones, we were able to refocus. Our most important milestones became the single-player demo which shipped just before Christmas 2000 and the multiplayer demo which shipped at the end 2001. Both these demos could have been very disruptive with regards to completing the final game, but we used them instead to get critical stuff finished. The single-player demo was the first time we used a focus group to test the whole combat system, an were very encouraging. The multiplayer demo gave us the chance to test the Game Spy support and get all the synchronizatio shaken down. Both demos were very successful in raising the public awareness of the game.

---

**What Went Wrong**

It might seem that given the scope of the game and the fact that we got it all done in 15 months that nothing could have gone wrong. Actually, plenty of things went wrong. We more or less completely redesigned the entire mission set in the last four months of the game's development. These are what I think were key blunders.

**1. Incomplete design documentation and unrealistic design goals.** Although our design specification gave a clear outline description of the game, it did not go into enough detail. This became very apparent in the final stages of the game when the level builders and programmers needed to know exactly what had to be done. There were problems with NPCs and PCs not being designed in sufficient detail, experience progression not being mapped out, not knowing how the quartermaster would behave, no information on how equipment should be released into the game, and other missing details. There were many areas that just were not designed at all. This meant that a lot of design had to be done on the fly by the people doing the implementation. I believe a flexible design is essential, but key design issues need to be nailed down before the programmers or scripters have to implement them. In our case, people outside of the design team ended up making the design decisions in a rather ad hoc manner. In any event, it went pretty well. I do not think there would have been any point in trying to design these things far in advance, but they should have been ready to go well before they needed to be implemented.

**Character Stat GUI. We wanted to keep as faithful as possible to the original *Fallout* look so we copied the character stat GUI right down to the Pip boy illustrations for each stat. The fact that we had a very complete library of UI functions in the game engine made creating this GUI much easier although we really needed to have put some effort into GUI design software.**

The game's design goals were another, more serious problem. When I joined the project I had the distinct impression that the game was going to be a turn-based RPG. As the game progressed, it became clear that this was not the case, and we started to move farther into the world of tactical combat and ultimately to real-time gameplay. The situation finally became intolerable in August 2000, when we attempted to produce our first single-player demo. The game absolutely stunk! It was obvious to all who played the game that this just was not working and at this point Brian Christian phoned my CEO and told him that the game did not play.

Our CEO stepped in and worked with us to help define what the game was actually about: tactics. He worked with our lead designer in defining key tactical elements that a level designer could pick and chose from in order to help design interesting tactical situations. We realized that the RPG element had to be peripheral to the game, and we concentrated instead on setting up carefully designed tactical situations for the player to solve. We started with just the opening section from the second demo mission and worked on that until it was fun to play. We gradually expanded this to the whole level. Over a period of a few weeks, we were able to turn the demo into a fun experience. By the end of August we had a finished level that was great fun and challenging to play. But the cost was further delays in the release date. The big problem was that by this time we had already started building game levels based on assumptions about how the game would be played, so most of this work was wasted. Combining this with the fact that the text had to be ready for localization and recording of voice-overs meant that we were unable to make all the changes we wanted. Eventually, we had to completely rewrite a large percentage of the missions with a substantial cost in terms of time.

**2. Poor asset management.** We used CVS for the source code, and this worked well, but we never came up with a good system for the art assets. Artists worked locally on their machines and then uploaded the finished renders onto the server. Level designers worked directly from the server, and at times this could be very slow. We had an ad hoc series of temporary and final directories where people kept work. Sometimes stuff got overwritten; normally it didn't. The other side of this was that we ended up taking a huge amount of server space up with multiple copies of redundant data. We ended up very short on file space, but we managed.

For future projects we will have to come up with a better system. It's a topic that is being discussed in many different places, so I guess a lot of other people in the industry are having similar problems.

**3. FTP.** Being located in Australia did not seem to be a major problem at the beginning of the project. We were able to talk to Interplay on the phone in the mornings, and if we had a substantial amount of stuff to send we could courier it over and it would take a few nights. It was not until we were due to deliver the first demo that we began to realize just how serious the problem of getting updates to Interplay was.

The fastest I could courier a disc to America was about four days. Using FTP was the only alternative, and FTPs from our location only managed speeds of about 5KB/sec. This meant it would take about two days to upload the complete game to Interplay, and that made it impractical to send the whole thing every time we produced another version. We had to send incremental updates and hope that the version that Interplay was testing had kept pace exactly with the version we had at Micro Forte. Up until a few weeks before we shipped I was still not sure how this would work. In the end, we had to lock our sprite and tile data down early and send that in the early version. Again we managed, but it was a constant source of concern for me, and we still had problems in the game that I am sure were caused by the versions getting out of sync. One of the main reasons I chose to spend the last week of the project at Interplay was to ensure that if things went badly I would be available to make sure that the version they had was sound. I made changes to voice-overs and sound effects at Interplay in parallel with those made by the development team at Micro Forte. We could have used a third-party patch program, but such programs can be very slow to execute, and towards the end of the project we were sending versions once or twice a day.

I think Australia is a great place to develop games and the FTP problems we had are actually not significantly worse than the problems any developer has when dealing with a publisher. The use of in-house QA really made a big difference here. Still, I think a faster FTP link is in order for our next game.

**4. Unrealistic deadlines.** Originally, we had 12 months from starting the design to delivering the gold master. If you take off a month for initial design and a month for QA, that only leaves 10 months for the development of the game. We made our first deadline, E3, but after that we were constantly slipping.

Our original art estimates were based on Interlay providing the model data for most of the in-game graphics, such as all the characters and creatures that had been created in the original *Fallout 1* and *2*. Once the project was started, Interplay informed us that it was not possible to retrieve the art from the tape backups they had. This meant that our original art schedule was blown away and we needed many more artists. We negotiated for more time and cash, but only received more money and no additional time. Fortunately, through our relationship with the Academy, we were able to hire six more artists quickly who were able to come right up to speed. Thanks to our illustrator Tariq, we were able to re-create from scratch all the creatures from the original series and the overall *Fallout* look.


**Exploding more death this time with explosions thrown in. We pre-rendered all our explosions. This scene shows a player accidentally detonating the explosives which raiders have placed on the generators. The object of the excersise should have been to save the lives of the civilians who are currently being cooked.**

We tried to stick to the original schedule, but it became obvious that we could not possibly deliver in October. Interplay agreed to extended our deadline to February 2001. Micro Forte had made considerable financial investment in the game itself, on top of what was negotiated with Interplay. At this point, we had already invested in extra artists to try to meet the original deadline, so not only was our burn rate higher than we had planned, but it went on for longer. At the beginning of the project, the QA team and many of the 3D artists recruited for the project had been advised that there was no guarantee of ongoing employment after the game shipped. We tried to keep everyone employed, but ultimately we were left with a situation where we had no choice but to let half the art team and the QA team go upon completion of the game. We finally delivered the game in the first quarter of 2001, but even with the extra four months, it was a close thing. The game did come together incredibly fast in the last few weeks, but it was very tense for everyone concerned.

When I first joined the project I produced a Microsoft Project document for the game. The Gantt chart still hangs on the wall downstairs as a testament to the hopelessness of our original goals. Two weeks for pathfinding and six weeks for AI seem hopelessly unrealistic now, but at the time we had no choice if we were to meet our original deadline.

Another major headache for us was that Interplay wanted to ship localized versions simultaneously. From a technical point of view this was not a problem, and it actually helped us to design the game from the start so that we could plug in the voice-over and text without code changes. The problem was that in order to get everything ready in time to ship, Interplay had to start recording and translating dialogue months before the game was due to ship. With our tight deadline we still had a lot of unknowns in terms of what would and would not work right up until the game went into QA. Unfortunately, by the time we were able to focus-test and find flaws in the game, the voice-overs had already been recorded and we were effectively prevented from fixing some basic flaws in some of the missions. This was a shame, because we were very aware of how important it was to give the player plenty of guidance in the missions, and dialogue was the obvious way to do that. We also wanted all of the core dialogue to have voice-overs attached, so that made it very difficult for us to add new elements late in the development.

**5. Too big a game and too much stuff.** The unrealistic time limit problem was further exacerbated by the size of the game we created. Originally, 20 single-player levels and 10 multiplayer levels seemed a reasonable size for the final game. However, when we started to get flak from the *Fallout* fans about the game not being true to the original, we decided to link the missions together with a world map rather than a linear mission structure. This in turn led to the inclusion of random encounters and unique encounters to add extra interest while traveling between the core missions. We produced special levels for bunkers where you could reequip and change your squad. This works very well in the game but increased our workload substantially. We also hopelessly overestimated how big our maps needed to be. We ended up with some truly huge playing areas and the associated problem of having to populate them with interesting encounters. Rather than reduce the size of the maps or lose maps altogether, we decided to press on and fill them with interesting objectives. We estimate that to complete all the core and secondary objectives and see all of the multiple endings takes about 40 hours of playtime -- even for someone who knows the missions well.

We should have been far more brutal in what we were going to drop from the original *Fallout*. We inherited a lot of nifty ideas, but many of them simply did not work in the context of a tactical game. For example, we have something like 150 different weapons. This sounds good, but the time spent rendering them and making sure that each one had its stats set up correctly was time that would have been better spent play-testing missions. This led to another problem: we had so many objects that it was difficult to differentiate clearly between them. In fact, all of our pistols are almost completely useless in the game, as they do not do enough damage and offer no clear advantage over an SMG. The character stats were also overly complex for a tactical game. We have 18 different skills, some of which are very poorly differentiated, such as the difference between doctor and first aid (we had to change the original doctor skill completely to make it useful). In our defense, *Fallout* also implemented these things badly, but I think we should have played the original *Fallout* games in a much more critical manner, noting things that just did not work in the original and stripping them out. The golden motto should have been "If it doesn't do something useful or if there is another thing in the game which does nearly the same thing -- take it out!" We stripped out some stuff, such as the speech skill, but added far more than we removed. The huge selection of stuff on offer is one of the game's selling points, but it has affected the quality of the finished product. In the end, I don't think we have really kept the original *Fallout* fans happy, and we compromised the game design at the same time.

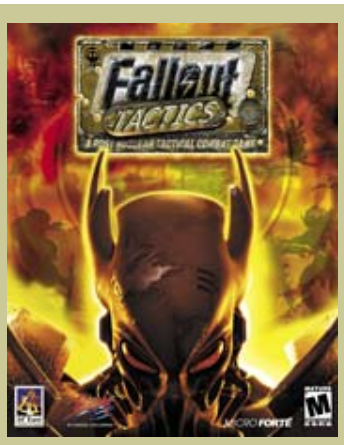
The size of the game also made QA very difficult. The problem is, of course, that every time the game changes the only safe way to test it is to start right back at the beginning. Because of the size of the game, that would have necessitated another 40 hours of testing after the final candidate was burned before it could ship. Time restrictions made this impossible, so the only way to test it was with the cheats enabled. Most of the obvious bugs were easy to spot in cheat mode, but the more subtle balancing issues only became apparent when the finished game was played through properly after it had shipped. The result is that some of the gameplay is rather obscure, and bugs have slipped in which can make it impossible to complete certain missions under specific circumstances. There are some missions where the whole map depends on finding one key in an obscure place, which is a shame because we had a very clear plan early on not do that kind of thing. There are missions which are almost painfully difficult to complete, resources such as medical kits and ammunition are also in rather short supply. Mind you, at least we don't have any jumping puzzles.

---

**Last Word**

Given the time restrictions, the scope of the project, and the experience level of the team I think this project went exceptionally well. There were a lot of things which we could have done better but hopefully we will do some of them next time. Unfortunately, while we had hoped to start on another Interplay project immediately after *Fallout Tactics* shipped, this did not eventuate, and in the interim we had no choice but to let our junior artists and QA team go. Still, the game has been well received, and we have a number of up and coming opportunities, so there is a strong chance that we may reemploy some of the junior artists again.

**The *Fallout* team.**




Game Data

**Micro Forte**
***Fallout Tactics***

**Publisher:** 14 Degrees East, a division of Interplay Entertainment Corp.

**Number of full-time developers:** Intitially 20 but ramping up to 27 for the second half of the project

5 programmers
3 QA
3 Designers
1 Producer
13 Artists

**Number of contractors:** None but Interplay produced all the Voice Overs, sound effects and music for the game

**Length of development (time):** 16 months plus two months

**Release date:** Gold on March 5, 2001

**Target Platforms:** PC

**Development hardware:** Mostly top end PCs and some Silicon Graphics workstations

**Development software:** Microsoft Visual Studio (C++), Photoshop, Max 2.5, Win CVS, Team Manager

**Notable technologies:** Bink and Miles

**Project size:**
~ 1500 files
~ 300,000 lines of code

Return to the full version of this article