Postmortem: Ion Storm's *Deus Ex*

By Warren Spector

Deus Ex shipped in June 2000. Sales were, and continue to be, strong, worldwide. Critical response (with one or two notable exceptions) has been positive. We've already won several "best of year" awards in the U.S., the U.K. and Germany. Needless to say it's gratifying when people appreciate your work.

We did a lot of stuff right on *Deus Ex*; we did a lot of stuff wrong. In this article, I'd like to take the opportunity to looks at some of that stuff. Specifically, I want to discuss:

- The design philosophies that led to the creation of *Deus Ex*.
- Technology licensing: where it helped us and where it hurt us.
- Scheduling methodologies and why they all failed (as they always do, on every project…)
- Management structures and team building techniques, some of which seemed like good ideas on paper but turned out to be unmitigated disasters in practice.
- The public relations triumphs and nightmares that often seemed as if they'd have as much impact on our success as the quality of our work.

Let's start with a simple question for those of you who have no idea what *Deus Ex* (a.k.a. "That Game with a Wacky Name") is…

The *Deus Ex* player's alter ego, J.C. Denton, strikes a heroic pose.

What is *Deus Ex*?

I'll try to be brief. (Those of you who know me know I'll probably fail…)

Fictionally, *Deus Ex* is set in a near-future version of the real world (as it exists if conspiracy buffs are right). For some real shorthand, call it "James Bond meets The X-Files." (Remember that seemingly innocent claim that *Deus Ex* is set in the real world. It'll come up again shortly…)

Conceptually, *Deus Ex* is a genre-busting game (which really endeared us to the marketing guys) -- part immersive simulation, part role-playing game, part first-person shooter, part adventure game.

It's an immersive simulation game in that you are made to feel you're actually in the game world with as little as possible getting in the way of the experience of "being there." Ideally, nothing reminds you that you're just playing a game -- not interface, not your character's back-story or capabilities, not game systems, nothing. It's all about how you interact with a relatively complex environment in ways that you find interesting

(rather than in ways the developers think are interesting), and in ways that move you closer to accomplishing your goals (not the developers' goals).

It's a role-playing game in that you play a role and make character development choices that ensure that you end up with a unique alter ego. You make your way through a variety of minute-to-minute gameplay experiences (which add up to a story) in a manner that grows naturally out of the unique aspects of your character. Every game system is designed to differentiate one player-character from another, and to allow players to make decisions that reflect their own biases and express character differences in obvious ways in the game world.

It's a first-person shooter because the action unfolds in real time, seen through the virtual eyes of your alter ego in the game world. Your reflexes and skill play an important part in determining your success in combat. However, unlike the typical FPS, *Deus Ex* doesn't force you to shoot every virtual thing that moves. Also unlike the average FPS, in which gameplay is limited to pulling a virtual trigger, finding blue keys to open blue doors and jumping to reach seemingly inaccessible locations, *Deus Ex* offers players a wide range of gameplay options.

And finally, *Deus Ex* is like adventure games in that it's story-driven, linear in narrative structure, and involves character interaction and item accumulation to advance the plot. However, unlike most adventure games (in which you spend the bulk of your time solving clever puzzles in a search for the next static, but very pretty, screen), *Deus Ex* asks players to determine how they will solve game problems and forces them to deal with the consequences of their choices.

Deus Ex was designed from the start to combine elements of all of these genres. But more important than any genre classification, the game was conceived with the idea that we'd accept players as our collaborators, that we'd put power back in their hands, ask them to make choices, and let them deal with the consequences of those choices. It was designed, from the start, as a game about player expression, not about how clever we were as designers, programmers, artists, or storytellers. Which leads naturally to a discussion of having clear goals -- the first thing I think we did right.

What Went Right

1. A clear high-level vision. It's pretty self-evident that you can't achieve goals if you're not clear about what they are. We knew with a high degree of confidence what kind of game we wanted to make. This was possible for two reasons. First, *Deus Ex* is a natural outgrowth of work done by and in some cases with the late, lamented Looking Glass Technologies. We were inspired as well by games made at Valve, Origin, and a host of other places. Many of the things we wanted to do were a reaction to things they (or we) didn't do, didn't do well or couldn't do at all in earlier games. We weren't building from scratch, but rather building on a foundation already laid for us.

Second, and on a personal level, *Deus Ex* is a game I've been thinking about since right around the time *Underworld 2* shipped. I've tried to get a game like this started several times (as *Troubleshooter* at Origin; in some respects, as *Junction Point*, for Looking Glass). Those games didn't happen for a variety of reasons:

- I didn't or couldn't sell the concepts to the money people.
- Then current technology wasn't up to the job.
- I didn't have a team that wanted to make this kind of game or the resources to build one.
- Publishers weren't interested in a first-person, cross-genre game.

Still, I never stopped thinking about these games and, despite their failure to reach production, they laid much of the conceptual groundwork for *Deus Ex*. The lesson here is that if there's a game you really want to make, don't give up on it. Someone will be foolish enough to give you the money eventually.

As an interesting (I hope!) historical footnote, I include here for the first time publicly, complete with typos and misspellings, the very first proposal I ever submitted for the old *Troubleshooter* concept back at Origin. Note the budget and projected release date and, oh, those system requirements! Note also the similarities (and differences) between *Troubleshooter* and what eventually became *Deus Ex*. [*Troubleshooter* Proposal]

Troubleshooter
IBM PC
WHY ORIGIN?
1994

1.0 High Concept:

It's Underworld-style, first-person action. But this is no fantasy. It's today. The real world.

No monsters. No magic. All action.

Everyone knows the movies: *Die Hard, Passenger 57, The Last Action Hero, Under Siege, Dirty Harry*...

Everyone knows the stars: Arnold Schwarzenneger, Steven Seagal, Bruce Willis, Wesley Snipes, Clint Eastwood...

Everyone knows the weapons: .44 Magnum, Ingram Mac-10, Atchisson assault shotgun, Browning High-Power, mini-Uzi...

Everyone knows the situations: It's you against the world, you against terrorists, psychos, the dregs of society. They're armed with high tech weapons and they've taken hostages.

You know what to do....

The question is, are you good enough?

2.0 Why this is an ORIGIN product:

It's Hollywood-inspired, big-budget, non-stop action. It's significant new technology. It's a logical extension of our existing first-person line -- we have fantasy covered with *Underworld*, we WILL have science fiction covered with*Bounty Hunter*, we have the real world covered with... well, we don't. *Troubleshooter* is bigger than life, but it's clearly rooted in the real world. I can't believe no one's done this before -- we have to jump on it before someone else does!

3.0 Product Overview:

You're an ex-cop turned "security specialist." That just means you get all the dirty jobs no one else has the guts to do. When the government or the police or business can't handle a problem, they call on you. Bomb threats? You get to check 'em out. Hijackers threaten to take over a plane? You end up on board. Some radical group takes a millionaire's daughter hostage? You get the call to go in and get her out.

You scope out the situation, checking maps and photos, walking around the site, probing for the best way in, the way that will put the fewest innocent people at risk. You try to talk a madman into surrendering before he blows himself and his hostages to kingdom come. You crawl through air ducts and sewers hoping you don't attract the attention of the bad guys with all the guns. You shoot it out with terrorists wielding enough firepower to take on a third world army.

Troubleshooter is a mission-oriented action simulation with no huge plot -- just get in and get out of each mission. Maybe 10-30 minutes of action per scenario. None of this 100 hours to finish the game and get your reward stuff. Like a flight sim, but it's just you, on the ground, with a gun.

I originally envisioned this as all new technology, but I could probably leach off of *Bounty Hunter*, once that project gets going. In game play, I see it being like *Underworld* in the richness of its world simulation, but like Wolfenstein in its emphasis on action over roleplaying and inventory manipulation. Ideally, I'd like to incorporate a head-to-head modem/network option, allowing one player to be the bad guy and the other to be the *troubleshooter*.

4.0 Technical Overview:

IBM PC 486, 4 Megs RAM, 320 x 200 VGA, full sound board support. Mouse, joystick and keyboard supported.

5.0 Audience:

Traditional ORIGIN buyers. I also hope the basis in reality and the short duration mission structure make *Troubleshooter* appeal to overworked older folks (the ones who have the money to buy machines capable of playing our games...) who just want to work off some frustration and then get back to their real lives.

6.0 Deal:

$500,000 Budget.
Planned ship in Q4 (March '95).

7.0 Risks:

High. There are all sorts of technological unknowns, things I want to do that haven't been done before. All in all, this is probably the toughest project on my wish list, but it might be the most satisfying... We might be able to minimize the risk by leaching off of *Bounty Hunter*.

8.0 Status:

Looking for concept approval so we can Go For Script.

Several years passed. Lots of games somewhat like *Troubleshooter* came and went. Game budgets went up dramatically -- $500,000 indeed! I left Origin and go to work for Looking Glass. *Troubleshooter* stayed on my mind.

In the fall of 1997, before Ion Storm entered the *Deus Ex* picture, I drafted a manifesto -- a description of an ideal game -- and also a set of "Rules of Role-Playing." Much of that material ended up in an article published in Game Developer ("Remodeling RPGs for the New Millennium". Here (for more of those historical reasons mentioned above) is the original draft of my Rules of Role-Playing, circa 1997:

The Rules of Role-Playing

- Always show the goal. Players should see their next goal (or encounter an intriguing mystery) before they can achieve (or explain) it.
- Problems not puzzles. It's an obstacle course, not a jigsaw puzzle. Game situations should make logical sense and solutions should never depend on reading the designer's mind. And there should always be more than one way to get past a game obstacle. Always.
- No forced failure. Failure isn't fun. Getting knocked unconscious and waking up in a strange place or finding yourself standing over dead bodies while holding a smoking gun can be cool story elements, but situations the player has no chance to react to are bad. Used sparingly, to drive a story forward, O.K. Don't overuse!
- It's the people, stupid. Role-playing is about interacting with other people in a variety of ways (not just combat… not just conversation…).
- Players do; NPCs watch. It's no fun to watch an NPC do something cool. If it's a cool thing, let the player do it. If it's a boring or mundane thing, don't even let the player think about it -- let an NPC do it.
- Have you patted your player on the back today? Constant rewards will drive players onward. Make sure you reward players regularly. And make sure the rewards get more impressive as the game goes on.
- Players get smarter so games get harder. Make sure game difficulty escalates as players become more accustomed to your interface and more familiar with your world. Make sure you reward the player by making him or her more powerful as the game goes on.
- Think 3D. A 3D map cannot be laid out on graph paper. It has to take into account things over the player's head and under the player's feet. If there's no need to look up and down -- constantly -- make a 2D game!
- Are You Connected? Maps in a 3D game world must feature massive interconnectivity. Tunnels that go direct from Point A to Point B are bad; loops (horizontal and vertical) and areas with multiple entrance and exit points are good.

A year or so later, *Deus Ex* lead designer Harvey Smith clarified and extended the original rules as follows:

The *DEUS EX* Rules Amendments & Addenda
Drafted by Harvey Smith (and endorsed enthusiastically by me) in 1998

1. Problems will have multiple solutions. Locations will be reachable in several ways. All missions, locations, and problems will be specifically keyed to:
    - Skills (and skill levels)
    - Augmentations (and augmentation levels)
    - Objects
    - Weapons
2. Gameplay will rely on a variety of "tools," rather than just one:
    - Character capabilities (skills/augmentations)
    - Resource management
    - Combat
    - Character interaction
3. Combat will require more thought than "What's the biggest gun in my inventory?":
    - A more relevant question might be, "How do I deal with this situation involving a few intelligent, dangerous enemies?"
4. Geometry should contribute to gameplay -- Whenever possible, show players a goal or destination before they can get there. This encourages players to find the route:
    - The route should include cool stuff players want or should force players through an area they wants to avoid. (The latter is something we don't want to do too often.)
    - Make sure there's more than one way to get to all destinations.
    - Dead ends should be avoided unless tactically significant.
5. The overall mood and tone will be clear and consistent:
    - Fear
    - Paranoia
    - Tension
    - Release (through combat and/or reaching a predetermined goal or NPC conversation)

The details of *Deus Ex* -- plot, character, game system design -- all changed radically since the days of *Troubleshooter* and manifestos and rules and rules addenda, but conceptually the game still follows most of the rules and meets the ideals outlined in the Game Developer article.

With these conceptual tools in mind, the *Deus Ex* team was able to assess design decisions and game system specifications, in light of what we wanted players to experience during the game and in light of our ultimate design goals.

So What Were Our Goals (In the Beginning)?

How did we intend to move from abstract ideas to game design specifics? We had to take our thinking to a deeper level. We had to start thinking about what we wanted players to be doing and thinking about as they played the game, rather than what we would be thinking about as we developed it.

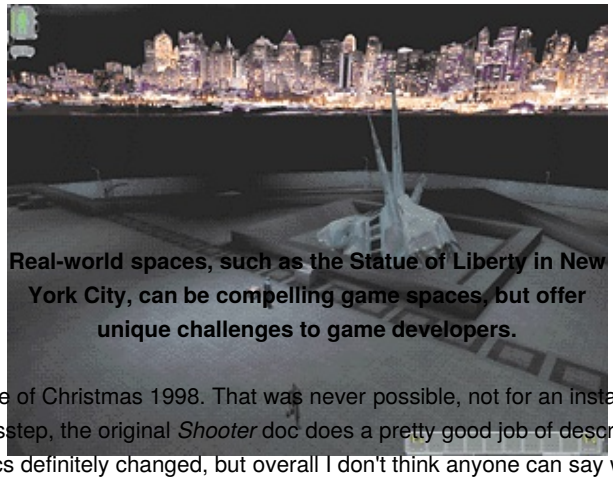
This led to some critical concepts, outlined here:

- Who are you? We wanted players thinking about who they wanted to be in our game world. Character differentiation was to be paramount. Even more important, though, was the desire to ensure that those character differences would be more than cosmetic -- they had to be expressible, minute-to-minute, in a deeply simulated game world, resulting in a unique gameplay experience for each player. This proved to be a far more important goal than any of us expected it to be.
- How do you behave? We wanted players thinking about how they wanted to interact with our game world. We knew we'd have to wean players from traditional puzzle/solution thinking and show them that *Deus Ex* was a game of problems (not puzzles!), all solvable in a variety of ways. This seemed critical to making character differentiation meaningful. The idea was to create a believable world and then offer game systems that encouraged players to explore that world in whatever way or ways they chose. The game would tune itself (however slightly) to the player's play style rather than forcing the developers' desired play style on players. We were tired of games that kept us on rails, offering the illusion of freedom and interactivity but without the reality, and we hoped players were as tired as we were of guessing what developers had in mind. We're a long way from being able to create a game in which players are truly free to do whatever they want -- believe me, there's plenty of illusion in *Deus Ex* -- but we knew we wanted to start taking at least some steps on the road to player control.
- Are you willing to pay the price? "Choice" and "consequence" were the two most frequently uttered words during our two to three years of development. What good is player control if all choices lead to the same result? Without real, predictable consequences, choice is irrelevant. (Which is probably why so many games seem so trivial -- they are trivial!)
- Are you there? We wanted players to feel like they were actually there, in the real world. We wanted this not only because we thought it would be cool (though we did think that!) but because we thought it was critical to making the above concepts actually work. If players were going to solve problems, they needed to be able to make some informed guesses about how those problems worked. If they were going to deal with consequences, they needed to be able to predict what those consequences might be. In other words, they needed to be able to apply some real-world common sense. If you fire a gun in the room you're sitting in as you read this, you can pretty much tell what's going to happen. (If you're in a public place, all hell is going to break loose; if you're at home alone, your neighbors might complain about the noise but that's about it…) Fire a gun in a game set in a fantastic alternate dimension and there's no telling. How can you possibly make a plan and execute it if you can't apply simple, real-world logic to the most straightforward situation imaginable? We wanted that real-world common-sense stuff. Firing a gun in *Deus Ex* should result in the real-world response. (Speaking personally, I was also just sick and tired of goofy fantasy settings and alien invasions.) Everything in the game would be real, based on something real or based on something someone, somewhere believed to be real. (We can show you the research behind most everything in the game, no matter how outlandish it may seem.) We looked for two kinds of real world spaces: Those that were naturally great game spaces (highly interconnected, multi-level stuff) and places people would just enjoy poking around in ways they couldn't in the real world (such as the White House). There was never a time when we thought realism should get in the way of fun -- anytime that happened, reality lost.

To recap: Know what your gameplay goals are and what kind of experience you want players to have before you spend ten seconds thinking about anything specific. Nice talk, but what did clear goals, manifestos, and commandments buy us?

2. We didn't skimp on preproduction. We spent the first six months of I (before we licensed a game engine), with a team of about six, just thinking about how we could turn our high-level goals into a game. We hammered on the setting and decided to move the game into the near future to buy ourselves some room to play around -- the real world, as we quickly discovered, was very limiting. Ultimately, we settled on a conspiracy-oriented background.

Here's what we had when we started: the very first design proposal (again, as is) for *Shooter*, our ironic working title for a game we never intended to be "just" a first-person shooter. [Shooter Proposal]



Real-world spaces, such as the Statue of Liberty in New York City, can be compelling game spaces, but offer unique challenges to game developers.

First of all, ignore the projected ship date of Christmas 1998. That was never possible, not for an instant. I don't know what I was thinking. Anyway, other than that (ahem) little misstep, the original *Shooter* doc does a pretty good job of describing the game that eventually became *Deus Ex*. Details changed. System specs definitely changed, but overall I don't think anyone can say we didn't deliver the game we said we would.

But how did we get from Shooter to Deus Ex? What were our first steps?

Shooter
Roleplaying in a World of Secrets, Lies and Conspiracies
Executive Summary
v. 1.0

Delivery Date
Target is Christmas 1998.

Genre
Modern Day plus about 50 years (to allow us to fudge reality, where necessary).

Category
RPG Adventure

Similar Titles
Half-Life (Sierra), *Fallout* (Interplay), *The Dark Project* (LookingGlass) *Goldeneye* (N64).

Setting
Several unique, real-world locations in the U.S., Europe and Far East.

Look
1st-person 3D with external camera views available as player option.

Plot Concept
The world of the 2050's is a dangerous and chaotic place. Terrorists operate openly, espousing a hundred beliefs and killing thousands of innocents to call attention to their causes. The world's economies are close to ruin on a scale not seen for over 100 years. The media openly encourage the worst in mankind. Governments seem powerless to deal with the situation.

In hidden meeting places around the world, a cabal of men and women confer. These are the leaders of societies so ancient and yet so secret, most people refuse to believe they exist. For hundreds, perhaps thousands, of years, they have controlled the world's finances, information flow, weapon production, medicine, religions… Now, as the world descends into chaos, they meet to determine if the time has come to emerge from the shadows, to take overt control of a world they have run from behind the scenes for so long. But, unable to reach agreement, they begin instead an internecine war that brings civilization even closer to the brink of destruction.

At the same time, in several major cities around the globe, another group of men and women -- the deadliest the world has ever known -- lie in wait. Enhanced senses, increased intelligence, biomechanically supercharged muscles and fantastic weaponry make these augmented agents all but unstoppable. And each is implanted with a tiny, high-tech brainwashing device which, activated, can turn its host into a helpless buffoon or a remorseless, conscienceless, killing machine -- whatever the controller wishes.

You are one of these agents.

Recruited from among the elite of the world's intelligence agencies, and endowed with nearly superhuman abilities, you await a signal from the most ambitious, most manipulative and most dangerous member of the shadowy cabal, a man known only as "Adam." Horrified at the chaos he sees all around him, he has concocted a plan to end the secret war and restore order to the world -- at his signal, you and your fellow augmented agents, will emerge to cripple or destroy each of the ancient and secret societies, in turn, clearing the way for a savior (Adam, of course) to take his rightful place as leader of all the societies. Once in control of the societies, he will offer mankind a simple choice: Obey and live in an orderly world free of pain and suffering, a world of Adam's creation; refuse and die.

Adam wants only to save the world from itself -- it's up to you to stop him.

The question is, can you assemble enough clues to figure all this out? Can you find and recruit the allies you'll need to survive, including the other augmented agents? Can you free yourself and the others from the grip of Adam's brainwashing device and put a stop to the secret society cabalists and the master manipulator, Adam. himself?

Each mission leads the PC deeper into a morass of suspicion, false motives, paranoia and conspiracy involving the highest levels of government, the media and the military-industrial complex. The missions stand alone but, taken together, they add up to a big story, with the player at the heart of earth-shattering yet believable events.

Set in a world very much like our own (if the conspiracy buffs are right), *Shooter* combines the best of *The Manchurian Candidate, Robocop* and *Colossus: The Forbin Project* in a world inspired by *The X-Files* and *Men in Black*.

Competitive Analysis

- Exciting, first-person, 3D roleplaying in a world that teeters on the brink of madness.
- True six degrees of freedom 3D engine, polygonal figures, killer AI and an innovative conversation system put you right in the thick of things.
- In-depth world simulation allows players to solve problems in a variety of ways.
- More than 30 core missions, and plenty of optional adventures provide 40+ hours of gameplay.
- Plugs into two popular fantasies -- the millennial madness that's gripping the world, exemplified by *The X-Files* and M*en in Black* and a general fascination with conspiracy theories and the desire to play with high-tech espionage toys.
- Emphasis on character development ensures that every player character will be unique. This, combined with our deep world simulation, ensures that each player's experience of the story is different, without having to resort to brute force branching tree structures.
- Non-combat interaction with dozens of unique non-player characters. A simple, elegant conversation system results in non-player characters you really care about. Engage them in conversation, seek information from them, recruit them to your cause, decide for yourself who you can trust…
- Clear goals, constant rewards, varied interactions with people and places as well as varied mission types (including Sabotage, Infiltration, Extraction, Rescue, Intelligence-Gathering, Thievery, Reconnaissance, Assassination, and all-out Combat) keep players coming back for more.
- Goals can be accomplished through stealth, careful planning, undercover work or conversation, through the use of unbelievably high tech equipment or brute force combat tactics.
- No weird "game spaces" -- every map recreates either a real place or a place with an instantly recognizable real-world function. We hope to recreate places like Camp David, the Kremlin (and the tunnels beneath it), the bone-strewn catacombs beneath Paris, Hong Kong's junk-filled harbor, the London underground and more.
- Situations that evolve over time, influenced by player action (or inaction). The state of the world changes to reflect the impact of player choices. Game events and the passage of time turn familiar locations into strange and terrifying places; trusted friends become the most bitter of enemies.

Technology

- Engine: 3D 6-degrees of freedom
- Characters: Real time 3D models.
- Creatures: Real time 3D models.
- Surroundings: Real time 3D models, sprites.
- Structures: Real time 3D.
- Vehicles: Real time 3D models.
- Interface: 2D graphics.

System Requirements

- Operating System: Windows 95
- Processor: Pentium 133
- Memory: 16 Megs
- Format: 4X CD-ROM Drive, 2 CDs
- Hard Drive Space: 50/100/200 Megs
- Video: DirectX 5; 8- and 16-bit color (hardware only); All DirectX-supported resolutions; 3DFX, etc.
- Audio: DirectX 5
- Multiplayer Support: DirectX 5 (DirectConnect, Modem, Network, Internet)

We worked on back-story stuff so we'd know what was going on in the world, even in places the player never got to visit. Some of this stuff may come to the forefront in *Deus Ex* 2 but, for *Deus Ex*, it was just a way of making sure we knew enough to include the kinds of small details that make a fictional world convincing.

We did a vast amount of research into "real" conspiracies -- the Kennedy assassination, Area 51, the CIA pushing crack in East L.A., Dwight Eisenhower's UFO connection, and of course Freemasons tunneling below the Denver airport and building abducted-baby cafeterias for alien invaders at George Bush's direction. Only a fraction of this stuff ended up in the game, but it gave us a peek into the minds of conspiracy buffs that was both scary and useful.

We also created a cast of more than 200 characters, many of whom didn't yet have specific roles in the game. Ultimately, this list proved to be both a help and a hindrance to designers as they fleshed out the missions. Characters sometimes suggested missions or subquests, but just as often ended up being filler we were reluctant to cut, even though their missions or story purposes changed during our storyline-focusing passes.

We worked out a bunch of missions -- more than 25 of them, taking the player from New York to London to Paris to San Antonio, to Austin to Siberia to Washington, D.C., to NORAD to Sunken L.A. (post-earthquake) to the Moon. We had wars in Texas, raids on concentration camps to free 2,000 prisoners from UN troops under FEMA control. Those of you who think the *Deus Ex* story line includes everything plus the kitchen sink now should have seen what we started with!

Either a failed stealth attempt or a frontal assault -- the choice is up to the players in *Deus Ex*.

We hammered on game systems. We conceived a skill system that didn't depend on die-rolls or tracking skills at a fine level of granularity. We came up with a system of "special powers" (nanotech augmentations) that differentiated the player character from ordinary humans. We designed a conversation system with some cinematic elements and some elements borrowed from console RPGs. We mocked up 2D inventory, skill, and augmentation upgrade screens, map screens, even a text editor so players could take notes. We conceived several player reward systems, including skill point awards, augmentation upgrades, weapon availability timelines and tool/object availability timelines.

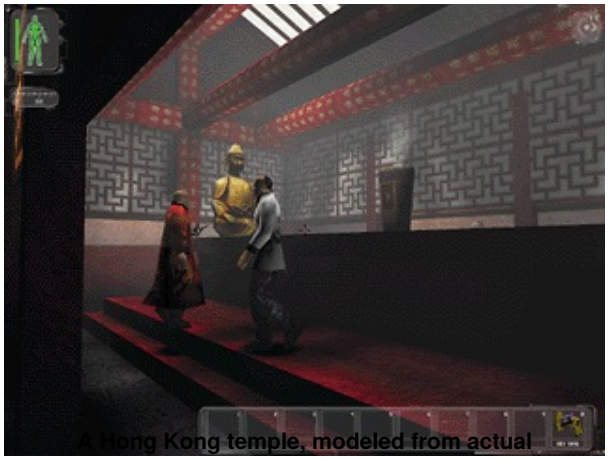
By March 1998, we had 300 pages of documentation and thought we knew everything we'd needed to know to make a game. Were we ever wrong. In the time between March 1998 and our Alpha 1 deadline of April 1999, that 300-page document mushroomed into more than 500 pages, much of it radically different from what we thought of and wrote initially. Clear goals and a detailed script are all well and good, but goals change, thinking changes, and game designs have to change, too. Which leads nicely into the next thing that went right.

3. Recognizing that game design is an organic process. Why did our thinking and goals change? There were lots of reasons.

NEW PEOPLE = NEW IDEAS

First, new people joined the team, with new ideas. Our staff grew from six people to roughly 20. I hired a bunch of people, of course, but we had the added excitement of integrating an entire art team assigned to us, in Austin, by an art director a coupleof hundred miles away in Ion Storm's Dallas office.

Despite all the preproduction work, despite all the rules and manifestos, *Deus Ex* was, like all projects, going to be created by a group of people, each with his or her own agenda, many of whom hadn't been involved in the preproduction process. In other words, like all projects, Deus Ex was a living, evolving, growing thing. And there were some amazing growing pains associated with its development. Getting everyone on the same page wasn't always easy. (O.K., sometimes it was a nightmare!)



A Hong Kong temple, modeled from actual photographs.

As we brought on new people, we found ourselves to be a team of hardcore Ultima geeks, hardcore shooter fans, hardcore immersive sim fans, strategy game nuts and console gamers. Some of our new team members proved to be "maximalists" -- wanting to do everything, special-case lots of stuff, and stick as close to reality as possible. Other team members proved to be minimalists -- wanting to include fewer game elements but implementing them exceptionally well, in ways that could be universally applied rather than special-cased.

Also, we made a point of letting select friends and colleagues play the game at various points along the way. We were interested in well-reasoned opinions from folks who understood the kind of game we were making intimately and who had a handle on the development process that was at least as good as our own. With all the new folks contributing and all the feedback from our chosen critics, well, let's just say we had some interesting debates at Ion Storm, Austin. Out of those debates new ideas arose, and the game changed as a result.

TECHNOLOGY IMPOSES LIMITS

Technology forced design changes, too. It took time to become familiar with the Unreal engine. I wish I could say we uncovered all its potentials and limitations quickly, but we didn't. Months of experimentation were necessary to reveal how best to do things in Unreal and what things not to do at all. When we stopped playing with Unreal andactually started working with it (roughly six to nine months after we got our hands on it), lots of ideas we'd come up with in the abstract didn't work quite as well in reality.

Multiple solutions falling out of our simulation didn't happen as often as we'd hoped. We just didn't have a deep enough simulation nor did we have the time or personnel to create one. We found ourselves in the position of having to brute-force the multiple-path idea, as developers on Ultima games, for example, have done for years -- though I think we do more of it, more consciously and more effectively, than anyone else has to date. Designers have had to plan a "skill" path past problems, an "action" path and a "character interaction" path. It works, but it's not what we originally intended.

Our original plan to build large, outdoor areas -- whole sections of New York, Area 51, lovingly recreated in excruciating detail gleaned from maps and satellite photos and, most notably, my dream of allowing players to explore the entire White House -- just proved to be unfeasible. There was no way any then-current renderer was going to allow us to do all that. The design had to change.

GAME SYSTEMS DIDN'T WORK AS INTENDED

A third area that influenced the changing nature of the game's design was when the game systems didn't work as we intended them to. High-level concepts imply gameplay but don't -- and can't -- define it. We quickly found that descriptions of game systems are no substitute for prototypes and actual implementation. We prototyped every game system, as documented, relatively early on. We built some test missions, not quite early-on-enough but still early.

These test systems and missions revealed gaping holes in our thinking or things that we thought would be true that turned out not to be true at all. For instance, our augmentation and skill systems proved dry and rather dull, once implemented, despite looking really good on paper. Those systems were designed around the totally valid idea that the computer would resolve actions without any secret (or even non-so-secret) die rolls required. Players would always know, with absolute certainty, based on their character development choices, whether they could accomplish something or not. The trick would be whether they wanted to do something or not, based on an assessment of the likely outcome and the likely consequences (for example, blowing down a door and setting off alarms versus the risk of picking a lock and being caught while doing it). In addition, I thought the tension of standing outside a locked door, not knowing if a guard was going to show up while you picked the lock would provide sufficient excitement. I thought knowing you could leap across a chasm because you had the Jump augmentation at Tech Level 3, opening up new paths through maps that were inaccessible to players without that augmentation, would be good enough to keep players interested.

When Gabe Newell from Valve came down and played our prototype missions, he correctly identified the utter lack of tension in our skill and augmentation use, as written up in the design doc and ably implemented by the coders. The worst was confirmed when Marc LeBlanc, Doug Church, Rob Fermier, and other friends from Looking Glass Studios and Irrational Games played the proto-missions and came to the same conclusions. Actually using skills and augmentations revealed things that merely thinking about them could never have revealed.

We took the criticism, and with it in mind, lead designer Harvey Smith revised the skill and augmentation systems pretty thoroughly, proposing an elegant system of consumable resources and time passage, all tied to skill level. This increased the tension level, provided new rewards, and allowed players to think and make informed decisions. Harvey also proposed a revision to the augmentation system, introducing an energy cost for their use (something I had foolishly rejected earlier on). Again, this gave us the opportunity to hand out items that would replenish energy -- in other words, we instantly had more things to hand out to players as rewards. It also introduced a level of tactical thinking to augmentation use that makes the system work. None of this would have happened without prototype missions and some harsh (but fair) criticism they allowed.

HOW FUN IS THE REAL WORLD, REALLY?

Another big reason for changes from our original design doc was our realization that the idea of a real-world RPG with real-world locations and real-world weapons was better in some ways than it turned out to be on the screen. Not to put too fine a point on it, Deus Ex became a lot less realistic as time went on.

When we started building places such as the Statue of Liberty, a couple of square blocks of New York City, the White House, Parisian streets, and so on, we found ourselves constantly asking several questions:

- How interesting is most of the real world as a gaming environment? The answer was a tough-to-swallow "not very." Hotels and office buildings aren't great game spaces. And we were a bit naïve in thinking we could create a believable environment that didn't include many such locations.
- How do we live up to people's expectations, particularly of places they've actually been? Believable settings raised expectations to unrealistic levels. We started facing comments like, "That's not what the inside of the Statue of Liberty looks like. I've been there. I know."
- How do we force current simulation tools to deal with the object and NPC density and, even more crucial, the level of object and NPC interaction people expect in a "real-world game"? We worked very hard to make sure our world was overflowing with people and objects. But we wanted every person and object in the game to be believable. People had to behave like people. Every last object in the world had to serve some purpose. There would be virtually nothing that was "just" decoration. But that doesn't mean that everything works just the way you'd expect. We started hearing things like, "Hey, why can't I use that telephone to call anyone I want whenever I want?" and thus had to cut some objects whose real-world functionality we couldn't capture in the game.
- Are humans interesting enough to carry an entire game? The answer to this question surprised me more than anything else we encountered during development. We were about year into the game's development when designers and artists started balking at a game that was entirely about human beings. Movies don't need nonhumans to be cool but the same cannot be said, apparently, for games. People seemed to want monsters and nasty bad guys. The feeling was so pervasive that it changed my thinking completely. The original design spec called for a couple of robots, but the team demanded that they be made a more important part of the landscape, so we added several new bot types. We also introduced genetically manipulated animals and even some alien-looking creatures, all of which grew naturally out of our game fiction. The game benefited, but this was a radical change from the original plan.

How do you know which of your core game goals are valid and which need rethinking? How do you know which game systems work and which don't? When is it possible to know what your game is really about? These questions are all answered by the "proto-mission." This is the first implemented mission, playable start to finish, the one that captures everything you want your game to be. Get this mission working as early as possible, so you can see all the things you thought would work that didn't. Then fix them

A genetically manipulated creature called a "Greazel" was added to *Deus Ex* in response to the team's feeling that human interaction might not be enough to carry

the entire game.

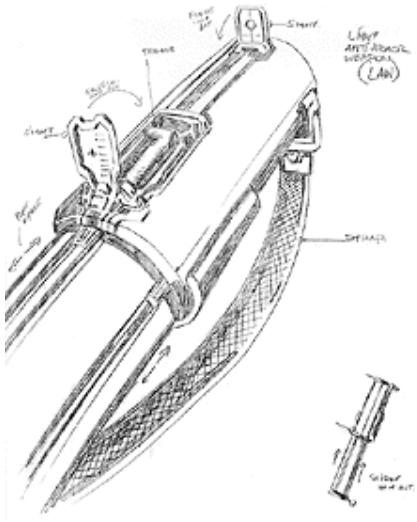
4. Scheduling around successively more refined "proto-missions" It's a truism that milestones should be testable, showing visible progress, whenever possible, and we lived up to that standard. We could always pull a version together, always show off for press or our publisher. Most importantly, we always knew where we were (even if that knowledge was sometimes painful). But the proto-mission idea is something beyond simply visible, testable milestones. The proto-mission is critical in the process of design, as well as in milestone and schedule setting.

One example of where our proto-mission idea was successful was in May 1998, when our milestone was to have prototypes of critical game systems in place and two test maps running, in this case the White House and part of Hong Kong. The maps were crude, the conversations raw, and the game systems hacked, but we could see -- and show -- the potential. To our advantage, we resisted the temptation to do just the stuff we knew would work and the stuff that would look the prettiest, and prototyped new, risky stuff first. Conversation, interface, inventory, skills, and augmentations were all at least hacked in so we could see them in action. The White House was likely to prove our toughest map challenge, so we built it first. (Almost unbelievably, I missed what may have been the riskiest, most critical game system in all of our early prototyping, NPC AI. I should have insisted on early prototyping of our AI but I didn't.) With the proto-mission system, we could immediately see some of the limitations of our technology. For example, we had some serious speed problems with areas as big as the White House and Hong Kong. After this, we knew we'd have to break maps up into small pieces. And we began to suspect, though I couldn't quite embrace the idea, that we'd eventually have to cut maps and missions from the game -- most notably the White House.

One of the many weapons which can be chosen by players.

In May 1999, we had a milestone calling for the delivery of the first two missions of the game, playable start to finish. All of our game systems were implemented (not hacked) as originally documented. You could start a game, create a character, upgrade skills, solve problems in a variety of ways, manipulate inventory, acquire augmentations, talk to NPCs, get and accomplish goals, save your game, and so on. To the team's chagrin, I had a tendency to call this the "Wow, these missions suck" milestone. It was around this time when Gabe Newell came for a visit and gave us our wake-up call, and where we all went, "Ulp! We have a lot of work to do." Our earlier demos had shown the potential of what we were doing. This demo showed us how far we had to go before we reached that potential.

This milestone also benefited us in that it showed us all the steps necessary to create a mission, and revealed the elements that really made the game work. That knowledge allowed us to go through our 500-page design document and cut everything that was extraneous, winnowing it down to a svelte 270 pages. Less game? Not at all. What was left was the best 270 pages -- the stuff that worked. "Less is more" was something Harvey Smith had said over and over, from the day he signed on as lead designer. While some team members resisted this notion outright, I took a middle road, which just frustrated everyone. In the end, we cut a lot, left a lot, and made a game that everyone on the team was happy with (I think). This milestone made it clear that the time had come to make cuts, while giving us enough knowledge to cut intelligently. If we had waited until beta to make cuts, with just a few months to go before our ship date (as many developers do), it would have been a disaster.



A detailed weapon sketch.

5. Licensing technology. We went into *Deus Ex* hoping that licensing an engine would allow us to focus on content generation and gameplay. For the most part, that proved to be the case. The Unreal Tournament code we ended up going with provided a solid foundation upon which we were able to build relatively easily. Dropping in a conversation system, skill and augmentation systems, our inventory and other 2D interface screens, major AI changes, and so on could have been far more difficult. UnrealEd, the main tool our designers used to generate our maps, was superior to anything else available. UnrealScript was very powerful and allowed programmers to do lots of interesting things quickly and easily. The dollars and cents of the deal were right, and I didn't have to hire an army of programmers to create an engine, 80 percent of whose functions already existed in Unreal Tournament. We were able to make what I hope is a state-of-the-art RPG-action-adventure-sim with only three slightly overworked programmers, which allowed us to carry larger design and art staffs than usual.

However, to my surprise, licensing technology didn't save us all the time I'd hoped it would. You'd think cutting a year or more of engine-creation off a schedule would result in an earlier release date. On *Deus Ex*, that didn't prove to be the case. Time that would have been lost creating tools was lost instead to learning the limitations and capabilities of "foreign" technology. Time that would have gone into making an engine went into focusing more on gameplay systems and tuning than normal. Unreal certainly allowed us to focus on content generation over everything else, but we spent more time doing it.

Everything in Deus Ex is about choices -- who you are in the world, how do you interact in the world, what are you carrying, and so on. In this case, the player has clearly decided to go through the game as a heavy weapons specialist, despite the fact that this will leave little room in inventory for anything else.
[Expand Image]

The biggest downside to licensing was that we were just never going to understand the code as well as we would have if we'd created it ourselves. That led to two distinct kinds of problems. First, there were areas where we ended up treating the engine as a black box. I think it's pretty well documented by now that we shipped *Deus Ex* with some Direct3D performance issues. Honestly, that didn't show up in any significant way during our QA process -- a slight problem here or there, but none of the dramatic slowdowns some players reported in the early days following our ship date. Once players started reporting troubles, we were kind of in a lurch -- we couldn't very well go in there and mess with the Unreal engine -- we just didn't understand it well enough to do that safely. We had built around the edges of Unreal without

ever getting too deeply into the nuts and bolts of it.

Second, because we didn't know the code inside out, and because we'd shelled out a fair amount of money for it, we tended to be conservative in our approach to modifying it. There were times when we should have ripped out certain parts of the Unreal Tournament code and started from scratch (AI, pathfinding, and sound propagation, for example). Instead, we built on the existing systems, on a base that was designed for an entirely different kind of game from what we were making. It's not that Unreal had bad AI or pathfinding or sound propagation, but those systems were designed for a straightforward shooter, which was not what we were making.

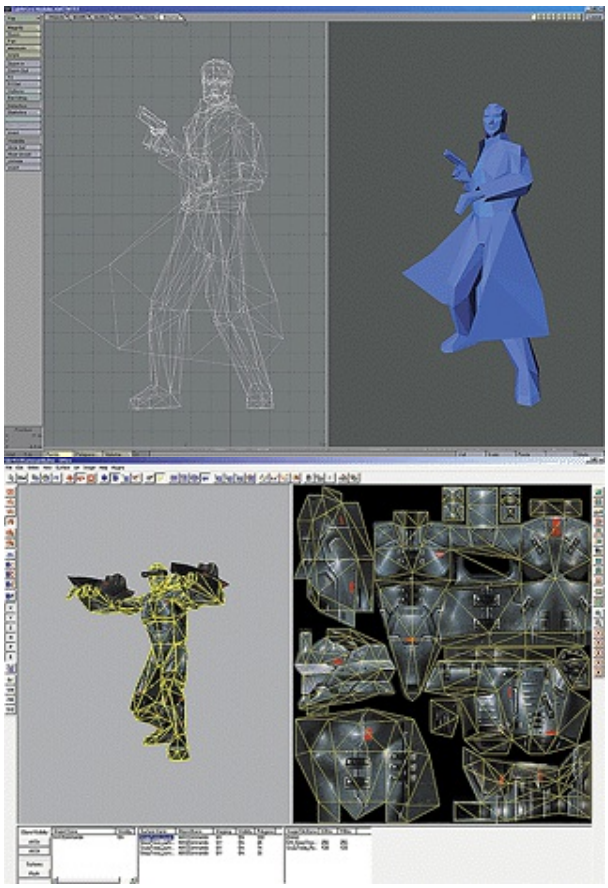
Technology licensing bought us a lot and cost us somewhat less. I guess the fact that we'll be licensing technology for our next round of projects, *Deus Ex 2* and *Thief 3*, says the price was right. But it remains an interesting dilemma, and we will be able to approach our next licensed engine with the wisdom gleaned from using Unreal for this project.

What Went Wrong

1. Our original team structure didn't work. You'd think after 17 years of making games and building teams to make games, I'd have a clue about team structures that work and those that don't. Ha! When I started pulling the *Deus Ex* team together I had a core of six guys from Looking Glass's Austin office. Having tapped Chris Norden to be lead programmer, I needed to find a lead designer and a lead artist. As I started casting about for the right person for the design job, something really good, but ultimately really bad, happened -- two guys came along with enough experience to expect a leadership position. Instead of doing the sensible thing and picking one of them, even if that meant the other chose not to sign on, I got cute. I created two design teams, each with its own lead.

I put together two groups of people with differing philosophies -- a traditional RPG group and an immersive sim group. We were making a game designed to bust through genre boundaries, and I thought a little competition and argumentation would lead to an interesting synthesis of ideas. I thought I could manage the tension between the groups and that the groups and the game would be stronger for it. My plan didn't work.

The design team was fragmented from the start. We had to name one of the groups "Design Team 1" and the other "Design Team A." (Neither group would settle for "2" or "B.") It became apparent -- later than it should have -- that I was going to have to merge the two groups and have a single lead designer. When I finally made that change I disappointed some folks, but the game was the better for it, and that's what's important in the end.



Bringing believable human characters to life is no easy task. The artists, whether working on concept art, 3D models, or texturing, had their work cut out for them. The job was made harder than necessary by a less-than-optimal team structure.
[Expand Image]

There were also challenges on the art side. *Deus Ex* suffered dramatically because for over a year, the artists "on the team" worked not for me or for the project, but for an art director in Ion Storm's Dallas office. Don't misunderstand -- the art director was a talented guy. But talent doesn't make up for a matrix management structure (wherein resources in a department or pool are "lent out" to a project until they're not needed anymore) ill-suited to the game business, and it doesn't make up for being off-site. During this time, the art department drifted a bit. It was unclear whether the artists worked for me or for the art director in Dallas. I couldn't hire, fire, give raises to, promote, or demote anyone on the art team. We were assigned some artists who weren't interested in the kind of game we were making. The matrix management experiment made things a little tense, and presented many unanswerable dilemmas. Matrix management may work in some circumstances, at some companies, in some businesses. But I've never seen it work in gaming, and I've seen it attempted at three different companies. It especially doesn't work when one of the department managers isn't on-site.

I argued for a year that matrix management had failed at Origin and at Looking Glass. I had no doubt it would eventually fail at Ion. Eventually I got my way, and things got much better on the art front once the artists were officially part of the *Deus Ex* team. Still, I can only imagine how *Deus Ex* might have looked if we'd been one big happy team, including the artists, from the start.

If the experience of *Deus Ex* taught me one thing, it's the importance of team dynamics. You have to build a team of people who want to be making the game you're making. You have to deal with personnel issues sooner rather than later. And there has to be a clear chain of command. Many decisions can be made by consensus, but there can only be one boss for a project, there can only be one boss for each department, and department heads have to answer to the person heading up the project.

2. Clear goals are great . . . when they're realistic. We started out thinking very big. That in itself isn't bad -- it's necessary to advance the state of the art -- but we were unrealistic, blinded by promises of complete creative freedom, and by assurances that we would be left alone to make the game of our dreams. A really big budget, no external time constraints, and a marketing budget bigger than any of us had ever had before made us soft.

Let me give you some specific examples of ways in which we outreached ourselves in the original design of *Deus Ex* (before we made significant cuts). For one, there's no way, in a first-person RPG, to stage a raid on a POW camp to free 2,000 captives. Also, there's no way to re-create all of downtown Austin, Texas, with any degree of accuracy. Third, blinded by the power of UnrealScript, many of our original mission concepts depended upon special-case scripting and lots of it. We discovered the need for general solutions rather than special-case solutions later in the project than we should have (this despite much harping on the subject by some team members).

Find your focus early and maintain that focus throughout. General solutions are better than special casing. Give players a rich but limited tool set that can be used in a variety of ways, not a bunch of individual, unpredictable solutions to every problem. Always work within the limits of your technology rather than trying to make your technology do things it wasn't meant to do. Big budgets, lots of time, and freedom from creative constraints are seductive traps. Don't fall into them. Don't settle for less than greatness, but don't think too big. Balance should be the goal.

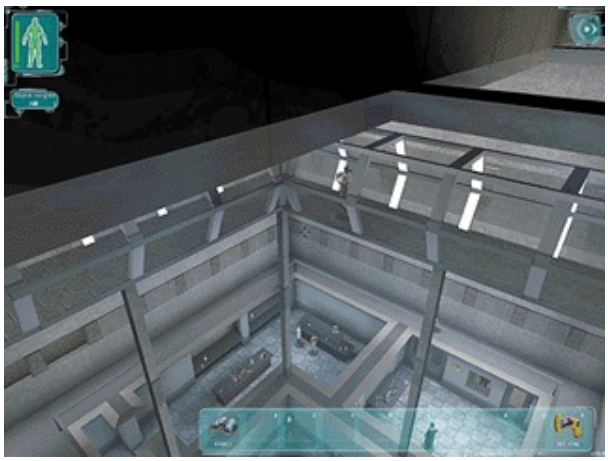
3. We didn't front-load all of our risks. In fact, we missed a big one. We were smart enough to realize we'd have to prototype and implement our new game systems early so we'd have time to tweak and refine them sufficiently. We did our conversation system and our complex 2D interface screens early, which was a good thing, too -- they required as much tweaking as we feared. And in the end, they turned out pretty well, I think.

Unfortunately, we missed one huge risk area -- artificial intelligence. I don't know how we missed it, but we did. It's not that we didn't spend time on AI. We started thinking about AI early in preproduction. Unfortunately, what that meant was that the AI was, to a great extent, designed in a vacuum, and as is often the case, we didn't really know what the game required with respect to AI until relatively late in development. And that meant implementing AI features early on that ended up being unnecessary later, once our design had evolved into its final form. In addition, building on the base of Unreal Tournament's pure shooter AI meant that, instead of designing a system specifically for our needs, we ended up adding stuff and tweaking until the bitter end, causing NPC behavior to change constantly, right up to the last day of development.

We ended up with some pretty compelling AI, but the problem of convincing people they're interacting with real people is immense, particularly when you're talking about characters whose reactions have to run the gamut from fear to friendliness to violent enmity. That's not a challenge many games take on (with good reason), but it was one we had to take on for *Deus Ex*. Our sin was, I think, giving people a hint of what human AI could be in games, but delivering the goods inconsistently.

As I write this, we're discussing whether we want to risk making some fairly radical changes to the AI in a patch for a game that most people seem to like, and in which NPCs basically behave as much like real people as they ever have in any game. There's no telling which way our decision will go at this time.

4. Proto-missions redux. Game Developer's Postmortems typically focus in on things the team clearly did right and things the team clearly did wrong. It sure is nice when things are that clear. Maybe it's just me, but I almost never see things in such black-and-white terms. Most of the time, problems are knotty and solutions are far from obvious or clear-cut, which is where the final two "What Went Wrongs" fall.



Not all of the places players visit in Deus Ex were modeled after real-world spaces. The team had to make compromises to gameplay and create spaces that were dramatic or more "fun" than one typically encounters in the real world.

As I already mentioned, we recognized the need for proto-missions relatively early on, and built our schedule around the idea. We implemented two such missions, which helped us identify many things that didn't work (and many that did). With proto-missions in hand, we found ourselves at a critical juncture with two possible choices to make, the implications of which I still don't entirely understand.

On one hand, I could have gone off with some subset of the team and tweaked our proto-missions until they were absolutely right and models for all subsequent mission implementation before turning the rest of the team loose on implementation of the rest of the missions. On the other hand, I could have kept the entire team in implementation mode, getting all of the missions to the level of the proto-missions, meaning none of them would be exactly right but we'd be able to see the shape of the entire game and all of the missions would be ready for tuning at about the same time. The first approach would have left large portions of the team in thumb-twiddling or make-work mode for some unspecified period of time. This promised to prove that we could create a ground-breaking, compelling game, but could leave us without a finished game to ship. The second approach would have kept everyone productive throughout the project and at least put us in position to decide whether or not to ship the game at some foreseeable point in the future. The question was whether we would be able to turn all of the bare-bones missions into something fun or not.

I chose the latter approach and told everyone to get the game "finished" and playable at a bare-bones level. We'd worry about fleshing out all the missions, making the game as interesting and fun and dense and exciting as it needed to be during the inevitable gameplay tuning, tweaking, and balancing phase at the end. This probably isn't so much of a "What Went Wrong" as it is an open question of whether that was the right call. I think so, and the plan clearly worked to the extent that we shipped a game that people seem to like pretty well. But it's unclear to me whether using our proto-missions to fine-tune might not have resulted in an even better game.

5. Is it true that any publicity is good publicity? Naturally, this wouldn't be a complete or accurate picture of the development of *Deus Ex* if we didn't take a look at the Sturm und Drang that was Ion Storm. In case you you've been living under a rock, there's been a lot of hype surrounding the company. On the negative side, Ion Storm was heaped with bad press for much of 1998 and 1999. The company did the same things all game companies do, went through the same problems, but because we painted a big ol' "suck it down" target on our chests, the gaming press and a fair number of hardcore gamers went after us with a vengeance.

Not too surprisingly, this had an effect on those of us working away in the Austin office. Morale hits were frequent and problematic. It simply isn't possible to be bombarded by negative press about the company you work for and not take it somewhat personally. Trust me when I say that seeing your personal and private e-mails posted on the Internet is a devastating experience. Also, recruiting was more difficult than it should have been. We were able to put together an incredibly talented team for *Deus Ex*, but too many talented people told us that while they would like to work on *Deus Ex*, they couldn't work for Ion Storm. Eventually, a "we'll show them" mentality became prevalent in Austin. I don't know that anyone who worked on *Deus Ex* thought of him- or herself as part of the same company making Daikatana and Anachronox up in Dallas. That kind of us-versus-them thinking is rarely good in the long run.

Now that we've shipped, the reviews seem to fall into two categories -- those that begin with some statement implying that Warren Spector makes games all by himself (which is silly), and those that begin with some statement proclaiming that *Deus Ex* couldn't possibly have been made by Ion Storm (also silly). Silly or not, there's a level on which we're still trying to live down our past, at least in terms of the media's perception of our game and the company that paid the bills here.

But, for all the problems, being associated with Ion Storm wasn't all bad -- far from it. On the plus side, it isn't as if anyone from Rolling Stone, Entertainment Weekly, the New York Times, the L.A. Times, USA Today, Mother Jones, the Wall Street Journal, Forbes, Fortune, Time, Architectural Digest, CNN, or the BBC ever banged down the doors at Origin or Looking Glass to talk to me or anyone on any of my teams. In reality, the bad publicity was almost entirely limited to the gaming press. The mainstream media, which barely notice anything about gaming (other than the fact that we supposedly turn normal kids into vicious killers) didn't seem to care about the bad stuff. But they sure did take notice of us. Ultimately, Ion's ability to attract attention to itself, even if it was sometimes in negative ways, probably worked to our advantage. Whether publicity at any cost is good or bad is still an open question for me.

What I'm Not Sure About

Given the benefit of 20/20 hindsight, it's easy to identify some things
as having gone "right" and other things as having gone "wrong." However, some of the most interesting things to consider are the ones that

aren't so easily pinned down. Here are some questions that are
still very much open in my mind. (If any of you have answers, feel free to share them!):

- **Is it better to start the design process with fiction and high-level gameplay goals or to dive right into game systems?** We did a much better job, I think, of the former than the latter. Too much of our system stuff had to be rethought relatively late in the project. Only the conversation and inventory systems are largely untouched from where we started. We remained true to our high-level goals, but I can't shake the feeling we could have done a better job early in the project on the system design front.
- **Is iconic/abstract representation of characters, power-ups, player rewards, tools, objects, and so on better than realistic/specific representation?** In other words, are instantly identifiable floating crosses better as healing items than a med-bot, something the player may or may not be able to identify? Is it compelling to wonder if that guy over there is a good guy or a bad guy? Or is it better to know just by looking at him, so you can plan accordingly? As in so many things, we went with a hybrid approach -- nothing as extreme as floating health restorers, but instantly recognizable good guys, bad guys, rewards, and objects.
- **Is it better to stop the action while the player is in interface screens or to keep the action going à la *System Shock 2*?** We chose to stop the action because, for us, the tactical decisions this allows outweighed the artificiality and the loss of immersion. Was that the right decision? Probably, but there's no way to assess the road not taken in this case.
- **Should you get to name your character or not?** A holy war almost broke out on the *Deus Ex* team about this. "If you can't name your character, it's not an RPG," said some. "If we don't name the character, how do we write and record compelling conversations and create a cool story?" said others. "Story isn't the point…" "Yes, it is…" and on and on and on. We compromised: we gave the player character a code name and back-story but let the player select his real name, which came into play in various ways (though never in speech).
- **Is it better to worry about graphics and art direction and cinematics early or late?** We did zero flic work until the very end of the project, and though the game looks good, we left it to the end to go back and make a pass at consistency (getting the lighting and texturing just right, and so on). I decided it was more important to get the gameplay under control than to get the game looking good. We did make our art direction pass and we did make the game look better -- really good, in fact, in my totally prejudiced opinion. But it's unclear to me whether the game could have looked even better if we'd gone the other way and dealt with art issues sooner rather than later.

The Bottom Line

Part of the challenge of game development is making the tough decisions along the way, leading to many difficult junctures when you have to determine that something that can't be done right in the game shouldn't be done at all. Notice the complete lack of references to multiplayer action in this Postmortem. We wanted to provide multiplayer support but didn't have the time to do the job we knew we needed to do, and so it got cut.

Now, generalize from that point: It's all well and good to have design goals and an ideal game pictured in your head when you start, but you have to be open to change and realistic about what can and can't be done in a reasonable time frame, for a reasonable amount of money, with the personnel and technology available to you. And if you don't have time to do something right, cut it and do everything that's left so well that no one notices the stuff that isn't there.

I'm not saying we did that perfectly on *Deus Ex*. We certainly didn't ship a perfect game. But if we hadn't gone into development with the attitude that we'd do things right or not at all, we would have fallen far shorter of perfection than we did. How close we did get is something all of you can decide for yourselves. All I know is we're going to get closer next time.

Game Data

Deus Ex

Publisher: Eidos Interactive

Number of Full-Time Developers: Approx. 20: 1 of me, 3 programmers, 6 designers, 7 artists, 1 writer, 1 associate producer, 1 tech

Number of Contractors: Approx. 6: 2 writers, 4 testers

Development Time: 6 months of preproduction and 28 months of production

Release Date: June 23, 2000

Target Platform: Windows 95/98/NT/2000 plus third-party Macintosh and Linux ports

Critical Development Hardware: Ranged from dual Pentium Pro 200s with 8GB hard drives, to Athlon 800s with 9GB fast SCSI, and everything in between. More than 100 video cards were cycled through during development.

Software Used: Visual Studio, Lightwave, Lotus Notes

Notable Technologies: Unreal engine and associated tools such as UnrealEd and ConEdit (our proprietary conversation editor)

Return to the full version of this article