UBM

INNER PRODUCT STREAMING VIRTUAL TEXTURES

VOL17 NO3 MARCH2010

gamedeveloper

THE LEADING GAME INDUSTRY MAGAZINE

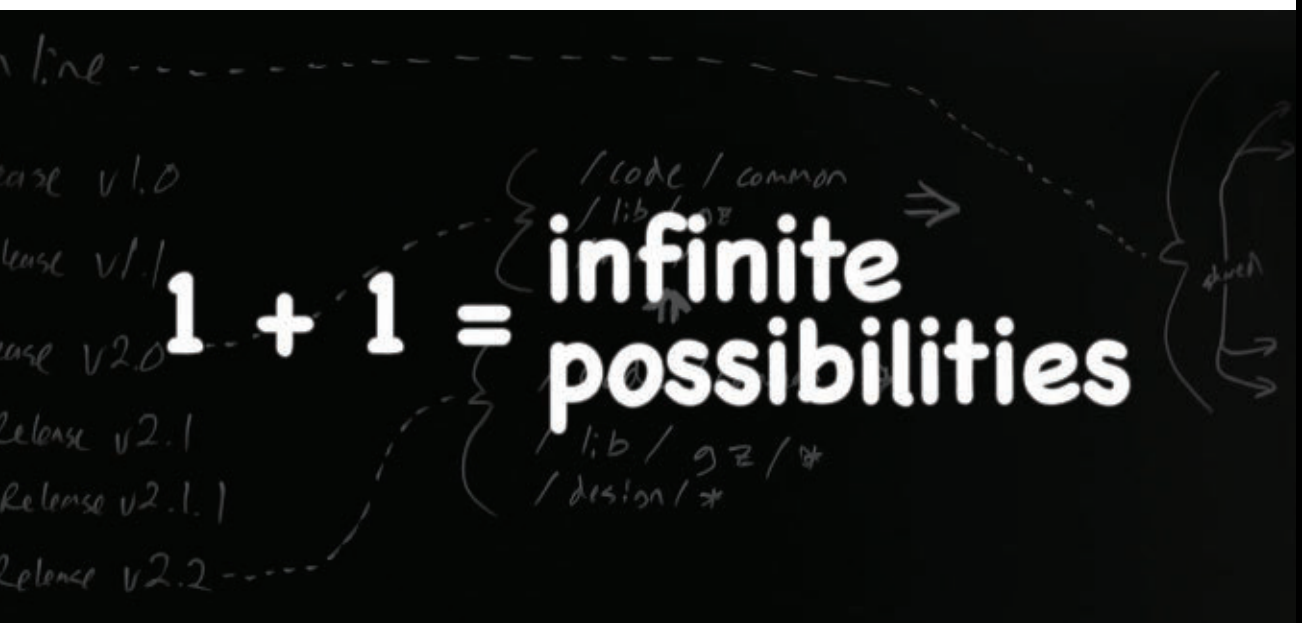
UNCHARTED 2
AMONG THIEVES

PROGRAMMERS STRIKE BACK
HACKS, KLUDGES, AND UNDER-HANDED TACTICS FROM BEYOND THE MATRIX

GAMES WITHOUT FRONTIERS
PRODUCTION TECHNIQUES ACROSS THREE LEADING STUDIOS

ART & SOUL
AN INTERVIEW WITH NCSOFT ILLUSTRATOR AND DIRECTOR HYUNG-TAE KIM

www.seapine.com/gamescm
Satisfy your quality obsession.



Get More Change Management with Seapine Integrated SCCM

TestTrack Pro + Surround SCM = infinite SCCM possibilities. Seapine's integrated software change and configuration management (SCCM) tools do much more than competing tools, and at a much lower price point. Start with TestTrack Pro for change management and add Surround SCM for configuration management—two award-winning tools that together give you the best integrated SCCM solution on the market.

- Link issues, change requests, and other work items with source code changes.
- Manage simple or complex change processes with flexible branching and labeling.
- Coordinate distributed development with RSS feeds, email conversation tracking, caching proxy servers, change notifications, 3-way diff/merge, and other collaboration features.
- Enforce and automate processes with incredibly flexible work item and file-level workflows.

Built on industry-standard RDBMSs, Seapine's SCCM tools are more scalable, give you more workflow options, and provide more security and traceability than competing solutions.

Get more, do more with Seapine tools. Visit **www.seapine.com/gamescm**.

Seapine Software™

QA Wizard® Pro Automated Testing

Seapine CM® Change Management

Surround SCM® Configuration Management

TestTrack® Studio Test Planning & Tracking

TestTrack® TCM Test Case Management

TestTrack® Pro Issue Management

gamedeveloper

CONTENTS.0310

VOLUME 17 NUMBER 3

POSTMORTEM

FEATURES

COVER ART: NAUGHTY DOG ART TEAM

DEPARTMENTS

PLAYING GOD

A CALL FOR NEW UNIVERSE CREATION IN GAMES

REMEMBER WHEN THE NORM FOR a video game was a blue hedgehog that ran fast and collected rings and emeralds? Or a plumber that took mushrooms to become large, and grabbed a flower to throw fireballs? In reality they do none of those things, but in the name of a game, they make sense, inspire wonder, and create a new universe.

This isn't another one of those articles about the good old days, and how everything used to be better. Rather, this is an article about missed opportunity.

TIME TO CRATE

>> As the graphical capabilities of computers and home consoles increased over time, and as demographics skewed older, the temptation to emulate and recreate reality grew stronger and stronger. To that end, games increasingly tried to make their systems and design follow realistic constructs, boasting the most realistic cars and licenses, or the most realistic guns, or a military contractor on staff to advise on tactics. But games are not reality, they are games.

We've seen time and time again that the closer you try to emulate reality, the more the "game" aspects begin to stick out. Invisible walls in FINAL FANTASY, or grenades spawning at your feet when you go the wrong way in MODERN WARFARE 2 are examples of kicking the player out of that illusion of reality, and letting them know that yes, this is a game, and yes, the rules are designed to keep you in the space of this world, not the real world.

In reality, as a soldier I could disobey my orders and go exploring around the other side. I could be cowardly and turn back to base. Games shouldn't have to plan for every eventuality, of course, but it's not so hard to create universes that are compelling but where the unusual—or even simple backtracking—is not so unfeasible.

Emulation of reality also brings with it all sorts of moral concerns about what is being taught to children or impressionable folk. Not that this should influence creative decisions, but when a cartoon mouse hits someone with a mallet it's a lot different from when a prisoner in MANHUNT does it. The discussion needs to take place, even if the decision is ultimately to go with reality.

PANDORA'S BOX

>> Games that emulate reality do have the powerful opportunity to make people think carefully about the world around them. By placing the player in real-world situations and applying real consequences in-game, a spark may well go off in little Jimmy's brain. But there's a high likelihood he still won't see the consequences as real. After all, you turn off the TV or the monitor and that entire world is gone.

I'm not a fan of James Cameron's *Avatar*, but its simple story resonated with a lot of people, and I've heard more people talking about injustice and the politics of power based on having watched a movie about fake blue people than I have heard them talking about Guantanamo Bay, or the war in Afghanistan. It doesn't mean these people are stupid, or even necessarily uninformed, but it does mean that in order to really reach them, they had to be approached in a different way. I think that this can be true of fun, as well as serious messages.

We miss out on some of the great potential of this medium if we focus too heavily on the real. We have the power to create entire worlds—isn't using this power to create a shadow of reality a bit of a cop-out? And really, it's only a conceptual cop-out. In practice, reality is quite hard to recreate. This is why the lushly-detailed world of *Avatar*'s Pandora is so compelling to people. It's new, but recognizable. It's compellingly different, but not alienating. This is the potential that exists within games.

Maybe in the past we created crazy games simply because we couldn't recreate reality with the technology. Consider BAYONETTA. Here you've got a woman whose clothing is made of her hair, and has guns in her shoes. I've heard a lot of people—journalists especially—talk about how crazy this is. In 1992 this would not be crazy, this would be par for the course in the creation of a video game.

CHOOSE YOUR ADVENTURE

>> There is a choice that developers can make now. We know that we can visually emulate reality to a pretty convincing level. Now is the time when we can decide whether we want to use that power to recreate reality or forge universes of our own. We don't need space marines or aliens to do it, either. The world of BIOSHOCK's Rapture captivated audiences immediately. It was recognizable, but different, and this is something that resonates with people.

I just downloaded and played the HEAVY RAIN demo on the PS3. I urge readers to do the same and see how they feel about the reality emulated there. Sure, there are complex emotions and scenarios in place here—but when I'm playing, I'm just pressing random buttons that come up on screen and have nothing to do with the actions I'm performing. Whether you agree with designer David Cage in his choice to make the game this way is not the point. The important thing is that he did make a specific choice, as he told me in a Gamasutra interview.

When a game is made, think—what universe fits my view? Can I tell my story with a Rapture? With a Pandora? Or does it need to exist in my reality? If these questions are answered honestly, and with real thought, games will resonate better, and their messages, their fun, and their immersion will only increase in potency.

—Brandon Sheffield

gamedeveloper
www.gdmag.com

UBM Think Services, 600 Harrison St., 6th Fl.,
San Francisco, CA 94107
t: 415.947.6000 f: 415.947.6090

SUBSCRIPTION SERVICES

FOR INFORMATION, ORDER QUESTIONS, AND ADDRESS CHANGES
t: 800.250.2429 f: 847.763.9606
e: *gamedeveloper@halldata.com*

EDITORIAL

PUBLISHER
Simon Carless | *scarless@gdmag.com*
EDITOR-IN-CHIEF
Brandon Sheffield | *bsheffield@gdmag.com*
PRODUCTION EDITOR
Jeffrey Fleming | *jfleming@gdmag.com*
ART DIRECTOR
Joseph Mitch | *jmitch@gdmag.com*
CONTRIBUTING EDITORS
Jesse Harlin
Steve Theodore
Daniel Nelson
Soren Johnson
Damion Schubert
ADVISORY BOARD
Hal Barwood **Designer-at-Large**
Mick West **Independent**
Brad Bulkley **Neversoft**
Clinton Keith **Independent**
Bijan Forutanpour **Sony Online Entertainment**
Mark DeLoura **Independent**
Carey Chico **Pandemic Studios**

ADVERTISING SALES

GLOBAL SALES DIRECTOR
Aaron Murawski e: *amurawski@think-services.com*
t: 415.947.6227
MEDIA ACCOUNT MANAGER
John Malik Watson e: *jmwatson@think-services.com*
t: 415.947.6224
GLOBAL ACCOUNT MANAGER, RECRUITMENT
Gina Gross e: *ggross@think-services.com*
t: 415.947.6241
GLOBAL ACCOUNT MANAGER, EDUCATION
Rafael Vallin e: *rvallin@think-services.com*
t: 415.947.6223

ADVERTISING PRODUCTION

PRODUCTION MANAGER
Pete C. Scibilia e: *peter.scibilia@ubm.com*
t: 516-562-5134

REPRINTS

WRIGHT'S REPRINTS
Ryan Pratt e: *rpratt@wrightsreprints.com*
t: 877.652.5295

THINK SERVICES

CEO UBM THINK SERVICES Philip Chapnick
GROUP DIRECTOR Kathy Schoback
CREATIVE DIRECTOR Cliff Scorso
CHIEF INFORMATION OFFICER Anthony Adams

AUDIENCE DEVELOPMENT

TYSON ASSOCIATES Elaine Tyson
e: *tysonassoc@aol.com*
LIST RENTAL Merit Direct LLC t: 914.368.1000

MARKETING

MARKETING SPECIALIST Mellisa Andrade
e: *mandrade@think-services.com*

UBM TECHNOLOGY MANAGEMENT

CHIEF EXECUTIVE OFFICER David Levin
CHIEF OPERATING OFFICER Scott Mozarsky
CHIEF FINANCIAL OFFICER David Wein
CORPORATE SENIOR VP SALES Anne Marie Miller
SENIOR VP, STRATEGIC DEV. AND BUSINESS ADMIN. Pat Nohilly
SENIOR VP, MANUFACTURING Marie Myers

UBM
WWW.UBM.COM

THE GAME INDUSTRY IS CHANGING

YOUR QUALITY REMAINS SECURE

Testronic Labs has launched a dedicated social and casual game division to embrace that change.

With ten years of experience in game and web testing, we set the QA standard.

TALK TO US AT GDC

Booth No. 1444 South Hall

Testronic Labs provides independent QA and localization services to publishers and developers worldwide.

TESTRONIC laboratories

For more information:
talk@testroniclabs.com
+1 (818) 845 3223
www.testroniclabs.com

the art history of games in atlanta

Brenda Brathwaite on creating board games about tragedies (Photo by Auriea Harvey).

WE WERE A MIXED CROWD. Professional game developers, art aficionados, students of all varieties, and a number of folks from the area that were just plain curious. John Sharp, Ian Bogost, and Michael Nitsche began the Art History of Games conference with a claim captured from SIGGRAPH. "Games: 18.25% art."

No matter our background, attendees could all agree that putting a percentage on game art was nonsense. This set the tone that carried through the entire event: we would welcome bold, over-generalizing statements, if only to have something to talk about. This was going to be fun, not clinical.

Each presentation brought something unique and valuable. Unfortunately 18 talks spread across three days can't fit on one page, so I'll present a few highlights.

John Romero delivered the opening keynote. His main point was that the innovators of our industry are still alive today, and we ought to be learning everything that we can from them. In 1998 John Romero caught up with the legendary programmer from FINAL FANTASY I–III and SECRET OF MANA, Nasir Gebelli, for an interview. "Our masters worked within a lot of constraints." Romero pointed out. "The Atari 2600 was created to play just two games."

Day two switched to an academic perspective, digging into the cultural roots of games as art. Much of the day included background on classical art, including Celia Pearce's introduction of the Fluxus movement, and John Sharp's discussion of games taking on new cultural roles during the Renaissance. Projects by the Jodi art collective—such as those changing the rendering process of existing first-person shooters WOLFENSTEIN and QUAKE to be more abstract—were offered by Jay David Bolter and Brian Schrank during their talk as examples of avant-garde game art. Jason Rohrer's PASSAGE received frequent allusions, and Rod Humble's MARRIAGE had a few references as well. Cory Arcangel's SUPER MARIO CLOUDS was brought up the most times in terms of video game art that has made it into museum culture. (SUPER MARIO CLOUDS is SUPER MARIO BROTHERS with only the clouds displayed.)

Marcel Duchamp was declared the patron saint of the conference, and referenced more than any other artist. For readers unfamiliar with his work, he famously achieved getting a urinal (Dadaist art titled "Fountain") put into a museum. He did a great deal of work involving skilled technique as well—including "Nude Descending a Staircase, No. 2," a piece many of us first heard of through *Calvin & Hobbes*. Outside the art world, Duchamp was a chess master, once declaring that "all chess players are artists," and most photos of him include a chessboard. Game makers and traditional artists alike seemed thankful to have a common hero.

The third day, each of the artists from a gallery opening the previous night had a chance to share with us their visions for future games. Tale of Tales kicked off the "Not Games" movement. Rohrer created a Gamist Manifesto, suggesting that games should not try too hard to be like films or novels in terms of narrative. The panel of designers afterward was intense, partly because it immediately followed an emotional presentation by Brenda Brathwaite about her current board game project, which is about the struggles of Native Americans.

The last panel put Richard Lemarchand (UNCHARTED 2), John Romero (DOOM, QUAKE), and Harvey Smith (DEUS EX), on stage along with art curator Christiane Paul, moderated by professor Ian Bogost. This was the strongest commercial industry presence of the show. These three ground breaking designers love their work, but it was clear that they love their work primarily as games, not necessarily as art.

The final night, IndieCade and IGDA Atlanta hosted an Indie Game Slam, giving 14 indies each a three-minute window to show off their projects. Jesper Juul and Brian Shurtleff showed their projects from the Global Game Jam, Connor Fallon shared a student game from Carnegie Mellon's Game Creation Society, and Nathan Jerpe demonstrated his ASCII epic LEGERDEMAIN. I took the opportunity to share a few highlights from my 219 daily experimental projects. The Art History of Games crowd responded very well to the original concepts.

The event was a success, from beginning to end. It came together with the right mix of perspectives, backgrounds, and attitudes. More questions were created than resolved, but for this type of gathering, that was the point. The next time this event rolls around, I can only hope attendance will grow, and more people can share in the conversation.

—Chris DeLeon

ANALYSIS: XBOX LIVE INDIE GAMES SALES FOR 2009

GAMERBYTES.COM HAS BEEN KEEPING A close eye on the Xbox Live Indie Games scene and while the service had a bit of a rough beginning, XBLIG now includes a ratings system, avatar support, an entirely new name, and new pricing tiers. Finding the sweet spot for hobbyist and user-submitted indie games has been a long process, but there's definitely been progress.

Larry Hryb, Microsoft's director of programming for Xbox Live recently released a list of the Top 20 XBL Indie Games for 2009 on his majornelson.com blog, and thanks to the participants of the official XNA forums—including many developers—the list has been fleshed out with sales data for the Top 20 games.

James Silva's GAM3 W1TH ZOMB1ES become the top selling game of the year. Like many top tier indie games it features simple play mechanics, an unusual art style, quirky music, and an extremely low price. All these points helped bring the game to the attention of gamers and game blogs.

Applications are also getting a lot of attention on XBL Indie Games. DRUMKIT allows players to take control of their ROCK BAND or GUITAR HERO drum controllers. AQUARIUM HD and MYFISHTANK turn Xbox 360s into habitats for digital fish, while RUMBLE MASSAGE and A PERFECT MASSAGE let users go crazy with the controller's rumble ability. EZMUZE+, a complex audio looping system, made it to the list even at a $10 asking price.

Other games that have done well are usually simple but direct—HEADSHOT and HEADSHOT 2, AVATAR DROP, and THE IMPOSSIBLE GAME, all of which sport very simple concepts, have also made it into the Top 20.

The Xbox Live Indie Games scene has been criticized for less-than-epic game sales, but looking at the numbers helps put this into perspective. SOLAR has sold around 10,000 copies over its nine months on the market. The game took about four months of work during the developer's spare time. For the majority of the time the game was sold at its original $2.50 price point, overall making a bit less than $17,500. That's more than a grand per week for that game's development.

One of the main complaints about the XBLIG space is that people just aren't looking at it. However, the numbers are increasing—55,000 people downloaded the trial to LITTLE RACERS, 26,000 people have played the trial version of AVATAR SNOWBALL fight, and NEXTWAR had 30,000 people give it a go. People are looking at more games, especially those in the top 20 lists. Just having them download the demo is a huge step—that means the premise has piqued their interest, or the box art has made the game look interesting, or that the developer has strong marketing skills.

To expect the same top-line numbers from Xbox Live Indie Games you see on the iPhone's App Store—as some critics do—is short-sighted. The App Store is a different animal. Top games sell up to 30,000 copies a day, but often at bargain basement prices—and the vast majority sell very few.

The release volume for XBLIG is somewhere between Xbox Live Arcade and the App Store. It's worth noting that with only one or two Xbox Live Arcade games released weekly, developers are guaranteed at least a little prominence.

Yet XBLIG titles can get lost in the shuffle swiftly after they move off the "New Releases" page. After that, they can't do much to get back up—or at least, price cuts like those implemented in Apple's App Store seem to have less of an effect.

To succeed developers need to keep the awareness up. Send out press releases to blogs, create trailers, Twitter about it, get on NeoGAF, IndieGames.com, and TIGSource and talk about your games. The more people you get to download your demo, the more purchases you ultimately get. For mid and high-level performers, XBLIG is fast becoming a viable platform for hobbyists and single-man shops to make some cash and get their game seen—and for end users to pick up some genuinely interesting games.

—Ryan Langley

XBOXLIVE indie games — Top 20 Xbox Live Indie Games For 2009

	Name	Sales	Trials	Sales %	Price	Money Made
1	A GAM3 W1TH ZOMB1ES	160,000			$1.00	$112,000
2	Avatar Drop	117,000	433,000	27.02%	$1.00	$81,900
3	RC-AirSim	74,000	450,000	16.44%	$2.50 / $1.00	$129,500
4	Headshot				$1.00	
5	Miner Dig Deep				$2.50 / $1.00	
6	Rumble Massage				$2.50 / $1.00	
7	Who Did I Date Last Night				$1.00	
8	Kodu Game Lab				$5.00	
9	Avatar Golf		259,000		$5.00	
10	Aquarium HD	24,000	172,000	13.95%	$2.50 / $1.00	$42,000
11	myFishtank	22,359	137,506	16.26%	$2.50 / $3.00	$39,128
12	Head Shot 2				$1.00	
13	DrumKit	22,000	200,000	11.00%	$2.50 / $1.00	$38,500
14	A Perfect Massage					
15	The Impossible Game	31,000	115,000	26.96%	$1.00	$21,700
16	Inside Lacrosse's CL2010				$5.00	
17	ZP2K9	16,000			$2.50 / $1.00	$28,000
18	ezmuze+ Hamst3r edition	14,000	93,000	15.05%	$10.00	$98,000
19	The Drinking Game				$1.00	
20	Physics Sandbox				$3.00	

Source: MajorNelson.com, the participants of creators.xna.com

FIGURE 1 The graph shows sales of the top 20 XBLIG games, the amount of trial versions that were downloaded, the conversion percentage from trial to sale, the price and the money made by the developer itself. Note that the money made by a developer on any XBLI game is 70 percent of its selling price—Microsoft picks up 30 percent of each sale.

XBOXLIVE indie games — Other Xbox Live Indie Games Sales Data For 2009

Name	Sales	Trials	Sales %	Price	Money Made
Solar	10,000	38,000	26.32%	$2.50 / $1.00	$17,500
Bricks4Ever	8,042	24,982	32.19%	$2.50 / $1.00	$14,074
NextWar	7,701	30,309	25.41%	$2.50 / $1.00	$13,477
Weapon Of Choice	7,182	97,336	7.38%	$5.00 / $3.00	$25,137
Fireworks HD	6,941	101,145	6.86%	$2.50 / $1.00	$12,147
Groov	6,598	26,019	25.36%	$2.50 / $1.00	$11,547
Scare Me	6,000	26,874	22.33%	$1.00	$4,200
Avatar Wave Snowball	5,549	26,874	13.21%	$3.00	$7,453
Little Racers	3,293	55,492	5.93%	$5.00 / $3.00	$11,526
LED Display	2,516	52,924	4.75%	$1.00	$1,761
Gerbil Physics	2,401	9,009	26.65%	$1.00	$1,681
Dr. Popper	2,383	15,405	15.47%	$2.50 / $1.00	$4,170
Foosball For Two	1,961	26,923	7.28%	$2.50 / $1.00	$3,432
PixelMan	1,906	15,379	12.23%	$2.50 / $1.00	$3,336
Slingstar	1,102	7,137	15.44%	$1.00	$771
ZenHak	907	25,853	3.51%	$5.00 / $3.00	$3,175
Square Off	850	8,300	10.24%	$3.00	$1,785
Being (2009 data)	738	6,519	11.32%	$2.50 / $1.00	$1,292
Wool	751	7,534	9.97%	$1.00	$526

Source: MajorNelson.com, the participants of creators.xna.com

FIGURE 2 Additional indie game sales data gathered from the MajorNelson.com and XNA.com forums.

Check out our work at GDC 2010!
Booth #1910

Sponsored Session
Facial Software Technology
Friday, March 12
3:00-4:00pm **Room:** East Mezz 220/222

Bring your character in for a little work.

Image Metrics combines leading edge facial animation technology and expert support to make characters more expressive.

- **INTEGRATE WITH EASE** > refined animations and non-baked animation curves delivered directly to any 3D package
- **CAPTURE EVERY EXPRESSION** > no markers no makeup, so the actor's performance can be free and dynamic

See how your character performs with our system.
➡ **Free Trial** Call 1.310.656.6565 or visit
www.image-metrics.com/freetrial to make an appointment today.

Recent Patient Referrals Include: God of War® III, NBA2K10, Assassin's Creed® 2, Grand Theft Auto IV, GTA IV: The Lost & Damned, GTA IV: The Ballad of Gay Tony, Devil May Cry 4

image metrics

Superior Facial Animation. Simplified.

RETURN OF THE TRICKSTER

It's time once again to revisit the mighty kludges and well-meaning hacks that are occasionally required to get our games into the hands of eager consumers. As we did last year, we've compiled a host of stories, this year using both voluntary submissions and online comments.

These hacks and heroics span the entire history of computer and video games, even extending laterally into the business software sector. Delight at the ingenuity, marvel at the audaciousness, learn from the mistakes of your predecessors, but most importantly, release games that work—on time, too.

—Brandon Sheffield

CODING TRICKS II

WING COMMANDER.

THANKS FOR PLAYING!

>> Back on the first WING COMMANDER we were getting an exception from our EMM386 memory manager when we exited the game. We'd clear the screen and a single line would print out, something like "EMM386 Memory manager error. Blah blah blah." We had to ship ASAP, so I hex edited the error in the memory manager itself to read "Thank you for playing WING COMMANDER."

—Ken Demarest

100 PERCENT PURE FRUIT JUICES

>> When I first started working in the game industry I spent most of my time shuffling between various small, underfunded startups. Here is a horror story from the good old days when men were men and used DirectX 7.

I worked for a company that had been forced to use a certain 3D engine by the publisher. The engine shall remain nameless, but the publisher had been persuaded to buy a number of licenses for it as part of a bulk licensing deal, and insisted that we use it. To be blunt, the engine did not work, and I spent most of my time at the company making the 3D engine do obvious things correctly, such as fixing the engine company's implementation of single-pass multi-textured lightmapping.

One of the more interesting things that did not work was the BSP compiler. Level designers would build level geometry with correct visibility, and then minor geometry adjustments would break visibility on the other side of the map. To this day, I don't know why this happened, but I believe that the engine's BSP compiler added brushes to the BSP tree in a random order, and certain combinations would just ... randomly break things. At the time, I had never heard of a randomized algorithm, but I invented one regardless—a preprocessing stage was added to the BSP compiler that shuffled the order of the brushes before they were fed to the engine's BSP compiler. That way, if the level geometry broke the BSP compiler, we could just try shuffling the brushes with different random numbers until we found a combination that worked, and then we stuck with it until the next time the BSP compiler broke. The game itself was a disaster, and both the engine and the game were featured in a *Penny Arcade* cartoon that contained the first ever appearance of the Fruitf*cker 2000. This remains a milestone in my personal career.

—Nicholas Vining

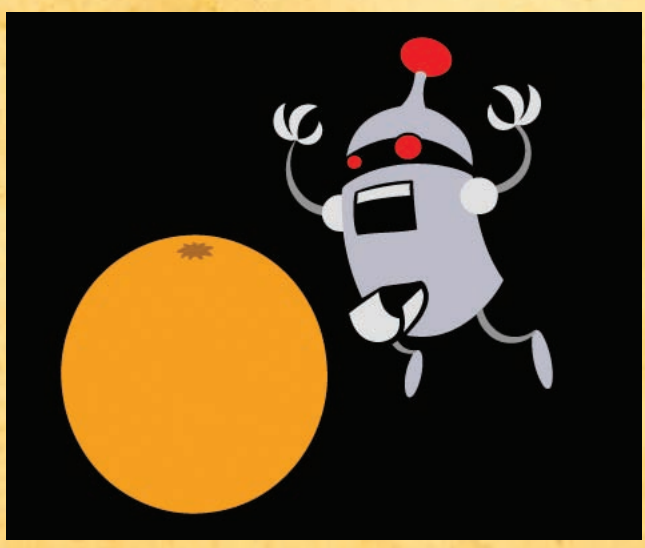


ILLUSTRATION COURTESY OF PENNY ARCADE

FLASH FORWARD

>> I was working on NBA JAM TE for the Genesis, which used a flash chip to store game data. The game had been tested for months, and everything was ready to go, so the publisher ordered 250,000 copies of the cart. But it soon became apparent that no one, for months, had reset the flash chips on the test carts to make sure the flash `init` routines worked correctly. Nor did anyone order any carts for testing.

It was only after all the carts had been ordered that we discovered the flash `init` code was dead, and that the carts could not save games properly! The studio went into meltdown trying to figure out how to ship 250,000 broken carts. Suggestions of production lines adding extra resistors and other hacks to every cart were tried and failed. When all seemed lost, someone figured out if you played the games in an odd, and very specific order, the flash memory would sort of work. So an extra leaflet was added to every box explaining how to use this "feature."

—Chris Kirby

SMOKING SECTION

>> My favorite last-minute hack was in the four-player mode of NITROBIKE PS2. As usual, the level designers and the artists had done their work without the slightest concept of real-world feasibility, and as usual, the job of "finishing" the game was left entirely to me.

After much battling and butting of heads, I got them to simplify the level artwork, create visually occluding features, and remove excessive dynamic objects, but that couldn't save one particular movie-themed level. The level design itself resisted occlusion; it consisted of two large rooms (sound stages, in the fiction) with no occluding features, connected by two open doors. One door was in the middle of the wall. There was no possible way to arrange occluding polygons to block one room from seeing the other room in any effective way, and there was no way to cut out any more level artwork without completely ruining the character of the level.

I really needed a way to put one big occluding polygon in the wall, covering

the middle door. That's when it hit me ... a particle-based "wall of smoke" covering the middle door would fit in well with a movie-themed level, and would completely solve my problem. A wall of smoke could be driven through, but not seen through, and would conceal the existence of my occluding polygon! I was able to commandeer one artist to create such an emitter, and one level designer to place emitters on either side of the door, both facing each other. Finally, in between, I put one very large occluding polygon. It looked good, and fixed the last performance problem I had with the levels.

—Steven Boswell II

... OR IS IT OVER HERE!?

>> I've been writing games for over 20 years, and was the recent global technical director for THQ, so I've seen a lot of terrible hacks. But there's one in particular that I still laugh at, going back to Beam Software in the early 1990s.

In those days, before nice IDEs and smart compilers, we used to write all games in assembly language. All the `.s` files contained the respective assembler code for a particular part of the game; `creature1.s`, `collision.s`, `controls.s`, and so forth.

We used makefiles too, where the programmer would create a new `.s` file, and place it after everything else in the makefile. The idea is that you'd type "make" on the command-line, and the assembler would assemble each file into a new `.o` and then the linker would link all of them together to build the final executable.

We had one programmer who notoriously wrote buggy code that would typically stomp on some random piece of memory. Usually buffer overruns. He would spend some time trying to find the bugs, but when he couldn't, he would ... get this ... reorder the makefile so the files statically linked in a different order in memory!

That just meant the piece of memory being randomly written over was now somewhere else, but by pure luck, the game wouldn't crash, either. He kept doing this until basically all permutations of the makefile resulted in a crash.

At the 11th hour when the game was about to ship, he solved the problem. He simply kept creating new `.s` files filled with little pieces of data, which he inserted into random places in the makefile until it somehow

World Series of Poker 2008.

didn't crash—then shipped it! There are at least two games (for the Game Boy) released during that time period that use this "technique."

—Shane Stevens

SUB-STANDARD

>> We were trying to ship World Series Of Poker 2008, which was our first PlayStation 3 game. The PS3 allows several different screen resolutions, and two screen aspect ratios. We had designed a widescreen 2D shell, but didn't have the time or resources to make a standard-definition 2D shell. I scoured the TRCs, and couldn't find any reason that letterboxing wasn't allowed. So our standard-definition view was simply our widescreen view with black bars above and below the picture.

The publisher tried desperately to invent TRCs out of thin air to keep us from doing this, but eventually ran out of ideas, and we went ahead with it. Besides, only a few hours after I bought my own PS3 and played it on my standard-definition TV, I started shopping for a widescreen TV. I doubt many people connect their high-tech PS3 to a low-tech tube TV anyway!

—Steven Boswell II

CONSTANT CRAVING

>> For one reason or another, there was this huge incompatibility between an existing code base, and a new (approximately) zillion lines of code that had been written to use the first set of code libraries. The first set of code was written by people who were more into the idea of "safe" programming, making it as strict and restrictive as possible, to avoid errors. So they used a C / C++ language feature called CONST. CONST means "constant," and makes sure that read-only variables can't be changed inside certain functions. CONST and non-CONST code are not compatible and cause the compiler to barf.

The team decided to just take a hacksaw to the code and performed this clever little trick:

```
#define const
```

which redefines `const` to be.... NOTHING. empty space. So:

```
const int x;
```

when pre-processed for compiling becomes:

```
int x;
```

This is the equivalent of buying a car, taking two of its tires off, and using it as a motorcycle.

Personally, I hate CONST with a passion, so I like this trick. But to some people this is like taking a pair of scissors to your seat belt.

—Anonymous

REALITY BITES

>> We had a bug on a PlayStation 3 Unreal Engine 3 title (it was the first PS3 UE3 title that shipped) where in debug, we'd inexplicably get a crash in `printf()` when connecting to a multiplayer game. The client in debug would print out the hashes of every content package the server told it to load, and apparently one of them (on this build) had a `%` in the hash. We couldn't figure out a reasonable fix for it, but `#ifndef` PS3 worked just fine until the next data build, when it disappeared.

About a year later on the next project I ran into exactly the same bug. I used exactly the same fix.

The worst one was actually after ship, when we were doing a content patch. The way our DLC/patching worked, we couldn't patch any compiled UnrealScript, but there was a bug where two RPC calls had not been marked "reliable," meaning that the packets to invoke them are resent until ACKed. As a result, under reasonably crappy network conditions, the functions to toggle readiness and voice status in the lobby would sometimes not get called. But marking RPCs as "reliable" needs to be done in UnrealScript, and we couldn't patch UnrealScript. So, at startup, we looped through all loaded `UFunction` objects (C++ representation of

KILLER TRACKS™

GDC ATTENDEES!
Visit Killer Tracks at GDC in BOOTH 2116 to receive a complimentary copy of our new Video Game Music Sampler on USB Flash Drive!

KILLER Video Game Music

Perfect For:
- Action
- Fantasy Adventure
- Shooter
- Strategy / Military
- Sports
- Music Games
- Trailers & Cinematics

AUDITION THOUSANDS OF MIND-BLOWING MUSIC TRACKS ONLINE AT:

KILLERTRACKS.COM

CODING TRICKS II

script function), did a string compare on the name, and set the "reliable" flag on those two. Worked great.

—Anonymous (taken from Reddit.com replies to the original Dirty Coding Tricks article)

YOUR FLY IS DOWN

>> Xbox Live Arcade games on the original Xbox needed to be packed entirely inside the .xex file. To accomplish this we stored our data in a .zip file embedded as a data section in the executable. As the file grew, it soon became impossible to load the data section into memory, and allocate enough memory to unzip it, and pull out the needed file.

To remedy this, I implemented some code which read the PE header for the executable as soon as the game loaded, and noted the offsets for data sections. That way, the file stream which was reading the executable could just skip to the offsets for the different zip files, and stream straight out of the executable without ever loading the data sections into memory.

—Pat Wilson

CAMERA OBSCURA

>> This is an older one, but FORCE 21 was an early 3D RTS which used a follow cam to observe your current platoon. Toward the end of the project we had a strange bug where the camera would stop following the platoon—it would just stay where it was while your platoon moved on, and nothing would budge it. The apparent cause was random because we couldn't find a decent repro case. This went on until, finally, one of the testers noticed that it happened more often when an air strike occurred near your vehicles. Using that info I was able to track it down.

Because the camera was using velocity and acceleration and could be collided with, I had derived it from our `PhysicalObject` class, which had those characteristics. It also had another characteristic: `PhysicalObjects` could take damage. The air strikes did enough damage in a large enough radius that they were quite literally "killing" the camera.

I did fix the bug by ensuring that cameras couldn't take damage, but just to be sure, I boosted their armor and hit points to ridiculous levels. I believe I can safely say we had the toughest camera in any game.

—Jim Van Verth

HEX APPEAL

>> I was a tester for THE NEW TETRIS on the N64. There was a crash that I could reproduce every time, which would display a dump of the registers just before locking up. You had to power cycle the N64 to get it to go away: even the reset key was unresponsive. Version after version, the developer said the bug was fixed, and version after version I reproduced it.

Closing in on the shipping deadline, the developer had to close out all crashing bugs in order to ship. (Testing is done by Nintendo even on third-party games, and Nintendo has to approve it.) But this bug would just not go away.

FORCE 21.

The game also had some unrelated secret codes you could enter to unlock various things. One day I joked that the developer should replace the hex dump screen with a screen that says "Congratulations! You have discovered a secret code! Turn your console off and back on, then enter the username HALUCI."

So he did. And that's how it shipped.

—Anonymous (taken from Reddit.com replies to the original Dirty Coding Tricks article)

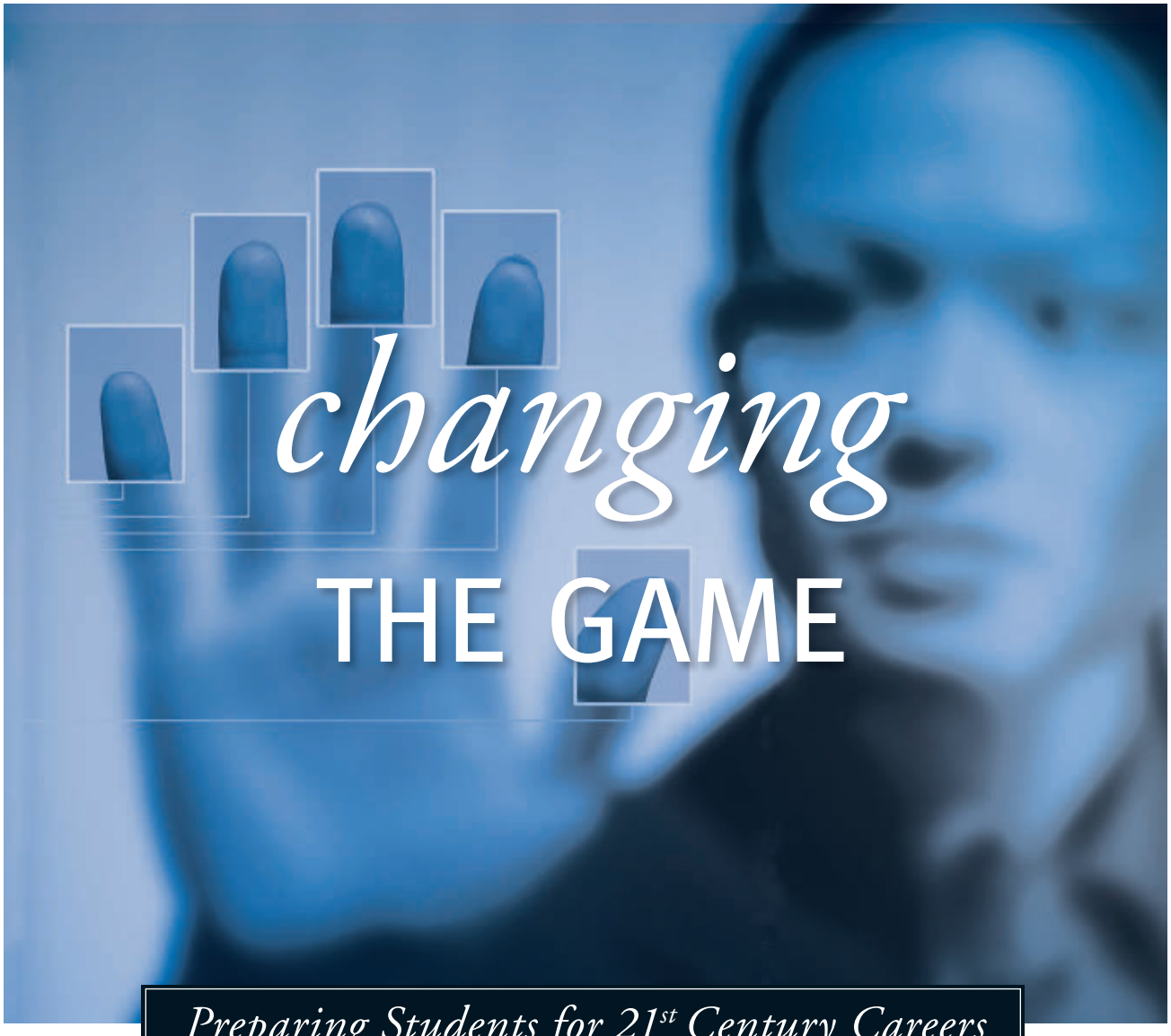
SHORTSTACK

>> I was one of a few interns at IMAGIC in 1982–83, and back then we were all doing Intellivision carts. One of the programmers had to leave to go back to school, and I was chosen to fix the random crash bug in his game. It turned out to be a stack overflow in the timer interrupt handler. Since the only reason for the handler was to update the display of the on-screen timer, I added some code to test the depth of the stack at the beginning of the interrupt routine. If we were in danger of overflowing the stack, it would return without doing anything. Since the handler was called multiple times per second, the player never noticed, and the crash was fixed.

—Anonymous (taken from Gamasutra.com replies to the original Dirty Coding Tricks article)

BACK TO BASICS

>> My cohort Mike Mika and I were porting KLAX, the arcade tile matching game, to Game Boy Color. It was a fun, intense, six-week project to bring one of our favorite games to the system. We had the C source code (which



Preparing Students for 21st Century Careers

DeVry University's bachelor's degree program in Game & Simulation Programming (GSP) positions students for success with an innovative experiential education.

Our graduates are well-rounded and ready to make an impact on today's ever-changing and demanding simulation and video game industries. The GSP curriculum includes training in a broad range of programming languages and software applications. These degree-related courses are integrated with a general education curriculum to reinforce essential critical thinking and communication skills.

Discover education working at devry.edu.
Visit us at The Game Developers Conference, booth #2534
March 9–13, 2010 | Moscone Center | San Francisco, CA

DeVry University

Program availability varies by location. **In New York, DeVry University operates as DeVry College of New York.**
DeVry is certified to operate by the State Council of High Education for Virginia. AC0060

CODING TRICKS II

OUT OF THE GAME

Games aren't the only place where coding hacks can save the day. Here are two non-game submissions that were too enjoyable not to include.

WINDOW WASHERS

>> Five years ago I was working as programmer in the video surveillance software industry on some very sensitive and complex security software. We had a great product that worked well, and the most difficult part of this software was that it involved the display of 50 video streams onscreen at the same time. The software needed a huge chunk of memory to work, and it was supposed to be up 24/7. We first shipped to our beta customers a few weeks before general deployment. A week later we noticed there was a huge memory leak—about 4KB per minute. I spent a couple days investigating the leak with no results, and there was no time to fix it before shipping. Memory was a key part of the software, and a leak of that size would completely kill the application.

Doing some testing (under Windows), I had to hide the software window to go back to the coding window, and noticed a huge drop in memory when I did so. I then remembered that when you put the window in the notification area or in the start-menu bar, Windows immediately reclaims unused/freed memory. Here was our chance!

I wound up adding a timer in the application that every couple of minutes would put the window in the start-menu bar, and display it fullscreen right after. It looked like a blink on screen, but it worked! We were able to ship after that, giving us some more time to fix the bug (which we found a few days later—some window handle was not properly cleaned).

—Yohan Launay

RESULTS MAY VARY

>> Back in the 1970s I was working on a banking system with a team using a long-gone programming language known as MPL2. This language had a restriction of 256 global variables and, since all were in use, adding new features to the system often meant searching for a variable to free up or use for two different purposes in different parts of the code. It was a risky and a time-consuming process. Within the program each function could have its own 256 variables limited to the scope of that function.

At home one night a revelation came into my mind. The next morning I proposed to my boss that we wrap the entire code in a function. We would then have 256 global variables of the program available to us, as the current 256 were now safe in the "inner" function. The program itself would just declare some more variables then call the function, which was the entire original program, but now able to "see" not only its own embedded 256 variables but also the newly available 256 global variables.

My boss was skeptical but allowed me the two hour compile window to try it, and was dumbfounded when it worked, overcoming what had been a problem for the team for a couple of years.

—Rob Hindle

was just ESCAPE FROM THE PLANET OF THE ROBOT MONSTERS with a lot of robot monsters commented out and KLAX put in), and we had chatted a lot with Dave Akers, the original arcade programmer, who had coded the prototype in Amiga BASIC over a weekend and ported it to C in something like one day. We coded the game in straight Z80 assembly. There was lots of fun stuff like copying code to a white board and stepping through it one line at a time mentally, updating the memory contents on another white board, because we didn't have a real debugger. Good times.

Anyway, we got to crunch time and everything was working great, I was playing the arcade version and testing the GBC version when I ran into a weird edge case scoring bug. I don't remember the actual case, but it involved something like doing a big cross which dissolved into some diagonals. Anyway, it scored wrong on the GBC compared to the arcade machine. Needless to say, I discovered this at around 11:30 at night (right before a milestone). We ran through the code a million times, compared our assembly to the arcade C code, and the bug just didn't make sense. We were scoring logically, and our code was doing the same thing in the same order as the C, which was just a line-by-line translation of what Dave had originally done in Amiga BASIC. (I suspect Dave was a little like Mike—great at assembly and BASIC, but not a huge C fan 20 years ago when KLAX was done.)

Finally, around 5am, after smashing our heads against this all night, we had an idea—not something we thought would work, but at least something worth trying. Mike coded the scoring system in Quick BASIC, and it scored just like the arcade machine. So we translated the BASIC, line by line to Z80 assembly. It worked. God knows why, but it behaved like the arcade machine (it might have something to do with the fact the original was coded in BASIC). We sent to build to Atari, printed out both versions of the code and went to Denny's. We looked at the code for an hour over breakfast and still had no idea why it behaved differently. We both swear today that the code should generate identical results! But sometimes when it gets late, you have to go to the voodoo programming!

—Chris Charla

MATTHEW S. BURNS

TAMING THE WILD WEST

PRODUCTION METHODS AT LEADING DEVELOPERS

Several years ago it was still common to hear comparisons of the game industry to the Wild West, in reference to its wide-open possibilities and anything-goes atmosphere. But while the pioneering spirit that has characterized game development remains one of its most exciting aspects, the newness of the medium has also lead to great pain over the years in the form of poor organization, unrealistic schedules, and mismanaged projects.

The big studios that have survived and thrived in this environment have consistently risen to the challenge of shipping high-quality titles. While there is good reason to believe it will always be something of an inexact science, an important body of knowledge has been growing about how big games are made.

In order to gauge how this knowledge is put in practice in the industry today, I spoke to production departments of three large but very different developers in order to get an overview of their approach and to see what each might have in common.

FACILITATING THE VISION AT HARMONIX

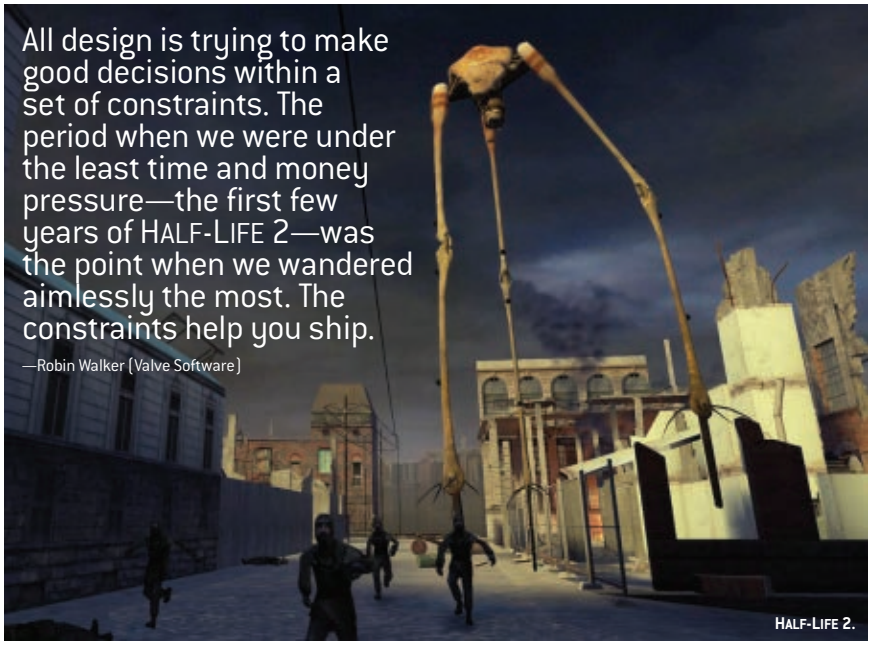
Harmonix Music Systems, creator of seminal music titles FREQUENCY, GUITAR HERO, and ROCK BAND, has recently experienced extremely rapid growth, expanding from under 100 staff to over 300 in just the last two years. The transformation of this Cambridge, Massachusetts company will sound familiar to many producers working in the industry today.

"In the 'good old days,' when we were very small, we managed using [Microsoft] Project and Excel, made traditional waterfall schedules and just printed out task lists for people, and iteration more or less happened automatically," says Tracy Rosenthal-Newsom, vice president of production.

Today, however, the studio is busily engaged in multiple projects of varying size and scope with a total of around 30 to 35 production staff. The engineering and audio departments are shared by the company, and each has a dedicated production team, while other departments are organized by project. The company also has to juggle internal development and external development with partner studios. With the size and complexity of Harmonix' business now, the studio knew that the all-important iterative process needed to live inside some structure.

SPLIT PERSONALITIES

» Harmonix approaches production at the project leadership level by pairing a creative vision holder with a project manager. "Early on, (creative vision and project management were) all handled by a single person, but over time we



HALF-LIFE 2.


CALL OF DUTY 3.


GUITAR HERO.

determined that splitting the roles was going to benefit the process," says Rosenthal-Newsom. While the primary job of the production lead is to facilitate the execution of creative decisions, he or she also needs at least some creative sense in order to understand what's important about the company's games in order to approach problems with the right mindset and make appropriate decisions.

The differences in the two roles essentially come down to a matter of priorities, with the producer or project manager continually evaluating the situation with schedule and budget in mind. At times, this leads to some pushback and discussion, but as Mike Verrette, director of production notes, constraints are important because they keep the team focused and can lead to creative problem-solving.

"Everyone's heard of the triangle of budget, quality, and schedule, and the saying, 'Pick two.' But I don't think it's a given that you need to specifically give one of those up," says Verrette. "We really try our best to achieve all three."

INTRODUCING CONSISTENCY

» As the studio added people over the last several years, production practices began to vary as new employees brought their own training and experience into the mix. The studio's production know-how grew organically as a result of these different styles, but it also meant inter-project communication was not as good as it could have been and employees were sometimes confused when transitioning from one project to another.

Harmonix has recently started working to standardize its production methods. One large part of that effort has been the studio-wide adoption of Swedish company Hansoft AB's eponymous server-based project management software, which the company began working into its process early this year.

"For years we'd been talking about wanting to standardize all our schedules on a network," says Rosenthal-Newsom, "and we found that Hansoft provides the kinds of features we needed."

Hansoft allows Harmonix' schedules and task lists to be pushed to the team automatically through its client application, which runs on every contributor's computer. It also provides a useful framework for communication and collaboration for a large team: The engineering department's daily stand-up meeting of over thirty participants was recently replaced by virtual coordination using the software, for example.

"It took several months for everyone in the studio to transition over to it because it required some training and education," explains Verrette, "but now we're using it exclusively. It gives us a lot of flexibility in methodology—a producer can enter in a very pre-planned, waterfall-style long-term project schedule but organize by milestone and look at it from the perspective of an agile-style burndown chart, too."

HYBRID APPROACH

» Verrette feels the production style that evolved at Harmonix isn't one hundred percent what project management gurus call "agile," though. For example, the studio does not work in small feature-oriented teams exclusively. "We will sometimes use a 'strike team' process for specific features when it makes sense," he says, "but other aspects of production, such as art assets, make sense to be tracked as a linear process."

The team also course-corrects immediately instead of on a per-sprint basis, and production still drives projects in terms of their priorities and schedules. "I see us continuing to take a hybrid approach," says Verrette.

AGENDA-SETTING AT TREYARCH

Part of the Activision Blizzard family, and recently known for CALL OF DUTY: WORLD AT WAR, Santa Monica, California-based Treyarch makes use of a strong production department that plays a central role in getting its games to quality expectations and shipped on time. "I would say that production sets the agenda based on the creative vision, and [on] what needs to happen to execute that vision," says Pat Dwyer, senior producer at the studio. In other words, while Treyarch's producers do not make choices about what the creative vision actually is, the department does take charge of creating and carrying out the development plan, and takes responsibility for delivering the final product.

MARCHING ORDERS

» Because the company's teams tend to be large (on the order of between one and two hundred members), the time and resources spent on organization can quickly balloon. At the height of production, "you could have anywhere from ten to fourteen people meeting to discuss a single

level," Dwyer explains. Like Harmonix, the studio has a ratio of about one production staff member to every ten other team members, organized by a hierarchy that divides major departments and sections of the game into areas of responsibility for senior producers, producers, and associate producers. Each must be convinced that the elements under his or her purview are on the right track or otherwise raise a red flag.

But while production is usually the final word when it comes to scope and features, it does not mean producers have license to dictate aspects of the game. Instead, Treyarch has chosen to centralize creative decision-making under a single creative director who advances the overall vision for the game, accepts or synthesizes ideas pitched by individual team members, and can step in to mediate or arbitrate disputes about game design or aesthetics.

BEING ADAPTABLE

» Having a well-defined production process doesn't mean unexpected innovation can't also occur during the course of a game's development, though. The popular "Nazi Zombies" mode in CALL OF DUTY: WORLD AT WAR, for example, was originally prototyped by a few enthusiastic team members on their own time. The mode steadily gained converts who pitched in with their own contributions to get the prototype off the ground. Production recognized its potential value as a bonus and made the call to dedicate official resources to it. When it shipped, it was one of the game's most talked-about features.

Treyarch's approach to production has evolved in recent years as the company has been given more time and resources to put into its games. While its previous CALL OF DUTY projects, CALL OF DUTY 2: BIG RED ONE and CALL OF DUTY 3, were each completed in a single year, WORLD AT WAR was the result of two years of development—a change in approach that the studio clearly prefers and seems likely to continue. Because of this, day-to-day production tracking at Treyarch has become less granular and top-down, and more flexible with respect to tasks and scheduling. "We try not to get hung up on half-days anymore," says Dwyer.

For the studio's next major title, the paper design stage was reduced, both in duration and in the number of people involved, in favor of getting rough geometry into the engine quickly, and beginning iteration as soon as possible. The production team has also begun incorporating some agile methodology into its processes, most notably for the campaign levels, which undergo weekly reviews. At each review meeting, a task list is generated for the following week, usually tabulated and tracked by an associate producer. The week-to-week scheduling allows for a level of quality not possible before. "On our earlier projects, there was no way to make a qualitative judgment on things because our focus was all about eliminating ship risk," Dwyer says. "Now we have the time to make those calls, and we can iterate on what we have."

At the same time, Treyarch feels the full-on agile approach is not always appropriate due to the lead times necessary to get certain assets such as outsourced models or motion capture data, favoring its use in combination with other techniques.

KEEPING TABS

» The production team at Treyarch keeps actual tracking techniques surprisingly simple for such a large operation, relying mostly on to-do lists in internal Wikis, publicly-displayed whiteboards with magnetized cards upon which tasks are written, and longer-term project plans laid out in Excel

spreadsheets. The combined experience of the team helps these methods' effectiveness: WORLD AT WAR was Treyarch's third consecutive CALL OF DUTY game, and each successive project's schedules and estimates have benefitted in accuracy from the knowledge gained during the last.

DISTRIBUTED DECISION-MAKING AT VALVE

Valve Software in Bellevue, Washington stands out from the crowd for myriad reasons, but one of the lesser well-known ones is that as a studio, it employs no dedicated producers. Instead, Valve works in a cooperative, adaptable way that is difficult to explain to people who are used to the top-down, hierarchical management at most other large game developers. "If you took people and had them observe us for six months to figure out our process, it would probably just look like this barely organized chaos," longtime Valve employee Erik Johnson acknowledges.

But that doesn't mean there isn't method to the madness. At its root, Valve simply doesn't distinguish between "creative" and "production" tasks as many other studios do. Instead of separating the work and the accountability for that work between two different people, they treat them as one and the same.

Valve believes that the efficiency lost to having each team member act essentially as his or her own producer is worth the expense in quality gained: "If we tell an artist, 'Your job is to figure out what to do on any given day,' then yes, he may spend twenty percent or more of his time not working on art—but the art that he does produce will have more value to its customers," Johnson says.

THE WISDOM OF CROWDS

>> Valve largely trusts that its employees are making the right decisions on their own, and that necessitates an awareness and tolerance of the risks inherent to that approach. "When we make mistakes, they tend to be big mistakes. We expect everyone to be doing a good job and making good choices, and it can take us a while to realize if that's not working out," Johnson says.

The coordination of effort in the absence of a single master plan is governed by the team's social rules. "There's a perception that we have this unconstrained process where people can just do whatever they want," says Robin Walker, another Valve veteran, "but that's pretty far from the truth."

For one thing, Valve believes strongly that no single person should be making decisions, and that involving groups increases the chances of a good decision being made. It's such a pervasive feeling that, according to Johnson, "no one at Valve wants to be the one single guy making decisions, because we know you're going to be wrong more often than if you made decisions together. Part of the process of ensuring you're making the right decision will necessarily involve other relevant parties."

LEFT 4 DEAD 2.

Valve also has strong principles about the assignment of work. For example, whoever is designing a system should be the one to build that system—no handing off of paper designs. The converse is also held to be true: the people doing the work should be the ones making the decisions about what to build and how to build it. In this sense, Valve's employees could be said to comprise a kind of decision market, where multiple agents, each acting more or less independently, cast their "vote" by spending time on the features most important to them.

TESTING FOR SUCCESS

>> One of the strongest driving and steering forces of a game's production at Valve is the company's relentless focus on playtesting. The tests are designed to objectively measure the game's current state against its desired goals, and their weekly frequency allows for detailed tracking of movement in the right (or wrong) direction as adjustments are made. If the team finds itself at odds over certain choices, the playtest is a large component of how the dispute is resolved.

"The goal isn't to get my particular idea in the game—the goal is to be right," says Walker.

After each playtest, the team's next short-term tasks become clear. "We're not trying to define what we're shipping a year and a half from now—we're making decisions about what we're implementing for next week's playtest. One of the ways I measure how long until we ship is how many pages of notes I'm taking from each session." This piece-by-piece, week-by-week construction method means that the team's end product could never have been prescribed by a declamatory blueprint in the early stages.

"We can't plan more than three months out. We think about those things, of course, but we know our chances of being wrong are high," Walker explains.

WHAT WOULD VALVE DO?

>> But is Valve a complete exception in a world where most of us labor under significant time or budgetary pressures? Or do its methods hold applicability to other developers? Walker believes the latter. "All design is trying to make good decisions within a set of constraints," he says, echoing the sentiment expressed by Harmonix' Verrette that limitations, when set properly, are good for the process. "The period when we were under the least time and money pressure—the first few years of HALF-LIFE 2—was the point when we wandered aimlessly the most. The constraints help you ship." He points to the release of LEFT 4 DEAD 2 a year after the original as an example of a team's decision to work under a strict and self-imposed deadline.

DRAWING CONCLUSIONS

>> While each developer's practices were home-grown and adapted over the course of several games to fit their varying situations and goals, there are some common threads running through these very different studios' approaches:

"HYBRID" METHODOLOGY The studios described their process as incorporating some agile methods, but none of them felt a full-on agile system like Scrum was applicable to the entire team. All three developers spoke of iterative process on certain aspects of the game, while Harmonix and Treyarch mentioned tracking the production of easily quantifiable pieces such as art assets in a waterfall-style manner.

DECISION STRUCTURE The developers found that a strong decision-making structure was necessary to their efforts. While the makeup of the structure varied—the pairing of a creative director with a lead producer in Harmonix' case, or an internally-reinforced distributed system at Valve—all spoke of maintaining a well-understood framework by which decisions were made.

PROCESS FLEXIBILITY Finally, the studios stressed that no fixed project management dogma is going to be the best for all the situations one encounters in game development. Harmonix' Verrette sums it up this way: "This is a cliché, but it's true: there is no silver bullet. You build a toolbox and learn how to use it, and eventually you have the right tool for the right situation."

Although game development studios continue to struggle against challenges unique to them, the maturing of practices as these and other studios across the industry suggest that the "Wild West" metaphor may soon no longer apply.

MATTHEW S. BURNS *is the founder of Shadegrown Games and formerly a producer at Bungie where he worked on the HALO series. E-mail him at* ***mburns@shadegrowngames.com****.*

Introducing Vision Engine 8

- Now optimized for performance on browsers, PS3, Xbox 360, PC and Wii

- Superior graphics power across genres and platforms

- Unrestrictive workflow, including many 3rd party integrations and built-in editors

- Flexible licensing to meet every project's needs. Free from royalties and back-end fees

TRINIGY

www.trinigy.net
www.sneakpeek8.com

Experience the freedom.
Visit us at GDC 2010, Booth 1334.


UNCHA
AMONG

RTED 2

postmortem

THIEVES

THIS IS THE STORY OF THE QUEST FOR A MYTHICAL TREASURE: THE HIDDEN gold of the ultimate cinematic action game. When Naughty Dog set out on our latest voyage at the end of 2007, we had a map of the territory in the form of our previous PlayStation 3 project, the contemporary pulp adventure UNCHARTED: DRAKE'S FORTUNE. We had a strong team of willing adventurers, and we knew we wanted to make an actively playable "summer blockbuster" that would keep the player in moment-to-moment control as much as possible. What transpired in the following 22 months was as much an adventure for the team at Naughty Dog as it is for players who follow our game's hero, Nathan Drake, into the world of UNCHARTED.

Naughty Dog, creator of the CRASH BANDICOOT and JAK AND DAXTER series of games, is a developer of around 90 people based in Santa Monica, California. It's an unusual studio in several ways. We don't have any dedicated producers in-house, and no one at the company, not even our studio co-presidents, only does management work. Everyone contributes directly to the creation of the game. Our games are produced by the game directors, game designers, discipline leads, and others who work with them. In fact, anyone who wants to take responsibility for some aspect of a game is encouraged to simply start organizing or building. This attitude naturally creates a meritocracy at Naughty Dog, as responsibility flows toward those who are doing the best job of keeping us on the right track and to those who make the most significant contributions to building out a given game.

As with every project that we set out to work on, we established a short list of design goals for UNCHARTED 2. We wanted to improve on the foundation we'd laid with UNCHARTED: DRAKE'S FORTUNE by creating a tighter mesh of character-driven story and adventure gameplay. Some of the sequences from UNCHARTED: DRAKE'S FORTUNE suggested to us that we were only beginning to scratch the surface of what was possible in terms of playable cinema. In particular, we knew that we could make our set pieces more epic with the application of more creativity and elbow grease.

We wanted to bring a larger cast of characters into play to act as "emotional satellites" and show Nathan Drake as more than the straightforward hero. We also wanted UNCHARTED 2 to work as a standalone adventure. We wanted the game to have a multiplayer component, and we knew we would be making a lot of changes and additions to our engine.

WHAT WENT RIGHT

1) CREATING AN ACTIVE CINEMATIC EXPERIENCE. One short sequence from UNCHARTED: DRAKE'S FORTUNE kept our attention: soon after Drake crash-landed on the island, a truckload of pirates pulls up on a broken bridge. When the player shoots a nearby explosive, the truck flips and falls, forming a bridge that opens the way. It was a simple set piece, but it played out almost entirely in gameplay and it got us thinking—what if we could do more to capture the flavor of the most impressive moments from our favorite adventure movies without taking control away from the player? This led us to what we came to call the "active cinematic experience" of UNCHARTED 2: AMONG THIEVES.

It turned out to be enormously time-consuming to put together scripted set pieces of that scale and complexity, but we started work on them early and staffed up to make sure we could get everything done on time. Even then, everyone involved had to work incredibly hard to make the sequences playable and polished.

We had an idea for a sequence where Drake climbs and fights along a train as it speeds through the countryside that we didn't want to fake. We wanted a chain of train cars moving continuously through a streaming series of environments, and the only way to do this was to create a major new piece of technology: our dynamic object traversal system. We had to give our player-character and all his allies and enemies the ability to use all their moves and abilities on any dynamic object in the game.

This was an enormous technical hurdle, and involved the re-implementation of many of the game's core systems, so we started work on it near the beginning of the project. The effort was well worth it. It paid dividends throughout the game—from major set pieces like the collapsing hotel in our fictional Nepalese city to incidental stumbles along collapsing balconies.

UNCHARTED 2

postmortem

We liked the "wide linear" gameplay style we used in UNCHARTED: DRAKE'S FORTUNE; our story is essentially linear, but the player has a good deal of choice about how they can tackle their moment-by-moment experience, especially in terms of combat. We decided to expand on that in UNCHARTED 2 through the use of "stealth action" mechanics, which meant more player abilities in support of sneaking around, an expanded repertoire of surprise melee attacks from behind and below for Drake, and "investigate" and "hunt" AI routines that allow enemies to search for Drake if they think they've seen him.

We used sequences of play with Drake's NPC allies to set up and pay off fairly complex emotions at carefully planned intervals throughout the game. We also did something similar, if emotionally simpler and more conventional for games, with climactic confrontations between Drake and vehicular enemies like helicopters and a tank.

We continued to improve the production pipeline for performance capture that we had developed for UNCHARTED: DRAKE'S FORTUNE, and we were able to do live capture of audio on the mocap stage for the first time. We made sure to keep our strong focus on the creative involvement of our terrific cast and superlative mocap director, Gordon Hunt, and people have responded positively to the character-driven story that resulted. One of the highest compliments that we get paid is that people like to watch their friends playing the UNCHARTED games almost as much as they like to play themselves—we even used this idea in one of our television commercials.

2) GETTING ON WITH MAKING THE GAME. The most important lesson that we took away from making UNCHARTED: DRAKE'S FORTUNE was one we had to learn the hard way. We spent too long making plans and not enough time simply getting on and building things. This led to a crisis that resulted in a mid-project production reset. We'd lost sight of the fact that theorizing about process and tools can only take you so far, and that it's only when you build something—whether it's a game mechanic, a tool, or a level—that you make the really valuable discoveries about what you're doing.

When we set out to make UNCHARTED 2, we kept this idea at the forefront of our minds the whole time and it served us well. For example, we shifted our level design process away from paper layout and toward iterating on prototype levels in simple "blockmesh" geometry. Our game director and one of our game designers would first sketch out an experiential flow for the player. The designer would rapidly build out an environment with a low level of detail to test on other team members, so we could see how navigable it was, what camera and line-of-sight issues arose, how long the experience would last, and so on. We would then start scripting interactive objects and placing enemies, and eventually give our art team the all clear to begin creating final art. This approach let us build out the game's footprint very quickly, although it wasn't without some dangers: we ran the constant risk of becoming too committed to level designs that might need changes demanded by the maturing story.

We thought on our feet about the order in which we should tackle our new and expanded gameplay systems, and started with the ones that would have the most wide-reaching effect on the game. We took a similar approach with our tools and engine improvements, tackling the things that would give us the biggest leg-up first. By the time we finished with all the changes and improvements we made, we felt like we'd virtually reinvented our engine, and therefore dubbed it the "Naughty Dog engine 2.0."

3) MULTIPLAYER METHODOLOGY. We decided early on in the development of UNCHARTED 2 that we wanted our game to have a multiplayer component. We made the right decisions at the right times to make this happen, beginning with getting the attention of the right people on the team.

We wrote our networking code in-house, which gave us a solid base to work with. Near the start of the project we put one of our gameplay programmers in charge of multiplayer on a full-time basis. We hired a dedicated multiplayer designer in August of 2008, which meant that we had someone championing the multiplayer experience during the most important phase of its development. We later hired a co-op designer.

We essentially took Nathan Drake's move set

Unreal Technology News

by Mark Rein, Epic Games, Inc.

Canadian-born Mark Rein is vice president and co-founder of Epic Games based in Cary, North Carolina.

Epic's Unreal Engine 3 won Game Developer magazine's Best Engine Front Line Award for three consecutive years, was inducted into the Hall of Fame in 2008, and won Best Engine again last year.

Epic's internally developed titles include the 2006 Game of the Year "Gears of War" for Xbox 360 and PC; "Unreal Tournament 3" for PC, PlayStation 3 and Xbox 360; and "Gears of War 2" for Xbox 360.

Upcoming Epic Attended Events:

Game Connection
San Francisco, CA
March 8-10, 2010

GDC 2010
San Francisco, CA
March 9-13, 2010

Triangle Game Conference
Raleigh, NC
April 7-8, 2010

E3 2010
Los Angeles, CA
June 15-17, 2010

Please email:
mrein@epicgames.com
for appointments.

POWERED BY UNREAL TECHNOLOGY

BIOWARE DISCUSSES THE DEVELOPMENT OF *MASS EFFECT 2* WITH UNREAL ENGINE 3

When creating the sequel to *Mass Effect*, BioWare focused on optimizing Unreal Engine 3 base technology to create a more immersive sci-fi RPG experience. Over 150 people at BioWare worked on *Mass Effect 2*, which has been honored as one of the most highly rated Xbox 360 games of all time. Having worked with UE3 on the original *Mass Effect*, lead producer Casey Hudson said his team pushed every aspect of the sequel forward from both a technology and gameplay perspective.

BioWare's critically acclaimed *Mass Effect 2*

"Having shipped the game on Unreal with a *Mass Effect* total framework in place, we looked at what our final performance memory budget was and billed *Mass Effect 2* to that budget," explained Hudson. "We didn't have the opportunity to do that in the first game, so that helped us to better develop content. We also were able to look at where we were spending the most time on the least effective tasks. So it's not that we're using more of the CPU, it's just that we look at things like the previz phase, for example, in scale form and we rewrote our code for that. We just found little opportunities where we were surprised at how much time we were spending in the wrong places like you do in any normal game development process."

Hudson said his team utilized Unreal Matinee and Unreal Kismet to improve the player experience in *Mass Effect 2*.

"Matinee is really integrated into the way we build a lot of our proprietary technology for digital acting and conversation and things like that, so we have our own system and tools that work with our conversation system," said Hudson. "Our writers populate a dialogue editor and that becomes fused with the way that you end up seeing many different pieces of Matinee play out in combination when you have a conversation with characters. We used Kismet for scripting a lot of the way that the level responds to the action or prompting our enemies to do certain AI or having movers react and start moving around and things like that."

Although BioWare's programmers communicated with Epic and other game developers through the Unreal Developer Network (*udn.epicgames.com*) on *Mass Effect 2*, they spent most of their time utilizing UDN on the first game, especially as they ramped up on all the details of the technology the first time around.

"I think UDN is a really good service for when you're first learning the engine," said Hudson. "The biggest challenge when using someone else's engine is figuring out how you're supposed to use it and how to best use it. We used it a lot on *Mass Effect* and I know that our guys are always in contact with Epic."

BioWare chose UE3 for the *Mass Effect* trilogy because they wanted to make an immersive third-person perspective shooter game with sci-fi environments.

"We started out already being a game that was going to work with Unreal, but we took further steps with *Mass Effect 2* to really build the content a lot more like you're supposed to with the engine," said Hudson. "With *Mass Effect*, we built a lot of things handmade at an intermediate level and with *Mass Effect 2* we used more of the Epic method where we build lots of pieces and then assemble them in the end. They're just little differences and it comes down to a team really learning a different methodology with the technology."

David De Martini, senior vice president of EA Partners, agreed that BioWare's work with Unreal Engine 3 shows off the technology's extensibility. "Unreal Engine 3 ships with the tools and technologies we trust for making triple-A games," he said. "We have a great working relationship with Epic from both the development and licensee perspective, and we're continually pleased with how they keep their game engine technology competitive, which helps us deliver the excellent quality games that EA customers expect."

EPIC GAMES

For UE3 licensing inquiries email:
licensing@epicgames.com

For Epic job information visit:
www.epicgames.com/epic_jobs.html

WWW.EPICGAMES.COM

GIVE SPREADSHEETS THE BOOT!

DevTest Studio

The game industry's #1 choice for test management and defect tracking.

DevTrack

Use DevTrack to track defects/issues

- Track each issue through a definable workflow
- SCM integration-track fixes against their source code deliverables
- Deploy a resolution across multiple releases, versions and products
- Reporting and metrics to illustrate the entire defect lifecycle

DevTest

Use DevTest to manage your testing

- Create a central repository for your test cases, Knowledge items and automation scripts
- Schedule releases and test cycles using a wizard-driven interface
- Execute test assignments and submit defects from the same interface
- Track results with real-time dashboards and reports

TestLink

Use TestLink to automate your testing

- Add automated tests to the DevTest test library
- Schedule automated tests along with manual tests
- Launch automated tests from the DevTest interface
- Track automation results with real-time dashboards and reports

Try DevTrack and DevTest live. Download free evaluation software. Watch a recorded overview demo.
www.techexcel.com

See Us at GDC in San Francisco

TechExcel | 1-800-439-7782

from the single-player game and implemented it in a competitive multiplayer environment. It immediately felt right. Everything went fairly smoothly as we made choices about what kinds of game types and rule tweaks to make. The one sticking point was online melee combat, which took us about six months to get right. We iterated through every approach we could think of, from synched like the single-player game, to semi-synched, to a button-timing mini-game, before we finally settled on the simple "throw the punch, deal the damage" system we shipped with.

We chose to use the Amazon Elastic Compute Cloud (EC2) for our statistics and for the machinima cinema files that players would be able to upload because of its scalability and the vastly reduced cost to Naughty Dog across the lifetime of the game.

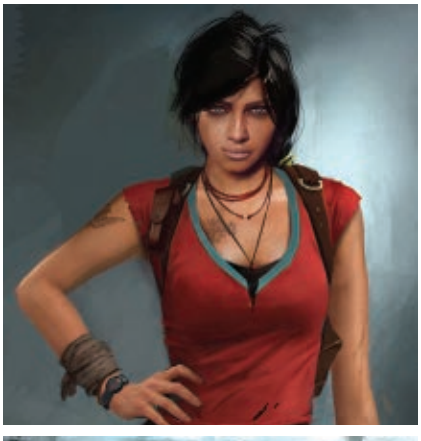
When we announced our multiplayer game in March of 2009, we heard rumblings that fans were concerned that the single-player game would suffer as a result of divided team resources. We're happy to say that tightening up our combat mechanics to make them snappy enough for multiplayer really helped us with the feel of the single-player game.

4) THE BRAVE NEW WORLD OF OUTSOURCING. The outsourcing of art and animation became increasingly important while we were making UNCHARTED 2. We had good experiences outsourcing part of the work on the cutscenes for UNCHARTED: DRAKE'S FORTUNE, and for UNCHARTED 2 we cemented an excellent relationship with the animation department at SCE San Diego Studio, which staffed up to give us the extra capacity we needed to get everything done.

In the late spring of 2009, once it became clear to us just how much effort was required for the 90 minutes of complex pre-rendered scenes that UNCHARTED 2 includes, we brought Technicolor's animation team back into the fold. Between Naughty Dog, Sony San Diego, and Technicolor, our cinematics team totaled 32 animators—more animators than on all of Naughty Dog's previous projects combined.

We also took our first steps into the world of outsourced art assets. We had always been concerned that it would be hard for us to hit the right quality bar using outsourcing, but thanks to the efforts of the outsourcing studios we worked with (XPEC in Taiwan and Ladyluck in the Philippines), and the process we drove at Naughty Dog, we were happy with the results.

We had to be very diligent in staying on top of every aspect of the outsourcing process to ensure that we gave our outsourcing groups everything they needed to succeed. We essentially trained them to make art that we could use. We would provide a package of reference materials that included specifications and detailed construction instructions, screenshots of the locations where the assets would be used to show their context, and prototype level layout geometry to define the volumes the assets should occupy.



UNCHARTED 2

postmortem

We outsourced a great diversity of work, from environments and cutscene stages to characters and accessories, along with monotonous work like UVing and LODing. It's really thanks to our outsourcing teams that our game is as full of eye candy as it is. Outsourcing enabled us to create an amazing amount of art and animation while keeping our team small. Our level artists were able to spend more time iterating level design with our game designers and maintaining the levels while someone else sweated some of the details, and we could scale our team up and down when necessary without having to strain our internal infrastructure through costly expansion and the discomfort of layoffs.

5) PLAYTESTING AND METRICS. In the course of making UNCHARTED 2 we did more formal playtesting than we'd ever done before. We ran fifteen playtests over the last ten months of the project, compared to seven over the whole three years of the first game's development. This resulted in fewer rough edges in gameplay than in any game we've ever shipped (although a couple still snuck through!).

We ran most of our playtests in a rather jury-rigged but functional playtest room in the Naughty Dog office. We had ten TVs, each with a PS3 test station that was hooked up via video capture boxes to a PC, to record events on screen. We didn't record video of the players' body language, though that would have been a good addition. The TVs had 2' by 3' pieces of card bought at a stationary store propped up between the TVs so that the players couldn't, even accidentally, see what their neighbor was doing in the game.

Running our playtests in-house had the enormous benefit of allowing all of our designers and QA leads to regularly see their levels in action with new players. Of course, there are few things better for a game's design than for the designers to watch it being played by people who have never played it before.

We got our playtesters to play through as much of the game as we had finished building, even to basic levels of completion. We didn't let them talk to each other and were merciless about not giving them any help when they were stuck, unless we knew that something was broken.

As they played, we uploaded metrics about their actions to a database over the network—things like how long it took them to complete each part of the game, or how frequently they died between continue points. We put the data into a spreadsheet and looked at the median values for each group. After color-coding cells with values above or below certain targets, parts of the game that were potentially problematic immediately jumped out at us. We then started looking at the gameplay videos to investigate each potential problem.

Doing so much playtesting was particularly important because several complex sequences of gameplay only came together very late in the project, and to ensure that the things we added or changed didn't present unforeseen problems for our players, we conducted "sanity check" playtests right up to the end.

WHAT WENT WRONG

1) NOT QUITE ENOUGH PLANNING. One of the downsides of our philosophy of simply getting on and building the game was that the line between preproduction and full production became blurred. We hadn't really begun to plan ahead until we finished work on the first UNCHARTED, so we scrambled to solidify as many elements of the game's content and story as we could in order to stay ahead of the team as they started building assets that we hoped would find a home in the shipping game.

The story team made a lot of key decisions in a timely enough manner to provide a framework for our forward motion. What emerged from preproduction was our focus on Asia as a location for much of the story, Marco Polo's lost fleet as our real-life historical mystery, the idea of an old friend of Nathan Drake's who would ultimately betray him, and some big chunks of the game's macro design.

However, even in the absence of a story structure to frame them, the first levels took on a life of their own. Their footprints grew and their gameplay firmed up to a good degree. When the game macro was finished in the spring of 2008, the story beats that related to parts of the end of the game were still a little fuzzy. We couldn't quite decide how the threads of the story would twine back together as we neared the end of the game. We eventually worked it out, of course, but we couldn't quite fix all the issues that had arisen.

So even though a lot of people who play UNCHARTED 2 don't notice anything amiss with the end of the game, when we play it through we feel that there aren't quite enough strong story beats in the monastery to match the length and intensity of the gameplay there, and it's the first place in the game where the pace begins to flag. Hopefully we'll learn the lesson of this minor

misadventure and pay special attention to pacing issues for parts of the game whose level design starts early and whose story design finishes late.

2) RAN OUT OF TIME AND SPACE. UNCHARTED 2 was the biggest project Naughty Dog had ever attempted. We had aimed to make a substantially longer single-player game than UNCHARTED: DRAKE'S FORTUNE, which would feature nearly a motion picture's length of pre-rendered cutscenes. The game would include both competitive and cooperative multiplayer modes, tools for the machinima community, and lots of engine additions and improvements. However, by the end of the project, we were increasingly squeezed for time.

Critically, we ran out of time to animate our cutscenes, and everyone involved had to sweat bullets to get them finished. We should have gotten underway with their creation a lot earlier. Also, most of the visual effects for the cinematics were created in the last two weeks of the project by the same team that worked on our in-game visual effects, dependencies having forced them to leave the cutscenes until last.

The implementation of our in-game dialog was pushed until way late because of similar dependencies, and we made a lot of dangerous content and scripting additions to the game when we should have only been fixing bugs. The added dialog also had an impact on the difficulty of the game as the banter between the characters made the player's goals very clear, where they might previously have required more deduction on the part of the player.

We decided to add co-op partway through full production, which was quite a large feature to add mid-project. Looking back, deciding to reach for co-op might have been the point at which we began to seriously overextend ourselves, but our co-op designer and team did an amazing job and we're very happy with what we shipped.

In a scenario that's familiar to game developers but which is becoming increasingly financially and artistically untenable, we had no time for any kind of postproduction. Our particle artists, lighting artists, and sound designers had to scramble for every second of polish time they could grab, since the rest the team—designers and artists in particular—were making changes and bug fixes that affected their work right up until the end. In the future we're determined to build proper postproduction time into our schedules so our games can have the level of aesthetic polish our audience expects.

Finally, we even ran out of space on the single-layer Blu-ray disc we were using. We hadn't planned ahead to use dual layer Blu-ray storage, so we had to remove some bonus content from the disc and compress some assets more than we would have liked.

3) BOSS DIFFICULTIES. Our bosses were difficult, both in the sense that a few of them provided too sudden an increase in challenge for players, and in terms of the difficulty to conceive and implement them.

We hadn't felt obliged to have a lot of traditional bosses in UNCHARTED 2. Boss monsters in the ZELDA vein that invite you to experiment with recently-acquired mechanics just don't work well in a game like ours. Those usually need to be able to soak up a lot of damage and have some gadgety attack or defense quirk, and that isn't a good fit for us in narrative terms, since UNCHARTED is set in a world that's mainly realistic. Any disruption of the consistent "grounding" can be really jarring to the player's suspension of disbelief.

Also, we don't hand out new play mechanics on a regular schedule like some other games, so we have to work a lot harder to help the player find novel ways to use play mechanics that have become familiar. Instead of traditional bosses, we mainly used elaborate set pieces to provide the same kind of climactic play and narrative experiences that create and punctuate the rhythmic flow of UNCHARTED 2. The confrontations with the helicopters and the tank are good examples of this.

However, we did want Drake to have a couple of fights against humans and we decided to tackle that challenge head on. Without giving away too much for readers who haven't finished the game, we got a narrative pass on one of these humans being a bullet-sponge thanks to events in the story. The other human proved trickier though, and as we did on UNCHARTED: DRAKE'S FORTUNE, we ran out of time to create special gameplay (and polish) for both humanoid bosses.

Neither boss turned out badly, but both are in danger of providing a difficulty spike and frustration for some players. By the time you read this we will already have implemented some boss ideas for our next project and we're determined not to make the same mistakes again.

4) SOME SMALL STUFF WE COULDN'T SWEAT. We were relieved when reviewers started praising the amount of detail and polish that UNCHARTED 2 has, because there were a few things that we felt we hadn't polished well enough.

For example, we didn't put enough early focus on collaboration between design and art to establish a crystal-clear language for edges that Drake could grab onto and climb. When we did try, it was hard to reach a consensus, with each side pulling hard toward either function or aesthetics. Our game ended up with too many low "grabbable"-looking ledges that you couldn't grab, making things confusing and frustrating for the kind of player whose play style tended toward "perimeter scans" of any level in which they felt stuck, as they jumped up against every wall looking in vain for something to climb on.

CONTINUED ON PAGE 29

Penn
University of Pennsylvania

Penn Engineering's

MASTERS IN COMPUTER GRAPHICS AND GAME TECHNOLOGY

offers recent Engineering and Computer Science grads:

A unique, one-year graduate-level program combining **art** with **engineering** and balancing **theory** with **practice**

A **wide array of interdisciplinary courses** from UPenn's School of Engineering and Applied Science, School of Design, the Wharton School of Business and the Annenberg School of Communication

Access to the **LiveActor Motion Capture Studio** for student game projects

Cutting edge research opportunities at the **Center for Human Modeling and Simulation**

A world class education at University of Pennsylvania's Philadelphia-based **Ivy League campus**

GET IN THE GAME

www.cis.upenn.edu/grad/cggt/

UNCHARTED 2

postmortem

CONTINUED FROM PAGE 27

Happily, because of the amount of playtesting that we did, we didn't end up with much of the opposite case—edges that you could grab that didn't clearly look grabbable. Every time a playtester couldn't see what to grab next because of texturing, modeling, camera position or lighting issues, we were able to spot it and fix it. We know from experience that these kinds of issues are the low-hanging fruit of what we call "progression brick walls," and that they're easy to catch with diligence and perseverance.

Also, while our game has a lot of "incidental breakables"—small objects that fall over when struck or shatter when shot—we didn't end up with an even distribution and density of them throughout the game. If you're eagle-eyed and looking out for it, you'll notice that a bottle that you can knock down in one area might be immovable in another. We simply ran out of time: we left the implementation of the incidental breakables until the very end, and as we all scrambled to get the game finished we didn't have the time to do a full pass of the game, removing the objects from the static environment and implementing them as breakable objects while keeping an eye on performance. We're planning to create tools that will help us speed up this relatively simple task so we can avoid running into the same problem next time.

5) CRUNCHTASTIC TIMES. As I mentioned earlier, UNCHARTED 2 was our most ambitious project to date, and by the spring of 2009 we realized just how much game we had bitten off, and that we were going to have to chew extra hard, make some cuts, or choke.

So we reduced the scope of several levels early enough that we hadn't invested too much in the way of art resources in the affected areas of the game. We also lived with our prototype levels and lists of gameplay ideas long enough that we could see fairly clearly what we should keep and what should be cut.

However, we didn't cut to the point where we would have been able to coast to the finish line, and life throughout 2009 was tough for almost everyone at our studio. We have never mandated crunch at Naughty Dog, but we have hired people with personality types that make them hard-working, willing to accept responsibility, and perfectionists and that led to many months of long hours, late nights, and truncated or skipped weekends.

The demo we made for E3 2009 marked the true beginning of the long hours, although many people had been working extra-hard for much longer than that, leading to a summer of stress on people's family lives and personal health and the problem of reduced productivity of tired people.

While we don't think we'll ever be a studio that works nine-to-five year-round, we take the threat that crunch presents to the integrity of our studio and the wellbeing of the Naughty Dogs very seriously, and we're discussing ways we can avoid ever having to repeat the experience of UNCHARTED 2 in terms of the toll that the project's crunch took. We know we have to become more disciplined about setting and hitting internal deadlines to get traction on our projects earlier, and we're going to try other approaches like putting mandatory limits on the amount of time people can spend at the office.

WE STRUCK GOLD

» UNCHARTED 2: AMONG THIEVES has been a big success for us. We got it finished on time and on budget without any major disasters along the way, and we're all very happy with the finished game. Our initial sales have shown players are happy with it, too.

Looking back, one of the keys to the success of the project was that we continued to keep an open mind about process, adopting only those ideas that were really working and jettisoning those that didn't. Knowing when to pull the plug on plans that weren't coming to fruition, but being tenacious about everything we approached, was another. Indeed, those are probably good ways to summarize what results from our "garage developer"-flavored studio culture of open communication and do-ocratic organization.

In the end, a lot of good old-fashioned hard work was needed to get everything done on time and to quality. The amount of passion, effort, tenacity, and talent that went into the design and production of UNCHARTED 2: AMONG THIEVES is its own tribute to everyone involved in the project, and a clear sign that we currently have the best team we've ever assembled under the Naughty Dog roof—and beyond.

We're very happy with the overwhelmingly positive critical reception we've gathered from both the press and the public—which is definitely the warmest welcome any game by Naughty Dog has received—and we're very excited about carrying the lessons we learned forward onto our next project. We hope that they're useful to you in your work, too, and we look forward to a bright future for the games we're all going to make.

RICHARD LEMARCHAND *is a game designer at Naughty Dog, and was the co-lead designer of UNCHARTED 2: AMONG THIEVES. Richard has worked on twelve award-winning and critically acclaimed console games in his 18-year career.*

GAME DATA

PUBLISHER Sony Computer Entertainment

DEVELOPER Naughty Dog

NUMBER OF DEVELOPERS Approximately 90 full time plus around 15 contractors

LENGTH OF DEVELOPMENT 22 months

RELEASE DATE October 13, 2009 (North America)

LINES OF CODE Excluding external libraries and tools: 2,539 source files with a total of 1,110,817 lines of code, 3,703 script files (including render settings and particles), totaling 708,951 lines of code

SOFTWARE Autodesk Maya, Pixologic ZBrush, Autodesk Mudbox, Adobe Photoshop, Adobe Illustrator, Apple Final Cut Pro, headus UVLayout, E-on Software Vue, CrazyBump, Perforce, Cygwin, SCEE Nova, Microsoft Visual Studio, Emacs, Notepad ++, SlickEdit, Perforce, Git, ProDG

PLATFORM PlayStation 3

Scaleform®

#1 Video Game User Interface Solution

Smokin' Fast UI

Create highly optimized Flash® UI with Scaleform GFx 3.1

Use the Scaleform AMP profiler to tune runtime memory and performance

Improve HUD performance by ~10x with direct access to Flash objects

Download the new SDK and HUD Starter Kit to learn more

www.scaleform.com

Scaleform GFx

BREAKING THE ICE

CREATING CONSCRIPT—A COMPLEX CONVERSATION SYSTEM

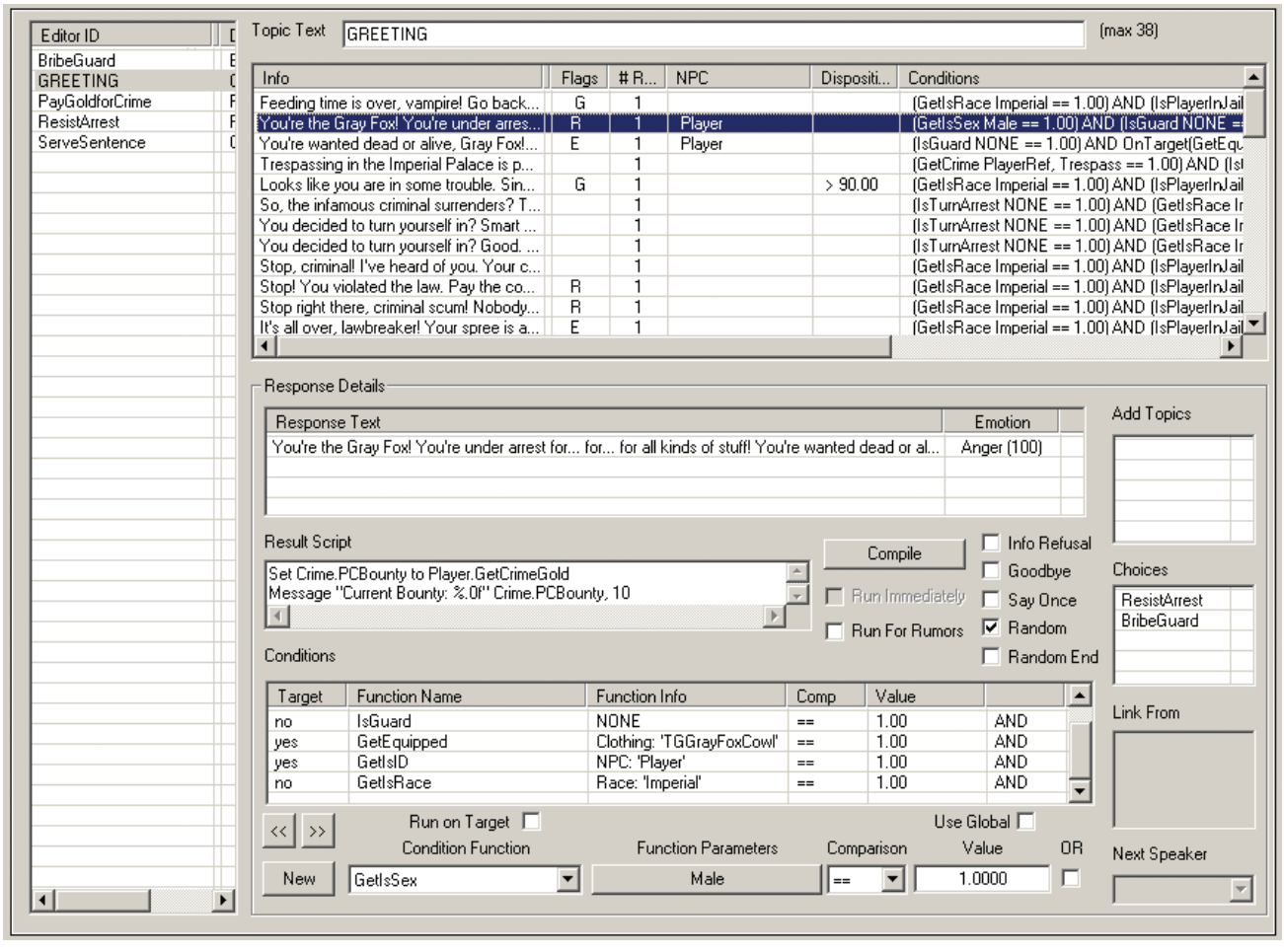
BRENT FRIEDMAN

Whether your game title is heavily steeped in story or requires only the occasional battle cry, it likely needs some sort of conversation system. The requirements of such systems can be very rigorous. Most titles will require the conversation to interact with game logic, cinematics, animation, and sound. In this article, I will review several modern tools used for conversation development and introduce a conversation scripting language called "Conscript."

BRIEF SURVEY OF CONVERSATION DEVELOPMENT TOOLS

» The conversation toolset used in recent Bethesda titles like FALLOUT 3 is a topic-based database editor. Each table in the database is a topic, and each row is a conditional response. Whenever a topic is selected, the character searches the table from the top down for the first response with a condition that holds true. For instance, if guards should always have a friendly greeting for fellow guards, the first row in the Greeting table would need to have that response and condition. If you later decide that a special greeting is needed for the Captain of the Guard, that response should go before the generic guard response. Getting the order of these entries correct can be challenging for a large table, as it is essentially a large `if-else if` block.

BioWare's conversation tools operate quite differently. Conversations are first-class entities that can be selected based on context. Every line of dialog has a list of branches to which the



conversation may progress, as well as a condition that enables or disables the response. It is fairly easy to see the logical flow of a conversation given such a diagram. This tool also has very robust preview tools that can be run directly from the editor, which is invaluable for testing purposes.

One issue to consider with both systems is that they require writers to use a scripting language to interact with game logic. In this sense, a writer must learn both the GUI tool and the scripting language to fully utilize the conversation system.

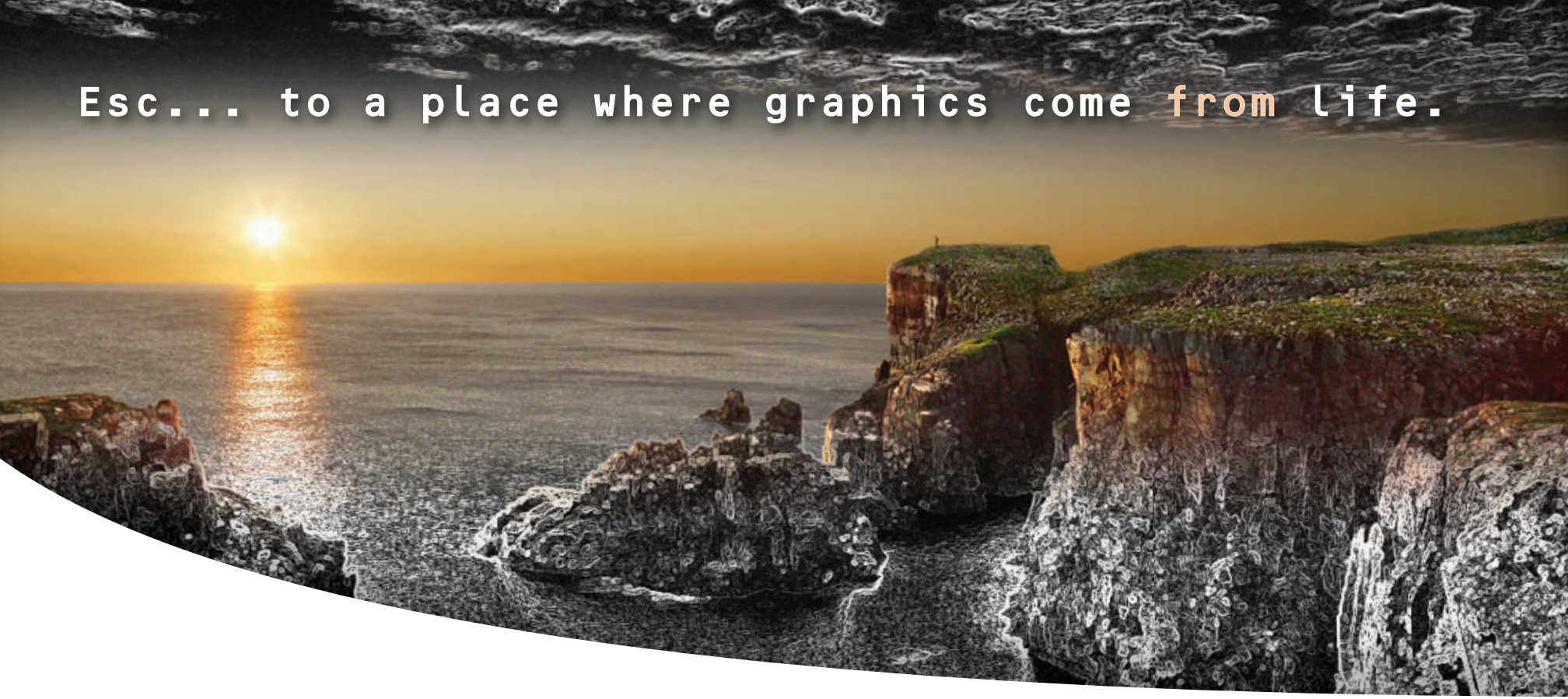
FIGURE 1 **Bethesda's conversation toolset uses the topic list approach. These tools do not represent conversations as discrete entities. Instead, each character has a list of available topics and responses to those topics change based on context. Conversations are simply threads of execution through these tables. A separate dialogue tree view was added for FALLOUT 3 to aid in understanding conversation flow.**

INTERACTIVE FICTION

» Interactive fiction and text adventure games typically have specialized programming languages. These languages provide an interesting comparison with the previous tools. They are languages for general-purpose game logic, not just conversations. Conversations in these games often use commands like ASK or TELL to maintain the illusion of freeform gameplay and to provide a consistent user interface. Some have also experimented with menu-driven UI like those found in most commercial games. The languages most prominent in the interactive fiction community are Inform7 and TADS3. The methods described here cover only the suggested method of conversation development as per their documentation.

Inform7 uses a rule-based approach to conversations: dialog occurs in response to some rule becoming fulfilled. Typically this rule will consist of a user command to respond to and a series of preconditions

FIGURE 2 **The tools used in the NEVERWINTER NIGHTS games and DRAGON AGE: ORIGINS represent each conversation as a dialogue tree. Most of this dialogue is conceptually linear even though the tool displays it as nested. This is a consequence of the "every response has a list of branches" approach.**



Newfoundland and Labrador.

A place where you can restart, refresh and, of course, recharge. It's where inspiration was born and a **creative and unique culture** reside. And it's not far away – just off the east coast of Canada, bridging the gap between North America and Europe. Game developers are already discovering that creativity lives here, and that our workforce is highly productive and loyal. Every year our education facilities release top talent from highly recognized programs such as **Digital Animation, Computer Science, Software Engineering and IT.** Run the numbers and you'll find that Newfoundland and Labrador is a leader for economic growth in Canada. And, that operating a business here can help you leave the competition behind. **We're ready to start whenever you are.**

Department of Business Government of Newfoundland and Labrador, Canada
877.727.6353 / business@gov.nl.ca / www.nlbusiness.ca

Newfoundland Labrador
CANADA

it's happening here.

June 15-17, 2010 * Los Angeles Convention Center * E3Expo.com

E3 Expo 2010

E3 Expo is the world's most important annual gathering for interactive entertainment.
It's where business gets done, connections are made and the future of the industry is revealed.

International Media Sponsor
MCV

Official Show Daily Sponsor
GAMEPRO MEDIA

REGISTER NOW AT E3EXPO.COM

Produced By
IDG WORLD EXPO

E3 Expo is a trade event and only qualified industry professionals may attend.
No one under 17 will be admitted, including infants. Visit www.E3Expo.com for registration guidelines.

specifying a conversation state. Branching dialog requires a new rule to be created for each branch. For instance, you might create a rule: Instead of talking to John when topic is **Politics**: say "Politics? I won't touch the stuff!" (see the Inform7 tutorial in Resources). Intricate conversations require intricate rules and potentially many "when" conditions to store the conversation state. Branching dialog can be unclear when written in rule form because nesting is implicit within the rule declaration rather than explicit and obvious.

TADS3 uses a very similar system for categorizing responses by command and conversation state. For instance, **+AskTopic @Politics** "Politics? I won't touch the stuff!" would create a generic response for **ASK ABOUT POLITICS** (see the TADS3 tutorial in Resources). If this **AskTopic** were grouped under a particular state it would only be applicable when the conversation was in that state. Complex branching, it seems, should be implemented as a hierarchy of substates. This additional state hierarchy sets it apart from Inform7's approach by making nested dialog explicit.

ORIGINS OF CONSCRIPT

» Conscript is a scripting language for designing menu-driven interactions and conversations. As lead programmer and lead designer on a cultural simulation title, I was tasked with developing a conversation system that would be accessible to non-programmers and that would be able to cope with rapidly changing requirements. I designed and implemented Conscript over the course of a few months. Once it was rolled out to our team, we recruited English majors with little or no programming experience to lead development in Conscript. Within a month we had several hundred lines of dialog written, including complex branching based upon character emotion and familiarity with the player.

Our project has been considered a spectacular success by our client in part because Conscript has a huge potential for end-user modding. The language continues to grow and certainly has a lot of room for improvement. Our team is currently experimenting with Conscript as a UI scripting language.

My first goal with Conscript was to make it clear and concise. As much as possible, I wanted Conscript to look like a nonlinear movie script. The language should also be very easy to learn and allow the writer to think of conversations as a logical flow. My next major priority, ease of implementation, was driven largely by time constraints and a personal lack of familiarity with compiler theory. Compared to most languages, Conscript is extremely simple to implement a compiler and virtual machine for. The Lisp-like square brackets property syntax, for instance, is very easy to parse and eliminates the need for considerations like order of operations. I feel Conscript was well designed to meet these goals.

Because each game title has its own unique data structures, requirements, and characters, Conscript is not intended to be a complete solution. Instead, Conscript provides a standard for structuring flow of execution and accessing your game's unique functionality. For instance, throughout the following examples, I will refer to two variables; **pc** and **npc**. These are variables that would refer to the player and non-player character during an interaction. Your title may require npc-to-npc conversations or more than two participants. Conscript can handle these situations equally well. What initial variables exist and what they are capable of is implementation-defined. Scripts can be written agnostically of the actual participants, so a single conversation can be used for an entire class of characters. It would also be possible to write conversations for an unknown number of participants (a randomly-generated mob, for example) by selecting speakers from a pool.

Conscript uses the topic list approach to conversation. **Topics** can be "**shown**" or "**hidden**," adding or removing them from the user interface. **Hidden topics** can be used like functions, providing reusable sections of code. One **topic**, called the **auto topic**, is marked as the entry point of a conversation.

Whitespace rather than braces or other begin/end markers is used for grouping. This "outline format" is intended to be more intuitive for those with no programming experience. This also forces each statement to be on its own line, which improves readability.

listing 1 Conversation *Casablanca*

```
auto hidden topic "Dawn"
 condition pc["agreed_to_leave"]
  =1
   npc: "No. Get on the plane where you belong."
   exit
 npc:"You´re getting on that plane with him, where you belong."
 option pc
  ="Okay, okay. I´m going."
   set pc["agreed_to_leave"] 1
   exit
  ="But " npc "! No! I--"
   npc:"If you stay we´ll both wind up in a concentration camp."

topic "You´re saying this only to make me go."
 hide[this] #remove this topic from UI
 set chk1 1 #set global variable chk1 to 1
 npc: "If that plane leaves and you´re not on it, you´ll regret it."
 npc: "Maybe not today, maybe not tomorrow, but soon."
 goto "FinaleCheck"

topic "But what about us?"
 hide[this]
 set chk2 1
 npc: "We´ll always have Paris."
 goto "FinaleCheck"

topic "I said I would never leave you."
 hide[this]
 set chk3 1
 npc: pc ", our problems don´t amount to a hill of beans."
 goto "FinaleCheck"

#do the finale if all three checkpoints passed
hidden topic "FinaleCheck"
 condition chk1["&"][chk2]["&"][chk3]
  =1
   narrate[pc " starts to cry."]
   narrate[npc " lifts up " pc "´s chin"]
   npc: "Here´s looking at you, kid."
   set pc["agreed_to_leave"] 1
   exit
```

Watch out for the NEW xaitControl 3.0 SDK.

xaitControl 3.0 offers a cross-platform live-live debugger featuring conditional breakpoints, per instance realtime-view, history and graphical decision path, manual triggering and blocking of transitions and dynamic FSM updates.

Contact xaitment today for more information about the BrainPack Program under brainpack@xaitment.com or visit our website www.xaitment.com

xaiTMENT
INTELLIGENCE ENGINEERED

listing 2 Selector Conversation

```
auto topic "Begin"
 condition pc["current_quest"]
  ="MQ_01"
   goto conversation "Quest1"
  ="MQ_02"
   goto conversation "Quest2"
  ="MQ_03"
   goto conversation "Quest3"
 exit
```

LISTING 2 **A practical example of sub-conversations.**

listing 3 Conversation MurderBase

```
auto hidden topic "Greet"
 pc: "Hello, " npc ". My name is" pc "."

topic "Recent murders"
 condition npc["questioned"]
  =1
   npc: "I've told you everything I know."
  =else
   random
    group
     npc["playanim"]["sigh"]
     npc: "I hate living in this part of town."
    npc: "I heard about that on the radio."
    npc: "I wouldn't know anything useful."
 set npc["questioned"] 1
 exit
```

LISTING 3 **Conversation MurderBase, the base conversation for all characters in our hypothetical murder mystery.**

THE PROPERTY-ORIENTED PARADIGM

» The central philosophy behind Conscript is the idea of properties. Every object has a list of properties. For instance, a character's ability to talk can be considered a property of that character, and the ability to say "Hello" can be considered a property of his ability to talk. Formally, every object is a mapping from a string to another object. Notice that there is no distinction between functions and data.

Conscript has no concept of scope. Because all objects are "owned" sub-properties of other objects, scoping is handled purely by the construction and destruction of objects by your game framework. This allows your game to handle issues like quest journaling in its own manner while still giving designers the freedom to influence such systems.

See Listing 1, which puts a conversation from the famous film *Casablanca* into Conscript, for an example of how it operates. Global variables (such as `pc`, `npc`, and the `chk` variables used in the *Casablanca* listing) are implemented as properties of the conversation context. As such, they are intended to exist only until a conversation has ended. Data can be saved between conversations by creating a property on something that will still exist after the conversation has ended. This idea is demonstrated in the *Casablanca* example by setting an "`agreed_to_leave`" property on the `pc`. This property would remain part of the character for the duration of her lifetime. Your game framework may create its own

listing 4 Conversation Witness

```
inherits MurderBase
topic "Recent murders"
 npc: "I don't feel comfortable talking about this."
 option pc
  ="Why is that?"
   npc: "I was a neighbor of old man Finkelman."
   show[ "The Finkelman Case" ]
   hide[ this ]
  ="I'm sorry."
   npc: "I'll be leaving now."
   exit
hidden topic "The Finkelman Case"
 npc: "I was knitting when I heard the scream..."
 narrate["Time passes..."]
```

LISTING 4 **Notice that it never creates or accesses the "questioned" property. Witnesses will not have this property.**

mechanism for deciding on the lifetime of properties, but in general they are intended to remain until the owner is deleted.

Because talking is the most common operation in a conversation, the `:` operator is shorthand for property "`say`." For instance, `rick: "Hello"` is equivalent to Rick ["`say`"]["`Hello`"]. For our game, "`say`" blocked the virtual machine from continuing until the character had finished speaking. This meant only one person could speak at a time.

CONTROL STATEMENTS

» Many flow control statements exist in Conscript. Here are a selection of them:

`Goto` ***acts like a function call.*** It executes some code and returns when that code has been executed. `Goto` can be used to execute a `topic` (`goto x`) or to execute a sub-conversation (`goto conversation x`). The compliment of `goto` is `exit`, which acts like a `return` statement. `Exit this` means to exit the current `topic`, while simply `exit` means to quit the conversation entirely. For sub-conversations, `exit all` is used to exit the entire conversation system rather than just the current sub-conversation (see Listing 2).

Conditonal commands allow one of many dialog branches to be taken.
`Random` randomly selects and executes one branch. `Condition` compares a variable to several conditional branches and executes the first branch whose comparison is true. `Condition` is analogous to an `if-else if` statement. `Option` prompts a character to select a branch from a predetermined list. Like much of Conscript, how your game presents these options in the user interface is determined by your particular implementation.

The *Casablanca* conversation demonstrates most of the functionality in Conscript. Conversation starts at the `auto topic`, first checking the "`agreed_to_leave`" flag. Players who pass this check will be presented with two options. Those who select the second option will choose between three selectable `topics`. The `FinaleCheck topic` handles the finale, only running the `=1` branch when those three `topics` have run. The `hide` command (for making a `topic` non-selectable) and single-line comments (`#`) are also demonstrated. `FinaleCheck` also demonstrates how string concatenation is implicit. Variables and strings are always appended to each other when arbitrarily mixed.

Originally in the design of Conscript there was a "selection" mechanism for determining whether the conversation was valid in the current context. When starting a conversation, the character would look at the selection block for every conversation attached to him and use

Fallout 3's dialog system has deep consequences for gameplay.

the first conversation with a selection block that returned true. The selection block never saw implementation. In practice, every game will have a different way of determining which conversation to use at a particular time. A more robust version of this functionality can actually be implemented with a selector conversation like the one demonstrated in Listing 2. The interaction will be deferred into a sub-conversation based upon various conditions and then exit once that sub-conversation has completed .

Many games have pseudo conversation dialog such as battle cries, persuasion responses, or barter activity. You may decide to reserve topic names for these purposes and specially call the appropriate topic when warranted.

INHERITANCE

Conversations can "`inherit`" and "`override`" `topics` from other conversations, much like in class-based inheritance. A simple example for a murder mystery narrative is shown in Listings 3 and 4. A conversation using the Witness file would begin at MurderBase's Greet `topic` (Listing 3) because the Witness conversation has no `auto topic`. The overridden Witness version of `topic` "Recent murders" would be used when that `topic` is selected (Listing 4).

Inheritance was a major productivity enhancer for our team. Our game involved a wide variety of randomly-generated characters and only a few characters with specific, useful knowledge. By overriding certain `topics` for special characters we were able to quickly prototype the basics of conversational gameplay. You can see how inheritance encourages rapid prototyping and "fill in the details later" development.

POTENTIAL IMPROVEMENTS

Conscript is a slow language because of its dynamic nature. This was an intentional design decision. Most single-player games pause the game world when a player enters a conversation so speed is not a concern.

Conscript should not be a performance bottleneck even for conversations that do not pause the game world (like overheard NPC chatter or multiplayer environments). Conversations only need to execute at a speed similar to that of human conversations, so your virtual machine may only need to be updated a few times per second. The performance of Conscript should barely factor into the speed of most games. Memory constraints may be an issue depending on implementation, but this should be manageable because only the text of currently executing conversations needs to be loaded. Even more aggressive measures like dynamically loading and unloading individual topics could also improve the memory footprint.

Perhaps the most significant problem with a simple implementation of Conscript is poor integration with game assets. Writers must refer to any in-game assets by name because Conscript (like most programming languages) is just text. The concern here is that referring to assets by name is error-prone. A compiler that searches for referenced assets at compile time would be able to find these errors before they made it into the game and a sophisticated asset-aware IDE would help writers to not make the mistake in the first place. In one of our games, `say` expected a dialog identifier rather than the verbatim text of the speech. This identifier referred to an instance of an implementation-specific data structure with localized subtitles, audio, and facial animation. Another one of our games automatically selected audio and animation from a pool based on character emotion, so `say` only needed the subtitle. Clearly, asset integration depends entirely on the conversation mechanics. Exactly how much of an issue this is will depend on the sophistication of the implementation and the nature of conversations in the game. I would strongly encourage studios making heavy use of assets through Conscript to add appropriate checking to their compiler and to create a specialized conversation development tool with asset support. Games with less asset integration (text-to-speech, for instance) may not need to take these extra steps.

Some aspects of Conscript, especially the need for quoting string literals, can be error-prone. The easy solution we developed was to assume quotes were intended if no variables are found to exist. For instance, if accessing the nonexistent variable `doesnotexist`, the value of the variable would be "`doesnotexist`." This masked most of the logical errors related to quoting but is a nasty hack that could cause even more confusion. Another solution is to implement a shorthand for accessing a string literal property. In Javascript, for instance, `npc.grin` is equivalent to `npc`["`grin`"]. Because these errors can usually only be detected at runtime, a sophisticated debugger or error reporting mechanism should be developed. The other major difficulty experienced by our writers was getting the proper level of indention. Deeply-nested `topics` can be difficult to indent properly. A well-designed IDE could reduce such headaches by making indention levels clearer, but ultimately this is a problem caused by nested complexity, a direct result of the writer's style. Hidden `topics` should be used to break up complex dialog into more manageable parts.

Most of the Conscript specification has been fully laid out in this article; it is a very simple language. A complete language specification is (as of this writing) still being developed. Feedback and interest would be invaluable. I hope to formalize Conscript as a free industry standard tool.

BRENT FRIEDMAN *is a C++ programmer at the University of Texas at Dallas. He is passionate about improving and simplifying our processes so we can make better games.*

resources

CONSCRIPT
A Notepad++ plugin and demo videos using Conscript are available at www.misterbrent.com.

INFORM7 CONVERSATION TUTORIAL
http://inform7.com/learn/man/Rex74.html#e74

TADS3 CONVERSATION TUTORIAL
www.tads.org/howto/t3conv.htm

Another narrative tool that may be worth considering:
PROGRAMMABLE NARRATIVE FLOW GRAPH (PNFG)
http://gram.cs.mcgill.ca/theses/martineau-06-pnfg.pdf

INSIDE . . . *R.U.S.E.**

Hands-On Means Exactly That for Eugen Systems' New RTS.

Pioneering souls who operate at the frontiers of technology aren't always the ones that get remembered for their efforts. It's a safe bet, for example, that you know the name of the company that first mass produced the vacuum cleaner and made it a household essential, but do you actually know who invented it?[1] There's no point being first if you're not also the best.

Take Eugen System's World War II-based strategy *R.U.S.E.**. It will get a well deserved place in the annals of gaming history for its headline innovation: It's the first Triple-A PC game designed from the ground up to work with the multi-touch features of Microsoft Windows* 7. It won't, however, become a landmark game simply because you don't need a mouse and scroll wheel to play through it. *R.U.S.E.* can be successfully completed entirely using the pinch/zoom and other gestures familiar to high-end Smartphone owners, but that's not all it has to offer hardened RTS fans and newcomers to the genre alike.

That being said, the game's release in 2010 couldn't come at a better time. All of the major PC manufacturers have recently launched multi-touch monitors, all-in-ones, and laptops to capitalize on public enthusiasm for the new features. While a multi-touch interface won't do

[1] Melville Bissell of Grand Rapids, MI

much for first-person shooter fans, picking up units in a real-time strategy game (RTS) to direct them around the screen with your fingertips seems like a natural evolution for a genre well suited to a form of interaction that is more direct than what the keyboard and mouse can provide. By staking its claim early, *R.U.S.E.* won't just be using the emerging control method, it will in some ways be defining it.

"The game is structured around a 'war-room'-like table which makes you feel like you're a real general pushing units around a map," said Mathieu Girard, senior producer for *R.U.S.E* 's publisher Ubisoft. " For our announcement trailer we wanted to have two guys playing together on a 'real' table, so we worked with a French company, IntuiLab, which produced an amazing IntuiFace* demo using a surface computing tabletop.

"We did some focus group work and a multi-touch monitor was high on everyone's must-buy list. Realizing that consumer computers with Windows 7 would be coming onto the market around the time of our release, and that they would have the same multi-touch capabilities, we knew we had to keep it."

Incorporating an entirely new type of controller mid-way through a project might sound like an enormous task, but two factors helped Eugen implement a successful multi-touch system. The first is that, for better or worse, the team wasn't held back by traditional design documents. The second, more important, reason is that they'd already committed to radically overhauling the familiar RTS interface and had the engine technology in place to perfectly complement the multi-touch screens.

Where the Old Meets New

Located inside a Knights Templar fortress, which dates from the Thirteenth Century, Eugen Systems is just a few blocks from the world famous Pompidou Centre in Paris. The majority of its 60 employees are engine coders, who report to Technical Director and founder Cedric Le Dressay. Their job is to translate the creative team's vision into working builds as quickly as possible so that ideas can be tried out, then pursued or abandoned as appropriate. Thanks to the continual dialogue between designers and programmers, new ideas like multi-touch can be demonstrated and refined or discarded relatively quickly.

"We don't believe in the 'design bible,'" Le Dressay explained. "We work better when everyone talks and can see a live version straight away. Ideas come from the creative team, we implement on specification, and these get tested by the creative team. By doing that kind of iterative process we believe we have a better chance of creating a great game that will appeal to our fans."

"With R.U.S.E., we really wanted to capture the feeling of being a real strategic commander. We wanted a huge battlefield . . . we wanted to create maps that are 100 times bigger, that gave you the feeling you're in complete control, like a god of war."

—CEDRIC LE DRESSAY,
TECHNICAL DIRECTOR AND FOUNDER, EUGEN SYSTEMS

While the iterative process is often frowned upon by many in the industry, largely because of the effect it has on hitting deadlines, it is practiced by some of the best known names in

strategy development, such as Firaxis' Sid Meier. Getting an early prototype up and running is absolutely vital to this game's success. Initial versions of *R.U.S.E.* were based on the company's existing game engine, which was used for its previous game, *Act of War**. Right from the start, Le Dressay knew that he wanted to make an engine capable of running a game that's very different from the standard RTS.

"With *R.U.S.E.*," he said, "we really wanted to capture the feeling of being a real strategic commander. We wanted a huge battlefield. Compared to our previous game, *Act of War*, we wanted to create maps that are 100 times bigger, that gave you the feeling you're in complete control, like a god of war."

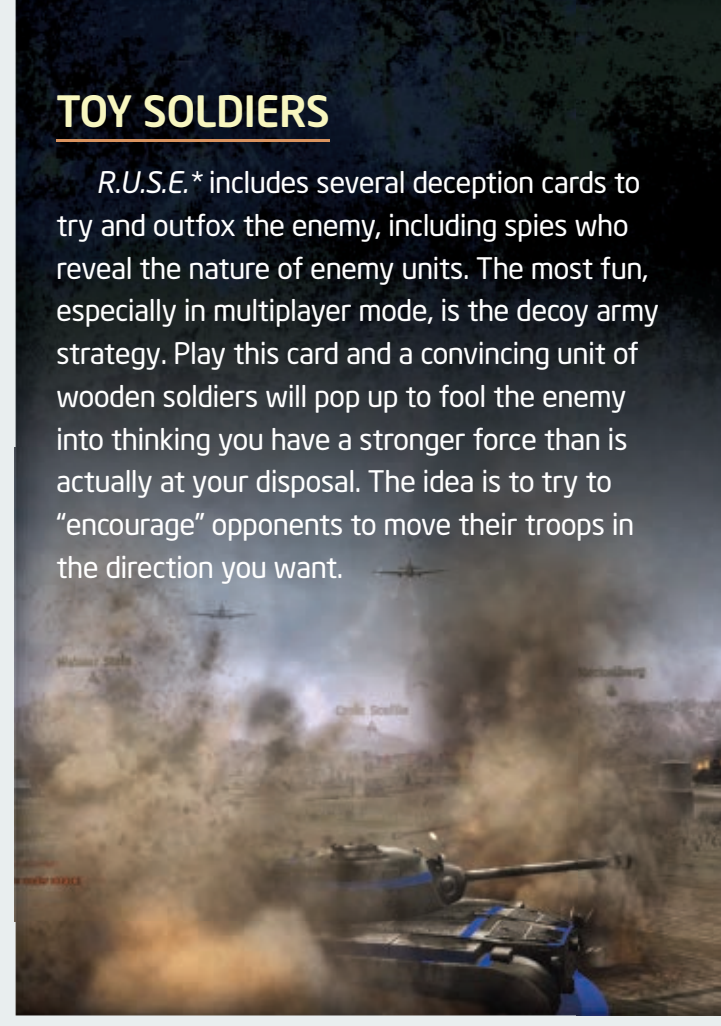
That vision became the IrisZoom*, the code base at the core of *R.U.S.E.* that is designed to move seamlessly between macro and micro tactical control. At its most extreme scale, the game is represented as a giant tabletop in a military HQ. Generals stand impassively by the walls as stacked units with abstract markers are dragged around during the issuance of new orders. When played from this perspective, it might as well come with a long stick and a wonky Bakelite headset in the box. IrisZoom, though, makes it possible to take the player down from that atmospheric overview to an almost first-person camera on the front line in a single, swooping zoom. Importantly, at this level the control system remains unchanged. Every command and gesture is available at any point on or above the battlefield.

The result is that *R.U.S.E.* not only looks different from other strategy games, it has an internal continuity that makes it play differently too.

"The technology we have is very important," Girard explained. "It's how we can express everything without breaking the immersive factor. We are able to scale up visual effects as you zoom in and zoom out so you can always read them, and that's not a simple thing to do. You have to continually re-spawn particles, for example, and keep them consistent as you simplify or make them more complex depending on whether you're moving up or down to ground level."

TOY SOLDIERS

*R.U.S.E.** includes several deception cards to try and outfox the enemy, including spies who reveal the nature of enemy units. The most fun, especially in multiplayer mode, is the decoy army strategy. Play this card and a convincing unit of wooden soldiers will pop up to fool the enemy into thinking you have a stronger force than is actually at your disposal. The idea is to try to "encourage" opponents to move their troops in the direction you want.



Immersion, not multi-touch, is the big idea behind *R.U.S.E.*. That's what influences every other part of the game's dynamic.

Building the Fourth Wall

In *R.U.S.E.*, every effort is taken to keep incongruous elements from breaking your involvement in the world. There's no mini-map, for instance—why would there be when you can quickly zoom out to space for a tactical overview? There's no overview panel with unit information sitting at the bottom of the screen either. If you want to issue an order, just point a cursor or finger at the spot where you want to interact and the right context menu pops up. In other words, it was an interface conceived even before the arrival of multi-touch controls.

"Multi-threading coding is still difficult. It's easy to make mistakes. But it's the golden path to performance, and if you want to get the most out of your game you have to think in terms of multicore processing. Like most developers, we divide the workload between AI and 3D, but it's a bit more complex than that. Both parts of the engine have a lot of asynchronous tasks, like gathering information about which elements need rendering, or making pass-finding requests that analyze strategies."

— CEDRIC LE DRESSAY, TECHNICAL DIRECTOR AND FOUNDER, EUGEN SYSTEMS

Perhaps the most significant innovation is the absence of the health bars that float above a soldier's head in traditional interfaces.

"There's no static panels as in other games," Girard said. "We want you to focus on what is on the battlefield. There are no life gauges on the units. If there's smoke or fire trailing from them, you know they're in trouble. You don't have to keep going back to bases to order constructions either. If you're fighting a battle and need to build some more, you just click to produce them and they'll turn up depending on your logistics.

"We're removing all the tedious tasks that hold RTS games back, but at the same time keeping all the elements that players enjoy—like building bases and units, building an economy, and gathering resources. We want you to feel like you own your army, because you've earned it."

This approach takes a cue directly from classics such as the *Black & White** series, but it's made possible on this epic scale thanks to the multi-threading capabilities of the IrisZoom engine It's this that makes the game fluid and fast regardless of the PC you're playing on, and allows important contextual information to be displayed as and when necessary.

"Without multicore," explained Le Dressay, "*R.U.S.E.* couldn't exist in its current form. As a reference, *Act of War* had around 150,000 individual elements on the map, but in *R.U.S.E.* we up that number to 25 million objects. Computers aren't 1,000 times more powerful, but spreading the load over multiple threads has the same net effect."

From the outset, Le Dressay knew that he had to make his engine multi-threaded, not only for the benefit of *R.U.S.E.* but also to guarantee its future. Thanks to this foresight, IrisZoom can scale up into as many cores as it has available. Getting that granularity right and splitting the task load into as

many separate parts as possible means that future versions of the engine won't have to be rewritten from the ground up.

"Multi-threading coding is still difficult," Le Dressay said. "It's easy to make mistakes. But it's the golden path to performance, and if you want to get the most out of your game you have to think in terms of multicore processing. Like most developers, we divide the workload between AI and 3D, but it's a bit more complex than that. Both parts of the engine have a lot of asynchronous tasks, like gathering information about which elements need rendering or making pass-finding requests that analyze strategies."

In *R.U.S.E.*, the more threads that are available, the more detail the player will see on the battlefield. Processor cores dictate how far certain effects are visible and where texture calls occur.

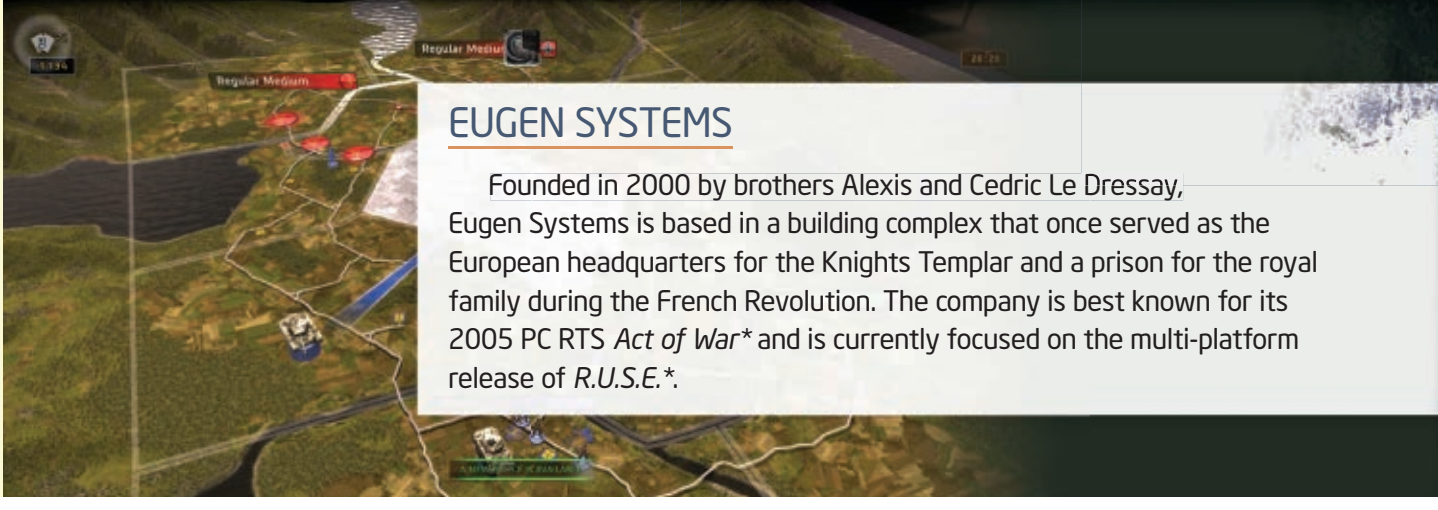
"You have to be able to see clearly what's happening to every unit on screen, no matter how distant it is, and the level of detail we use is dependent on the processor," explained Le Dressay. "If you have a single core, the FX will be basic but effective, while on a quad core you'll see everything in great detail."

Le Dressay knows that the technology is scalable into the future, thanks to the six-core Intel® processor code-named Gulftown working in the office. He is also developing exclusive bonus visual content for gamers who make the investment in the next generation of Intel® Core™ i7 processors.

A Mutually Beneficial Connection

"Our relationship with Intel has been mutually beneficial," Girard told us. "We have a technology that's a perfect benchmark for multicore, because *R.U.S.E.* isn't just broken down into three or four threads, it's made up of lots of small tasks that can be balanced across the whole CPU regardless of the number of cores."

Throughout the development process, Intel has provided Eugen with early access to forthcoming



EUGEN SYSTEMS

Founded in 2000 by brothers Alexis and Cedric Le Dressay, Eugen Systems is based in a building complex that once served as the European headquarters for the Knights Templar and a prison for the royal family during the French Revolution. The company is best known for its 2005 PC RTS *Act of War** and is currently focused on the multi-platform release of *R.U.S.E.**.

hardware for testing and analysis. Most important, though, has been the constant feedback from software architects with well-established skills in multi-threaded applications.

"Intel's engineers receive regular builds," Girard said, "And help us find the bottlenecks and what we can optimize. They've really helped us get the game running well on low-end hardware."

For Le Dressay, the Intel relationship has been important because, as he puts it, "the PC is the home of strategy gaming." Despite the fact that *R.U.S.E.* will be a multi-platform release, every member of the Eugen team is kitted out with a high-end Intel Core i7 processor-based machine for development work.

"That makes our compilation times much shorter and helps productivity," Le Dressay said. "It makes the daily job of every programmer more enjoyable. It's simple: the faster your PC is, the better you can fix bugs or add features and the easier it is to achieve your goal: making great games."

It doesn't, of course, have anything to do with the lunchtime games of Left 4 Dead* or the amount of time the team spend simply playing their own game to perfection. Honest.

Merely a *R.U.S.E.*

Considering the intricacy of its underlying technology, the most interesting thing about *R.U.S.E.* for many players will be the counterespionage system from which it takes its name. As a battle unfolds, generals are rewarded with cards that can be brought into play to launch a subterfuge tactic against the enemy.

"We used the World War II settings as inspiration for the *R.U.S.E.* tactics because there were lots of examples of subterfuge," said Girard. "There was the enigma machine, fake cities burning in the night to lure bombers, disguised commandos in the Ardennes. We interpret those with specific skills that allow you to create decoy buildings and units, to hide information like the position of an army and steal information like what orders a general is giving their units. Finally you can manipulate the psychology of units, by making them flee or fight to the death and so on."

It's an original way of varying the traditional RTS mechanic, especially in the promising multiplayer environments, which can support up to eight players in team-based or free-for-all combat.

Girard demonstrated the deception system in action. Stabbing an index finger at the infantry unit in the middle of the multi-touch Sony Vaio* L screen that sits in the middle of Eugen's office, he ordered the unit to vanish into a nearby forest. Hiding troops is the most basic of several deception cards you can play.

On the surface, it would appear that the deception cards have little to do with the rendering pipeline. That, said Girard, is exactly how it should be. Players should be unaware that the large view distances, single map view, and gesture controls are vital to making this element of the game work.

And that's the point, he concluded. It's not a question of how clever you can make your engine, but what you do with it. And with huge battlefields, innovative controls, new interfaces, and intriguing tactics, *R.U.S.E.* is doing more than most. ▪

ABOUT THE AUTHOR

Adam Oxford is a freelance writer who specializes in games and technology. The former editor of PC Format *magazine in the U.K., he has written for* PC Gamer, *TechRadar com, Edge, GamesMaster and more..*

A PARALLEL UNIVERSE — THE FINAL FRONTIER

TAKING GAME PERFORMANCE WHERE IT HAS NEVER GONE BEFORE

DISCOVER NEW FEATURES AND CAPABILITIES OF INTEL® GRAPHICS PERFORMANCE ANALYZERS. REDUCE BOTTLENECKS AND SURPASS USER EXPECTATIONS ACROSS A UNIVERSE OF PLATFORMS.

"Intel's Graphics Performance Analyzers suite is a robust, well-supported package useful for a variety of what-if scenario optimizations and system metrics. If you're targeting Intel® HD Graphics specifically, the IGPA suite will be of tremendous help." — Jimb Esser, Lead Graphics Programmer, Cryptic Studios

Your destiny begins here — Download the free trial of Intel® GPA now at: www.intel.com/software/gpa

CRYPTIC

STAR TREK ONLINE

VISUAL ADRENALINE intel Software

AUDIOKINETIC WWISE 2009.3

REVIEW BY DAMIAN KASTBAUER

FOUNDED IN 2000, AUDIOKINETIC began production on its Wave Works Interactive Sound Engine (Wwise) in order to create "cutting-edge audio solutions" for game developers. With the arrival of the recent 2009.3 iteration of its audio middleware suite, Audiokinetic has accomplished its initial goals and continues to set the bar for game audio.

A combination audio engine and implementation toolset, Wwise is a full-featured audio solution for the playback of complex sound behaviors in games. The toolset is organized through a series of workflow-specific views and editors which include several feature-specific hierarchy structures. Everything from importing sounds and their playback properties, to defining parameters, managing sound banks, adjusting 3D attenuation, and a wealth of other functionality can be directly accessed and utilized with an existing game engine or as a prototyping solution without access to a game. The interface goes deep and allows for a high level of interaction between the different elements.

GETTING HIERARCHY

>> In what seems to have become a standard for audio middleware, the event system abstracts sound content from direct reference to allow for additional functionality and workflow enhancements.

The actor-mixer hierarchy is where an audio asset begins its trajectory through the Wwise engine. Everything is nested in a parent/child type relationship of containers and folders, offering a plethora of options and values that can be applied in order to stylize the payback of your sound content. Through the user interface, all of the properties that govern the playback of a sound or group of sounds can be specified or inherited at any level of nesting. Within the different container types, sounds can be randomized, sequenced, switched or blended, all while sharing common values such as volume, pitch, low frequency effects, low-pass filter, and parametrization. The 2009.3 update also allows for sound in a random or sequence container to be crossfaded by both constant amplitude or power, an extension of the previous "amplitude only" crossfade.

While these myriad features and values can be a boon for late night tweakers and diligent implementers, it's often difficult to assess the multitude of values at a glance, especially when values are spread between tabs or buried within container-specific tools. Smart template creation and good internal documentation can help alleviate some of the pain, but there's a price to pay for so much flexibility in such a small space.

The master-mixer hierarchy allows for the routing of sounds into a user-definable mixing bus structure in which you can apply effects, perform changes to the mix based on states or parameters, as well as apply ducking rules to interactively mix the channels. The inclusion of a customizable mixer and the ability to dynamically mix at run time is a giant leap forward for audio middleware and has only recently become possible thanks to the increased processing power of current consoles.

> With the addition of McDSP plug-ins in version 2009.3 we are seeing some of the first pro audio crossover products available in game audio. The new FutzBox Lo-Fi Distortion effect provides a creative way to "dirty" your audio signal or simulate the sound of low-fidelity devices.

SWITCHES AND STATES AND BLENDS, OH MY!

>> Embedded within each container type is the ability to interactively affect the sound by using parameters coming from the game and set up within the tool. These can then be used by way of a graphing tool at each level of the hierarchy to control variables like volume, pitch, and low-pass filter, as well as advanced functionality like instance limiting, priority, and effects. At face value this may seem like a feature that's been available for years, but the ability to prototype this behavior in the tool and use the parameter data to control other elements is where Wwise really shines. Parameters open up a new way of thinking about implementation when used to control game syncs like switch, state, and blend containers, and they enable a host of features to assist in the transitions between content types.

The blend editor allows for the layering of containers and provides the additional ability to visually manage crossfading and other parameters across different sets of content. Although the interface has some visual and workflow hangups that prevent it from being fully intuitive, there is great creative potential here.

FACE THE MUSIC (SYSTEM)

>> The lure of fully interactive and dynamic music has existed just out of reach for most developers because it requires significant engineering and resources to support. In a sweeping statement of intent, the Wwise interactive music system has leveled the playing field by providing a comprehensive solution for composers who are looking for new ways to squeeze variation and emotion out of their interactive scores. Everything you need to intelligently control and play back music-specific content is available within the toolset and ready to audition using the same switch, state, and parameter system available throughout the project. Along with the host of values common to other types of containers, the music containers include additional values for tempo, time signatures, bars and beats, transition management, and the ability to trigger stingers in time with the specified music. While the potential of interactive music has been around for some time, tools that enable composers to try out different

CONTINUED ON PAGE 49

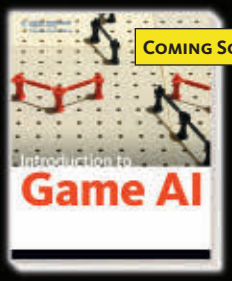
Visit the GDC 2010 Bookstore to purchase our titles and save $10!

Your Ultimate Resource

Course Technology PTR is your ultimate game development resource. Our books provide comprehensive coverage of everything from programming, story development, character design, special effects creation, and more, for everyone from beginners to professionals. Written by industry professionals, with the techniques, tips, and tricks you need to take your games from concept to completion.

Game Programming Gems 8
1-58450-702-0 • $69.99

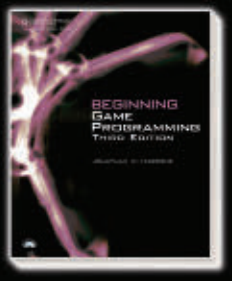
Wii Game Creation for Teens
1-4354-5555-X • $34.99



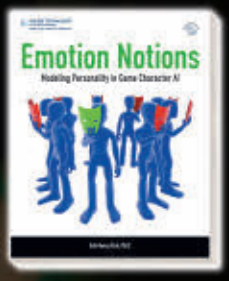
Introduction to Game AI
1-59863-998-6 • $39.99

XNA 3.1 Game Development for Teens
1-4354-5438-3 • $34.99

Introduction to Game Development Second Edition
1-58450-679-2 • $69.99



Beginning Game Programming Third Edition
1-4354-5427-8 • $34.99



Emotion Notions
1-59863-527-1 • $49.99

Virtual World Design and Creation for Teens
1-59863-850-5 • $34.99

Getting Started with Game Maker
1-59863-882-3 • $34.99

Attending GDC 2010? Visit the GDC Bookstore to browse our comprehensive title list. Our books will be sold exclusively at the GDC Bookstore during the show. **Bring this ad and save $10 on your bookstore purchase.** ONE COUPON PER PERSON, ONE TIME USE.

Also available at Amazon, Barnes & Noble, Borders, and other fine retailers nationwide. Visit us online at **www.courseptr.com** or call **1.800.354.9706**.

COURSE TECHNOLOGY
CENGAGE Learning™
Professional • Technical • Reference

courseptr.com

CONTINUED FROM PAGE 47

strategies have traditionally been strictly "behind closed doors" due to their proprietary nature. For anyone with the desire to plumb the musical depths of what is available, the reward will be the ability to prototype and audition systems using game parameters without the need for any actual game interaction.

THE MAIN EVENT

>> Everything discussed so far has been in preparation for the inclusion of various containers in Wwise Events which will eventually be referenced by the game engine in order to play, stop, mute, switch, trigger, and adjust values related to sound playback. This additional level of abstraction allows for valuable interaction not exclusively related to playing back or stopping a sound, and further empowers the user to make choices about how an event affects other aspects of sound. The ability to audition and prototype outside of the game engine allows for greater experimentation, iteration, and flexibility in order to make sound related choices, increase player immersion, and maintain the sound design direction for the game overall.

MULTI-PLATFORM MAJESTY

>> Wwise is designed to facilitate multi-platform development. Through the UI you can visualize functionality that may not be supported on a given platform and choose to exclude variations to minimize bank size, or adjust per-platform conversion and language settings to make localization easier. As a multi-platform developer using Wwise, you can easily manage your resources to leverage the strengths of your lead SKU without having to recreate everything to support a lower spec or different region. In many ways the challenges of multi-platform audio development have been elegantly solved.

TAKE IT TO THE SOUND BANK

>> Due to the custom nature of most game engines, the sound bank loading process is often a combination of game side scripting and intelligent sound bank management. By providing an accessible and inclusive interface for managing logic and sound, and extended functionality for programmer-specific processes, Wwise has given users a flexible environment to prepare and implement a loading strategy to fit most games. Any platform-specific inclusion or exclusion is reflected in the sound bank editor, along with the ability to further exclude assets in special cases or workflow, and the ability to organize at any level using folders. Sound banks are then generated to a specified location for each platform, with the additional ability to use the file packager utility to perform post-build command line functions.

CAN'T WE ALL JUST SHARE?

>> Wwise share sets are work units that include digital signal processor (DSP) effect presets that can be subscribed to at any level in the actor-mixer or master-mixer hierarchy. Attenuation profiles can be subscribed to by any actor-mixer, container, or sound source in the actor-mixer hierarchy. Attenuation profiles also allow you to graph sound propagation over distance, with the added ability to graph additional attributes and control the directionality of sound along with other special effects that can be used to modify sound over distance.

With the addition of McDSP plug-ins in version 2009.3, we are seeing some of the first pro audio crossover products available in game audio. The new FutzBox Lo-Fi Distortion effect provides a creative way to "dirty" your audio signal or simulate the sound of low-fidelity devices like telephones, televisions, or radios while the ML1 Mastering Limiter effect will be familiar to anyone trying to reign in out-of-control dynamics and transients. Custom effect share sets can be created, used, and modified as either a preset share set instance, or defined as custom for the specific instance in which it is being used. Outside of the toolset and audio engine, the real power of digital effects processing lies in the ability to manipulate effect settings based on incoming game parameters at runtime and Wwise gives you plenty of values to adjust.

LAY IT ALL OUT

>> Work units act as the building blocks that make up the various hierarchies. Everything from actor-mixers, events, game syncs, and sound banks can be created, managed, and checked in and out individually. Outside of the toolset, work units can be opened in a text editor. Where multiple people may be working in the same work unit, most merge utilities can resolve changes between updated files. The scalability of these work units allows for multiple people to work simultaneously within the same project, and it is also a convenient way to exchange implementations between Wwise projects. By simply copying a .wwu to the appropriate folder in a different Wwise project, the implementation will be found the next time Wwise is opened. Additional work may be needed to maintain file paths, but the ability to move implementations around between projects makes the work unit extensible when working in a multi-project environment.

To further accommodate the synergy between game and audio development, Wwise includes source control integration with Perforce functionality built in. Working in Wwise with source control allows for the automated checkout of any work units being edited as well as the importing of new content in order to keep your asset directories up to date.

To further accommodate the synergy between game and audio development, Wwise includes source control integration, with Perforce functionality built in.

PROFILING THE GAME AUDIO MIND

>> One of the great promises of previous-generation audio toolsets was the ability to connect a project to the game while running and make changes to sound playback in real time. While there have been

CONTINUED ON PAGE 51

Audiokinetic WWISE 2009.3

Audiokinetic Inc.
409 rue Saint-Nicolas,
bureau 300
Montréal (Québec)
H2Y 2P4
Canada
www.audiokinetic.com

PRICE

Non-commercial projects/prototype: Free.
Console titles: $15,000 for the first platform, $7,500 for additional platforms.
Electronically distributed games: $5,000 for the first platform, $2,500 for additional platforms.
Options such as Wwise Motion, SoundSeed, and McDSP are sold separately.

SYSTEM REQUIREMENTS

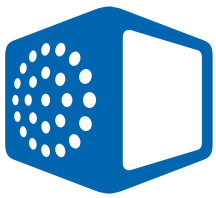
Pentium 4 processor 3.0 Ghz or better with Hyper-Threading technology, AMD Athlon 3.0 GHz or better.
Windows XP with service pack 2, and XMLLite for Windows XP Service Pack 2. DirectX October 2006 or later. The Wwise authoring application is available in both 32- and 64-bit versions. 1 GB of RAM.
Wwise works with any sound card or integrated onboard audio.

PROS

1 Fully integrated audio pipeline solution from low level engine to toolset integration.
2 Robust toolset features, functionality, and prototyping capability.
3 Elegantly solved multi-platform authoring.

CONS

1 Windows PC based.
2 Could use more 3D DCC tools for a faster export of assets.
3 Only works with 3D DCC tools that can export ASE or DAE.



CRIWARE™

Audio

CRI Vibe™ 1.5

Advanced Audio Toolkit
for
the Nintendo DS

- DMA-driven micro-streaming from ROM Cart
- ADPCM support for Sound Effects playback without MIDI
- New CODEC for higher compression ratios
- Simple ROM bank eliminates PRAM audio bank management

www.crivibe.com

Video

CRI Movie™ 2.1

Hi-Def Video Engine
for
the PS3,PS2,Xbox360,Wii,PC

- Optimized Hi-Def playback / low-bitrate movies
- Sample frameworks rewritten to help you get started faster
- Documentation revised based on customer feedback
- Easy-to-access SDK's and FAQ's via our new support portal

www.crimovie.com

CRI Middleware, Inc.
Audio & Visual Solution Company

www.cri-mw.com

GDC 10

Schedule a meeting with us at GDC
to see our latest technology first hand -

crimw.eventnewscenter.com

E1002GDM-A

product news

PATHENGINE SDK RELEASE

PATHENGINE
www.pathengine.com

PathEngine released version 5.23.00 of its pathfinding and agent movement SDK. The SDK includes streamlined mesh loading and setup pipeline and support for automatic ground mesh construction directly from 3rd party physics provider scene data.

New to this release is the ability to switch to a more memory efficient 3D partitioning data structure for resolution from world to ground, the option to save out the partitioning (and other mesh related data) with the ground mesh file for faster loading, a helper method for stripping out parts of a ground mesh not reachable from a specified set of "root" positions, as well as improvements in position resolution and improved interface checking. The SDK also includes code for running the PathEngine 3D content processing functionality directly against PhysX or Havok scene data.

The SDK is built around points-of-visibility pathfinding over three-dimensional ground meshes and enables pathfinding and collision to be integrated with a single agent movement model. Agent shape is taken into account exactly, with support for overlapping geometry and dynamic obstacles that are directly integrated into the core movement model.

TRINIGY LAUNCHES VISION ENGINE 8

TRINIGY
www.trinigy.net

The latest version of Trinigy's game engine expands its multi-platform support (PC, Xbox360, PlayStation 3 and Nintendo Wii) by bringing the Vision Engine to browser-based games through a new browser plug-in called WebVision that is downloadable for all common web browsers.

Vision Engine 8 adds a host of new graphics features including support for Microsoft DirectX 11 graphics processors and Shader Model 5.0 for soft shadows and tessellation. It adds a new water rendering system for easier creation of realistic water surfaces from rivers to oceans. Vision 8 also provides a new modular post-processing system that integrates with both the forward and deferred renderers of the Vision Engine. New post-processing features—such as a new sun glare renderer—are included as well.

Trinigy has extended the Vision Engine's resource viewer to support all major platforms (Xbox360, PS3, and Wii). It offers integration with Perforce to enable better versioning of assets and more secure asset management within complex production environments, as well as a new debugger for LUA scripts that allows developers to inspect and debug the script code of a running game. Vision Engine 8 has also been optimized to run on Intel six-core, hyperthreaded processors.

Additionally, the new version of Vision Engine includes extended Havok Physics integration and a connection to the Havok Remote Debugger. Vision's built-in sound system has been extended and optimized to provide streaming performance and support for additional sound formats.

SUBSTANCE REDUX RELEASED

ALLEGORITHMIC
www.allegorithmic.com

Allegorithmic has released Substance Redux, part of the Substance Air toolset. While Substance Air is used to generate content, Substance Redux was created to fulfill the need to reduce the size of already-produced textures for online games. While dramatically reducing downloadable client size by close to 50 percent, Substance Redux condenses these bitmap textures within a few seconds without noticeable loss of in-game quality or performance. The key features of Substance Redux include detailed texture map analysis and optimal compression scheme, use of custom automatic filters to improve quality, ultra fast in-house DXT compression at runtime, and plug-ins for Epic's Unreal Engine 3 and Emergent's Gamebryo engine.

HANSOFT VERSION 6.0

HANSOFT
www.hansoft.se

Hansoft is an integrated solution for agile and lean development, collaborative Gantt scheduling, real-time reporting, bug tracking, QA, workload coordination, and portfolio and document management. Among the new features in Hansoft 6.0 are a prioritizing functionality that merges the benefits of prioritizing categories with numbering, enterprise license management that enables user accounts to be shared across geographical locations, servers, and databases, and improvements in usability and workflows around quality assurance and working with multiple bugs or tasks at once.

CONCERITY ANALYTICS LAUNCHED

CONCERITY
www.concerity.com

Concerity announced the launch of Concerity Analytics, a new type of integrated toolkit designed specifically to help software companies measure real-world usage of their software applications.

Designed for Windows desktop and ASP.NET software products, Concerity Analytics allows software creators to gather key data on how their software products are used in the field by end users, either in a test environment or in production. The Concerity Analytics toolkit provides software teams with the ability to measure and optimize feature usage, understand beta tester behavior, and prioritize product development efforts.

CONTINUED FROM PAGE 49

realizations of this concept in both proprietary and middleware tools, it has never been easier or more comprehensive than with Wwise.

Any platform currently running a game can be connected to a corresponding project, where values can be adjusted in real time for most sound attributes. It would be easy to overlook this functionality were it not for the addition of the robust profiler, which can monitor every aspect of the audio engine. This means that data related to events, sound banks, streams playing, or sound voices used can be captured during gameplay and examined to balance resources across all areas. Several statistics can be graphed via the performance monitor to identify problem areas in need of optimization. New to the profiler in version 2009.3 are improved error monitoring messages and a profiler statistics view specifically used to display statistical information related to dynamic dialog. Future support for other queries is expected to broaden the scope of this new feature. The combination of connectivity to affect changes as well as monitor usage is an indication of Wwise's ability to handle all aspects of the audio engine and maintain ownership of the runtime audio pipeline during production, something your programmer will thank you for.

Not every project will use everything that comes with a Wwise license, but once you get locked into serious audio implementation the tendency is to push the limits as far as you can. There is nothing in the world more frustrating in game development than not having access to features that could maximize the quality aspects of production. Thankfully, "access" and "features" are what Audiokinetic delivers.

DAMIAN KASTBAUER *is a freelance technical sound designer working with the Bay Area Sound Department, pulling off cool implementation tricks, experimenting with noise, and spreading the word about interactive audio. His contributions can be heard in* Conan, Star Wars: The Force Unleashed, *and* The Saboteur *among others, while additional articles on technical sound design can be found linked at* **www.lostchocolatelab.com**.

Pyramind Studios




Music VO SFX

Compose · Record · Mix · Master
Implement · Sync · Deliver · Smile

Select Clients Include:
SEGA of America
2K Marin
Sony Computer Entertainment
LucasArts
Ubisoft
Electronic Arts

Recent Projects Include:
Iron Man 2 - The Videogame
BioShock 2
God of War I & II, MAG
Lego Indiana Jones 2
H.A.W.X.
Spore

415.896.9800 · www.PyramindStudios.com · San Francisco

HYUNG-TAE KIM

INTERVIEW

HYUNG-TAE KIM IS PERHAPS THE BEST KNOWN Korean game developer—even if his name isn't known by all, his art is very recognizable, as seen in the later entries to the WAR OF GENESIS series, and MAGNA CARTA for PC and PlayStation 2. His style exaggerates the female, while also promoting the masculine. His characters straddle the line between Japanese minimal lines and Western emphasis on musculature and detail.

I've long been impressed with his work, especially his willingness to go outside the constraints of human physicality to make appealing characters. All of his characters, male or female, are soft and curved in ways that make sense, even if they are not tied to real anatomy. For Kim, it's all about flow, and in the past most of his actual art was relegated to 2D. Now that he's working on BLADE & SOUL, NCSoft's next big MMORPG after AION, he has the chance to completely control his work in 3D as an art and technical director on the game.

In this interview, conducted in his native Seoul, we discuss his process, his thoughts on anatomy and character, and his influences.

Brandon Sheffield: ***Can you tell me, step by step, your process for character design? When you're designing one character, how do you go from envisioning that character to the actual process of making them into full illustrations?***
Hyung-Tae Kim: First of all, you have to decide the direction of the character, in concert with the design teams. You have to know where the character will be used, and in what manner. And then I have to actually begin the sketch, which is the most difficult part of the process. What I take into greatest consideration when doing my sketches is first, whether this style reveals the personality of the character, and then if it will be attractive in three dimensions. It didn't really matter when I was working on WAR OF GENESIS though, because it was 2D.

Then after the sketch is finished, we start the color planning, deciding which colors we want to use. If the process goes well, it can end in two days. Sometimes it can take as along as three weeks. Then for a final illustration, we would polish by zooming in and fleshing out all the details.

BS: ***Do you have assistants drawing in your style, or are all illustrations in your games by you?***
HTK: Until recently, all illustrations were completely done by me. But for BLADE & SOUL, which is an MMORPG, it's a game that requires a lot of characters and a lot of design. So now there's a team doing our drawings. Besides me, we have eight artists working on the images you see.

MAGNA CARTA's male main character Calintz was deliberately given feminine features to make the audience feel somewhat uncomfortable.

BS: ***And are they drawing your style, or adding elements of their own?***
HTK: When we hire people, I try to recruit artists who naturally can understand and draw the kind of image I need. It's not that I need them to draw exactly in my style, it's more important that when it transitions into 3D it'll be up to my standards and fit my direction.

So at first the artist's style could be different, but since the beginning of the BLADE & SOUL project I've had everyone practice drawing these characters and images in a certain style. Everyone's pushing for the same end result with the 3D models. That's actually what I did from the beginning. I provided a 3D polygon model so that everyone had a specific goal to move toward, and the direction to get there. It's all basically an attempt to maintain a consistent style.

BS: ***I've noticed with 2D illustrations in Korea that there's a more frequent use of vibrant color. I'm wondering if you have any theory why that is, why color is so well used?***
HTK: One of the characteristics of Korean digital art is that it's pretty energetic. I guess that's something to do with the cultural background of Korea. Of course I can't speak for others, but in my case, I'm not really comfortable with a stable kind of picture, with something balanced, because what is average is so well-represented in reality. You see it anywhere you go. I want to express through my artwork something that's not possible in reality, something that cannot exist.

BS: ***One thing that's very distinctive about your style is how you draw women. How do you approach this?***
HTK: I'll preface this by saying that when you're drawing something that's related to culture or region, it's incredibly difficult, because trends change so often. This is the hardest thing to depict with character, for me.

For example, there are people who like strong characters, weak characters, innocent characters, feeble characters, on a region-by-region basis. But that sort of thing changes very often. It becomes very complicated. So I try to simplify everything by erasing all the cultural elements, and try to make it more internal or instinctive. In other words, it's the natural identification of what we really like since birth, before cultural context. I try to exaggerate all those aspects and then tie it to some of the more complex or deeper emotions, and then just draw something that people will really like innately.

BS: ***Is that "back to basics" approach why the women that you draw tend to have an emphasis on chest and hips? Kind of an Earth mother element?***
HTK: To put it simply, you have to first look at how the body is actually connected. Of course, the chest and hips sort of stand out a lot because they're, you know, the biggest parts I guess. But the most important part of the character is actually fat. Any character we create is composed of bones, muscles, and of course fat.

Despite the beauty of bones and muscles, those aspects tend to not be particularly feminine, and lend themselves to something more male. But if you focus more on the fat of a character, and then you sort of create a flow into the chest and the hips and form the body around it, how force and physics can change a body, including fat, it becomes more beautiful for people who can appreciate that innate nature. In books that explain the body, there aren't that many explanations of fat, so it's really hard to find this kind of information actually.

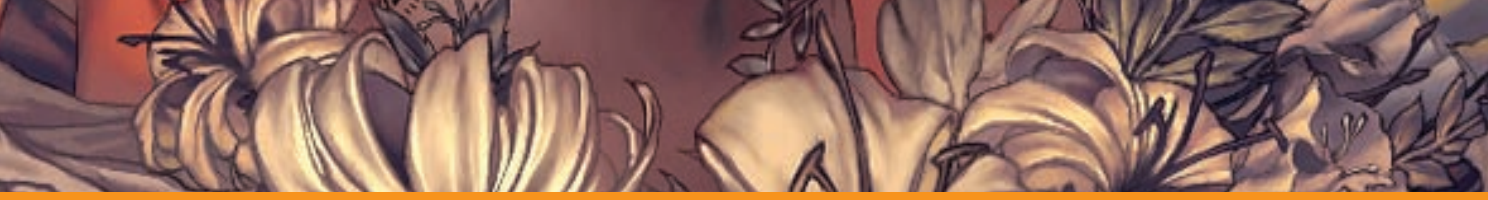
BS: ***I was wondering how much you pay attention to anatomy. Sometimes, it feels like you may stretch people a little bit, altering their actual skeletal structure.***
HTK: I do try to exaggerate my characters, but only to the point of still being able to perceive them as human. But then I try to exaggerate those parts that people will find most attractive, like when a man looks at a woman, or a woman looks at a man. Especially, for example, as you mention, the pelvis or the hips in women. I do accentuate the bones and the fat around the body, which makes it a lot more attractive. One problem that I have through this process is that when I exaggerate this to the maximum, the character starts to become inhuman. And then there's a clash between the two thoughts of what I'm trying to draw. I'm constantly having this tug-of-war between these two ideas when I'm making new characters.

BS: ***I noticed that in, for example, MAGNA CARTA, you also drew the main male character with feminine hips. Does your approach change for male characters as well?***
HTK: That was just in the case of MAGNA CARTA, because the main character was supposed to kind of make the audience feel a little awkward. That is why the clothing and also the anatomy of that character was more feminine than usual. I think that's not the most attractive element of male characters. I wouldn't draw a normal male the way I drew him.

BS: ***What is the kind of essence that you try to bring out for male characters usually?***
HTK: Of course, everybody knows that the attractive aspects of male characters are different when seen from the perspective of a man or a woman. One thing that people tend to miss is the importance of the waist. People say that the shorter the waist, the better it looks. But the actual thing is the more detail you add to the muscles, how the waist or stomach moves and how it changes when the muscles move, that's one thing that's really attractive when looking at a male character.

BS: ***How did you wind up developing your style?***
HTK: Well, I really took a lot of things from Japan, not only comic books and games. In my early days I really liked a lot of things from Japan. When I started studying art though, I actually preferred Western styles of painting. I tried to combine both that Western painting style with Japanese style content, and that's pretty much how I got here.

BS: ***How do your goals change, if they do change, from 2D illustration to 3D model?***



HTK: To be straightforward, it's not exactly that 3D is our goal. To be more precise, in 3D I'm trying to express the attractiveness of the 2D paintings that I made. By moving my art from 2D to 3D, I open up the audience for my work. In the past, my true artwork was only on the cover of the game—inside it wasn't necessarily my work. Now, I have larger audiences through actually representing my characters in my own way in 3D.

BS: ***With 2D, it's obviously easier to direct the eye of the viewer because you're completely controlling what they see. With 3D, how do you make sure they see what you want in a character ... Or can you even do that?***
HTK: Once it becomes 3D, it is true that there are limits to the kind of structural exaggeration I like to insert in my characters. It's especially hard to visually express what I want when it's put into gameplay, because it's not so much about looking cool, it's about how people can enjoy the game, how they play it. So what we try to do is simply input camera technology and light technology that can make the structure of the image as beautiful as possible.

BS: ***What do you feel is the ideal background scenario for your characters? Is the background meant to be a main feature, or does it just accent the characters?***
HTK: Regarding backgrounds, I provide the direction of the concept art. For the most part it's drawn by our specific background artists. In this case, more than with characters, I try to leave it to them rather than doing it myself. But I do try to set the background in relation to the culture and character of each race in the game, and once I provide the direction and the details of how it should be drawn, then the artists who are in charge of the background tend to work on it.

BS: ***Would you ever be interested in making a game that's completely in 2D, where you would have complete control over how the characters would look?***
HTK: I've been thinking of different ways to express my 2D art. One of them, of course, would be a 2D game. But I'm also thinking of what else I can do. I can't mention anything in detail though. Right now, the first objective for me is to shape BLADE & SOUL in my own style.

BS: ***You have said that these characters are fantasy oriented. Do you worry at all about... Well, sometimes these exaggerated, really attractive characters can create an unrealistic expectation for a player when they view reality.***


BLADE & SOUL.

HTK: Exaggeration of certain points is really important, but overall the character must be understandable. In other words, it must be human. The person who looks at it must be able to understand it. In some ways, this is the case with the art style of Japan, where the attractiveness of characters are sometimes symbolized.

BS: ***Yes, it can often be idealized.***
HTK: Yeah. When people see something, it looks really attractive, but they're not sure why. This is because the symbol of attractiveness is hidden within the character. I don't want to make that attractiveness as impenetrable as in the Japanese style. When a person sees a character and thinks it's attractive, he'll know why it's attractive. It will be right there on the surface. I think it's alright as long as it's feasible. It doesn't really have to be able to exist in reality. But there does need to be a balance in between.

BS: ***Do you have certain character archetypes that you rely on?***
HTK: I try not to do that because the character becomes symbolized, which is something I'd rather avoid. But it would certainly make things easier. I think the reason is that I'm not just creating a character when I create these people. I'm creating a painting, a piece of digital art. There are times when we need a lot of flashy characters. So I can't just work from a template. You need these characters to have, well, character! This is really important to the quality of the game overall.

BS: ***What artists do you personally admire?***
HTK: Too many of them actually. For example, among painters, Jeffrey Jones. Range Murata, Masamune Shirow...but actually, the artist who influenced me the most was the illustrator of LINEAGE 2, Joong Ho Chung.

I've known him for 20 years, since I was in middle school. His kind of painting and drawing have really influenced me a lot. We actually have painted together over the years.


BLADE & SOUL.

BRANDON SHEFFIELD *is the editor in chief of* Game Developer *magazine. He has absolutely no problem determining the difference between fantasy and reality.*

fmod™

FMOD is a cross-platform audio middleware solution enabling developers to add the latest audio technology to their titles with a powerful API and FMOD Designer authoring tool.

World of Warcraft: Cataclysm Lego Universe Borderlands Guitar Hero Arcade Grand Ages: Rome East India Company Demigod Need For Speed: Shift Torchlight DJ Hero Alan Wake Topspin 4 Drawn To Life 2: The Next Chapter Aion Bioshock 2 Dragon Age: Origins L.A. Noir BMW Operation Flashpoint 2 Guitar Hero: Metellica Moto GP 09/10 Madden NFL '10 (iPhone) Doom: 2012 Geforce Demo Little Big Planet Starcraft 2 Batman: Arkham Asylum Guitar Hero 5 Transformers 2 Spongebob: Truth or Square Warhammer 40,000: Dawn of War 2: Chaos Rising Galatrix Brutal Legend Forza 3 WWE Smackdown vs. Raw 2010

Visit us at booth 1524

GDC 10

FMOD Ex API

- The most efficient low level audio engine
- Multiple engine integration now featuring full Unreal 3 integration and support. (CryENGINE®, Unity, Vision)
- CELT codec support - better than MP3 and OGG and no license fee
- Voice chat powered by Dolby® Axon
- Cross platform 7.1 surround sound powered by Neural Audio™ by DTS™
- Personalised 3D audio headphone mapping powered by MyEars™

FMOD Designer 2010

- The event editor allows for complex audio models such as car engines, dynamic background ambience and crowd simulations to be created.
- Randomization
- Mix audio in game over network
- Bulk editing
- Interactive Music System
- Wave Bank Management and Localization
- Realtime Profiler
- Auditioning in 3D with FMOD Sandbox

Firelight Technologies Pty Ltd. | FMOD Sound System
www.fmod.org | sales@fmod.org | support@fmod.org | T: +61 396635947 | F: +61 396822001

An example of virtual textures can be seen in the early screenshots of id Software's upcoming RAGE.

VIRTUAL TEXTURES

STREAMING MANAGEMENT FOR OUT OF CORE RENDERING

ONE OF THE CENTRAL TECHNICAL CHALLENGES IN MODERN GAME ENGINES is managing large quantities of data. Typical games pack dozens or even hundreds of hours of content into the handful of gigabytes available on a DVD, and then employ combinations of simple compression and streaming techniques to manage a smaller working subset of a few hundred megabytes of relevant data in memory. The power and programmability of modern GPUs now enables the practical use of more advanced out-of-core streaming techniques, such as virtual textures. A virtual texture system, as the name implies, allows very large texture resources to be broken up into chunks and assembled into a database for streaming. Peering into the future, voxel rendering systems can make use of the natural extension of virtual texture streaming to three dimensions. In this article I will present an overview of a virtual texture system as suitable for modern consoles and the challenge of effective streaming management.

VIRTUAL TEXTURES

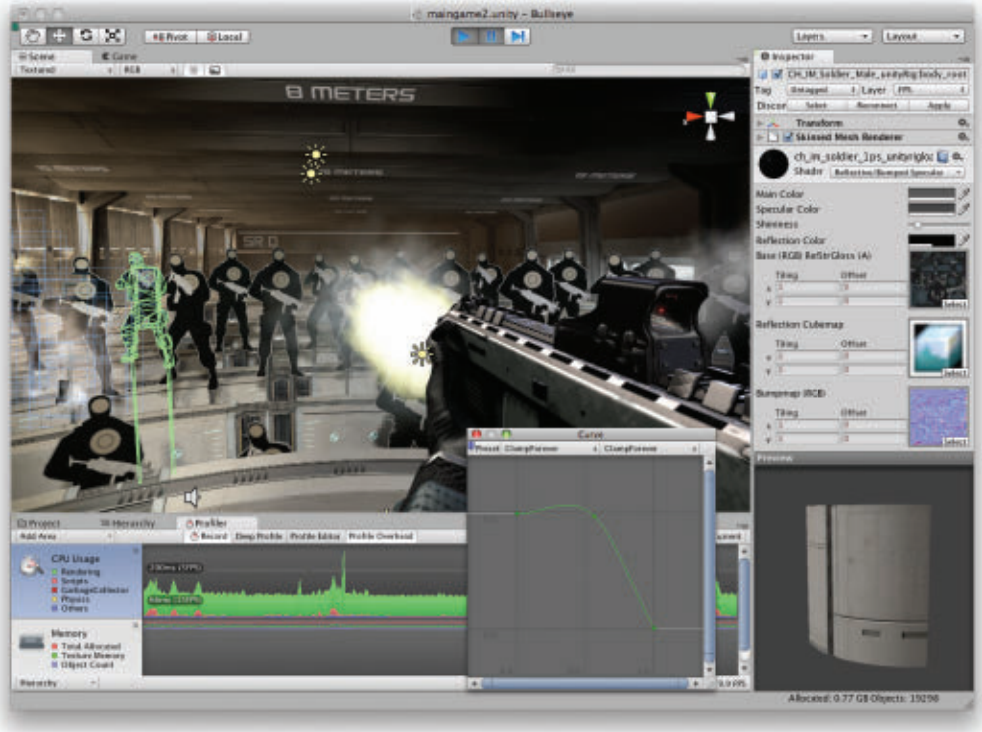
>> Virtual textures have been around in some fashion since the early days of computer graphics. Flight simulators and terrain navigators such as Google Earth aim to model large sections of the earth's surface at high fidelity, so their terrain engines typically break up a large 2D surface into manageable chunks for streaming and rendering. Virtual textures extend these techniques beyond terrain and apply them to general models. The core of the idea is to replace the typical texture mip map pyramid with an indirection structure such as a quadtree, allowing very large mips to be broken down into a set of smaller, more manageable chunks which can be sparsely allocated and reassembled in physical memory. Virtual textures have been popularized more recently as a core feature of id Software's upcoming id Tech 5 engine, and have been researched at Crytek. They have even been implemented in graphics hardware. For a rather extensive overview, see the 2008 SIGGRAPH chapter (in References), which covers many aspects of a practical implementation. I will briefly summarize the concepts and then delve into a couple of the interesting practical challenges.

Each mip level of the virtual texture is broken up into small MxM chunks or tiles, each associated with a node in the equivalent quadtree. Even when using anisotropic filtering, the GPU references at most a handful of unique texels for every pixel, so the working data set for a given frame is bounded to a multiple of the screen resolution. At runtime, a subset or partial expansion of the total quadtree is loaded into memory, corresponding to just the tiles in the virtual texture required for rendering the current scene. These tiles are allocated in physical memory, typically in a packed texture atlas, in an arbitrary fashion. For a terrain engine, each chunk of texture in the quadtree can be directly associated with geometry and rendered, but to generalize this to arbitrary models, we must support rapid random queries into the quadtree texture structure.

The idea is that a virtual texture should become a drop-in replacement for regular textures and should present a similar interface in shader code, allowing a texture fetch directly from the indirection structure.

CONTINUED ON PAGE 59

Author Once, Deploy Anywhere.



UNITY3D.COM

unity

CONTINUED FROM PAGE 57

Given a 2D UV coordinate in the virtual texture space, we could find the appropriate quadtree node and physical offset by starting at the root and then walking through the data structure until we encounter a leaf node or an internode of adequate resolution. However, the large number of dependent texture fetches would make this impractical.

Instead, the quadtree traversal can be dramatically accelerated by flattening out the top levels of the structure into a separate flat indirection array. In a more general sense, this amounts to flattening the quadtree into a shallower B-tree structure optimized for fast queries, a standard approach in large databases. The idea is to store a larger BxB subarray of children offsets at each level, collapsing numerous traversal steps into a single step at the expense of potentially wasteful extra storage.

For an initial example, we could flatten the whole first B levels of the quadtree into a single level, storing a 256x256 subarray of expanded offsets, for example. Each entry in this indirection array would store the physical offset to a level 8 node in the quadtree if it exists, or the offset for the deepest internode containing it otherwise. Besides the offset, we now also need to store an appropriate scale factor for the UV coordinate (based on the node depth). This can be thought of as storing a scaling rectangle into the relevant physical tile for that node. This can be reduced to just a handful of GPU instructions to translate from a virtual to a physical address: a fetch from the indirection texture followed by a frac(x) and a multiply-add. Supporting correct bilinear filtering, mip-mapping, and even trilinear and anisotropic filtering is also possible, but will require careful padding of texture tiles. For more details on padding issues and shader code examples for traversing the indirection structure, see Mittring's paper.

With just a single flattened indirection level and thus a single dependent texture lookup we can support somewhat large virtual textures. A 1024x1024 indirection table with 128x128 texture tiles could support a virtual texture of up to 128,000 x 128,000 in size. This is not sufficient for a large flight simulator-sized world, but is just in the ballpark for a game level. We could push up the indirection level table or tile size a little more, but it starts to waste memory. Larger tiles incur significant overhead for tiles where only a small fraction of the texels are actually used. One fairly straightforward solution is to use numerous virtual textures with appropriately-sized indirection tables, and then use a standard variable-sized texture memory management system for all the indirection table textures. This is something of a messier hybrid, but it's workable, and most streaming engines must already deal with dynamic allocation of variable-sized textures, often implementing some sort of background defragmentation system. In this approach the world is broken up into numerous virtual textures, and from the artist or world builder's perspective they work just like regular textures, except they are much larger.

For a more general approach which avoids the fragmentation issues and allows all the memory to be efficiently packed into atlases, we can add additional levels to the flattened B-tree beyond the first to support any size of virtual texture, although in practice the limited resolution of floating point texture coordinates puts a limit on effective resolution at around a couple million texels in each dimension. This could allow all the game's texture resources to pack into a single large virtual texture. For example, the limits of floating point resolution could be reached with a scheme that uses a single 512x512 first-level indirection table, followed by a second-level composed of 64x64 indirection tiles (packed into their own physical atlas), and physical texture tiles of 128x128 dimensions. In the worst case you will need up to one second-level indirection tile per physical tile, but that worse case is pathological. In practice adding a second level of indirection does not need to occur much memory overhead. The real expense of a two level indirection scheme is the more complex shader code for address translation with two dependent texture fetches in the shader instead of one. Although these fetches are highly coherent, this still adds overhead in the pixel shader. However, the costly address translation need only be done once at the top of the shader if all of the material textures use the same UV coordinates.

A virtual texture system, as the name implies, allows very large texture resources to be broken up into chunks and assembled into a database for streaming. Peering into the future, voxel rendering systems can make use of the natural extension of virtual texture streaming to three dimensions.

RENDERING FEEDBACK

>> For a particular frame, the rasterizer will reference a fairly small number of texels, typically just a couple times the number of pixels, even when using higher anisotropic filtering. This small texel set corresponds to a subset of the total nodes in the quadtree, and this working set is what the runtime management component of a virtual texture system must discover for each frame. Ideally we would have a feedback loop which could tell us exactly which texture tiles or even texels were used during rendering. There are several routes for achieving a reasonable approximation to this working set. One approach is to use visibility queries and associate geometry LOD chunks with the texture tiles they reference. This is obviously straightforward in a terrain system where there is a one-to-one correspondence with geometry chunks and texture tiles, but it could also work for general models with a preprocessing step. However, a very large model could reference so many tiles that it would defeat the scheme, and it would have to be broken up into smaller chunks that match the texture quadtree. Having to preprocess models and break them up into a hierarchy of chunks to match the texture quadtree is possible, but it costs performance, adds complexity, and goes against the grain of using virtual textures as a drop-in texture replacement.

An alternative approach explored by Sean Barrett in his 2008 GDC talk (see References) is to use direct feedback data from rendering. All the models using virtual textures are rendered in a separate pass, into a lower resolution buffer with a simple shader which looks up and writes out the ID of the texture quadtree node referenced. We can get away with a lower resolution buffer because the quadtree chunk granularity is much larger than a pixel. This ID buffer can then optionally be reduced a bit more in a screen pass, and then can be read back on the CPU to write out an array storing a binary visibility value or a rough texel reference count for each quadtree node. This scattering pass can even be done on the GPU, which is a viable option on the Xbox 360 using its stream out feature. On the PS3, the SPUs can perform this scattering pass efficiently, given that the ID buffer rendered out by the GPU resides in main memory and the output reference count buffer is fairly small. The benefit for this extra platform-dependent work is an automatic and reasonably accurate approximation of the quadtree working set data for each frame.

Another feedback approach, which I find more powerful and efficient in most use cases, is that taken by Crytek. In this approach, you render the geometry in texture space instead of screen space. The shader can fetch the z-buffer to determine if a particular texel or block of texels is visible from the current camera view. This approach is more straightforward to downsample as you render into a temporary target which matches the layout of your physical atlas, but is considerably smaller. The output can then be reduced to sum a direct

CONTINUED ON PAGE 61

CONTINUED FROM PAGE 59

count of the visible texels for each tile and how many pixels they map to. This approach also lends itself easily to partial updates and allows you to still compute screen/texel mapping for offscreen tiles, which is quite useful for predictive streaming.

WORKING SET MANAGEMENT

>> Given the feedback system which provides per-node visibility data from the renderer, we can devise a management system which decides when and which nodes to refine or coarsen. If we assume that the complete set of required tiles can always be found and loaded for each frame, the problem is simpler, and we just need a simple binary determination—a tile should either be resident or not. The list of desired tiles can then be compared to the list of currently resident tiles, unloading tiles which are no longer needed to make room for new desired tiles. In reality the quadtree is an approximation, memory is limited, and the system which generates texture data for new quadtree nodes (such as a disk loading process) will only have the ability to generate or load a fraction of the tiles each frame and will probably have many frames of latency. The simple binary solution with a LRU cache can work, but the pop-in can be an obvious and huge problem for typical cases of rapid camera motion, as described in previous implementations.

Dealing with latency and limited bandwidth for generating nodes moves the problem into the realm of optimization, and I have found a greedy heuristic algorithm to be effective. Each node is assigned a priority, and these are used to create two sorted lists: one for all the leaf nodes which can be expanded, and one for all internodes one level above the leaves which can be merged. These two lists are sorted in opposite order and traversed. Each pair of nodes is then examined for a potential swap. If the leaf node has a higher score than the internode, the internode is merged and the leaf node is expanded, reassigning the relevant texture tiles. Expanding a leaf node to create its four new children would typically involve requesting the relevant data from disk, or the system could procedurally generate the corresponding patches of texture. This incremental refinement continues until the relevant scores no longer result in a positive exchange (implying the structure is fully optimized for the current view) or for some fixed number of iterations (to fit into a hard per-frame budget). This simple incremental refinement system can implement a variety of management strategies depending on the choice of how priorities are assigned to nodes.

As a simple starting priority function you can assign nodes a constant maximum score, say 1.0, on frames in which they are visible and requested, with the score decaying over time on frames where they are not visible. This would emulate the earlier simple LRU scheme. A more accurate scoring function should consider the partial relevance of texel LOD used for mip-map filtering. This can be formulated as an error minimization problem. For even more efficiency, this scoring function should extend over a set of future frames, because we'd like to cache nodes over multiple frames to reduce bandwidth demand and anticipate the potential latency of generating new tiles. The ideal error function would estimate the error reduction that performing a swap (generating four new nodes at the expense of removing four others) will have on image quality integrated over some number of frames into the future, with the weighting decaying over the time period taken to refresh all the tiles. A node required for rendering the current frame is surely important, but if that node will only be used for a frame and then discarded, other nodes which aren't useful yet but will be useful for a number of frames in the future may be a better choice to start loading now.

This type of scoring model could be evaluated by predicting future camera motion and calculating tile visibility over multiple frames, but as this is an approximation it can suffice to just forward-predict camera motion a small number of frames into the future, update tile visibility only once, and then apply an exponential weighting over time. As general camera motion is unpredictable in a game, creating a perfect control system for streaming in a game is not fully possible, and its good to have some margin of error. Camera rotations are particularly unpredictable and rapid, so I've found it's good to assign a base level of node weighting just based on an estimation of texel coverage for offscreen surfaces. (In other words, assume there's a certain chance that the camera could rotate randomly.) This fits most naturally into the texture space update scheme, as it allows per-tile computations to be updated for both onscreen and offscreen tiles at once. The priority of a node can be estimated based on its total projected screen surface area with a bonus weighting for area visible onscreen in the near future.

As general camera motion is unpredictable in a game, creating a perfect control system for streaming in a game is not fully possible, and its good to have some margin of error.

CONCLUSION AND FUTURE POSSIBILITIES

>> Virtualized texture resources are practical on current console and PC hardware, and allow unprecedented artist freedom in surpassing some typical game resource limitations. The price for that freedom is some shader performance cost, a potentially even greater cost for real-time texture recompression or procedural texture generation, and a complex system implementation, although to be fair a practical high end streaming solution using the more standard mip substitution strategy is quite complex in itself.

A full virtual texture engine as used in id Tech 5 is also associated with a complex runtime system which decompresses texture data streamed from disk in a high compression format such as JPEG and then recompresses it in real-time into DXTC textures suitable for the GPU. This adds some rather significant CPU overhead but saves disk space and increases effective disk streaming rate versus storing and streaming DXTC textures natively. An interesting option along this line of thought is to use an advanced video compression codec, such as MPEG-4, which could be adapted to compress large texture surfaces at far higher efficiency than JPEG. This is possible because video compressors find and exploit spatial redundancies across macroblocks, and typical textures are highly redundant.

Looking further out, the same virtual texture techniques described here map well to three dimensions and thus have applicability for storing and streaming volume data. A voxel rendering engine would replace the quadtrees with octrees, and use smaller tile sizes, but could otherwise take advantage of the same streaming management techniques and B-tree indirection structures to accelerate spatial queries. Even now, current high end modelling programs allow artists to create enormously detailed models and worlds, games continue to grow in size and complexity, and compressing and managing vast rendering databases is a key challenge in game engine design of today and tomorrow.

JAKE CANNELL *most recently worked as a graphics programmer at Pandemic Studios, and has been creating graphics engines and demos at work and home for about ten years. He maintains a blog to speculate on such matters at www.EnterTheSingularity.com.*

resources

SIGGRAPH Chapter
http://ati.amd.com/developer/SIGGRAPH08/Chapter02-Mittring-Advanced_Virtual_Texture_Topics.pdf

Sean Barrett's GDC Talk
http://silverspaceship.com/src/svt

Offerpal MEDIA

Offerpal Media invites you to join us for lively discussions on

GDC 10 San Francisco

SOCIAL & MOBILE GAMING: What's Next?

PANELISTS*

HOST
George Garrick
CEO, Offerpal Media

MODERATOR
Michael Arrington
Editor, TechCrunch

MODERATOR
Jeremy Liew
Lightspeed VP

Trip Hawkins
CEO
Digital Chocolate

Vish Makhijani
COO
Zynga

Peter Relan
Chairman
Crowdstar

Christa Quarles
CFO
Playdom

Bart Decrem
CEO
Tapulous

Neil Young
CEO
ngmoco

Arthur Chow
COO
6 waves

Aaron Wong
Sr. Product Mgr.
Zynga Mobile

John Earner
GM
Playfish Europe

About Social & Mobile Gaming: What's Next?

The social gaming industry ballooned in 2009. But what comes next for the industry? Is the growth sustainable, or are we experiencing a bubble? If so, what can we do to keep it from popping? Where are the next growth opportunities? What will be the biggest risks and challenges? Join us as we debate the future of social gaming.

Networking Reception to Follow!

Visit the Offerpal Media Booth #2212

*scheduled to appear

GDC 2010

FRIDAY MARCH 12
3-5:30 PANEL
5:30-7:30 RECEPTION

MOSCONE CENTER SOUTH

SOUTH HALL
ROOM 310

THEME IS NOT MEANING

PART 2: PRACTICAL EXAMPLES

AS EXAMINED IN PART ONE OF THIS series, a game's meaning springs from its rules, and not necessarily from its theme, especially if the two are in conflict. Such a dissonance can leave players feeling lost, perhaps even cheated. Thus, designers should strive to keep the two in harmony. At the very least, they should not be at odds with each other.

ULTIMA IV.

When they are, the game's mechanics can actually undermine the theme that the designers want to deliver. For example, BIOSHOCK presents players with a true ethical choice—players can "harvest" Little Sisters in the game by destroying them or "rescue" them by releasing their minds. The reward for harvesting is double Adam (the game's genetic-modification currency), which tempts players to choose a morally disturbing path.

However, the game sprinkles other rewards on players who rescue Little Sisters, so the ultimate difference between the two paths is negligible from a statistical perspective. Players are told by the game's fiction that their choice matters—that they are making a sacrifice by deciding to rescue the little girls—but the game's mechanics tell them a different story. Of course, when theme and mechanics are in conflict, players know which of the two actually matters, and which is actually telling them what the game is about.

Similarly, many traditional RPGs put the player in an odd position. By giving the player an epic goal from the beginning ("Kill the evil wizard!"), the game casts him in the role of the world's savior. However, the actual gameplay involves roaming the countryside killing most of what falls in the player's path and looting everything else. The story tells the player that he is a hero, but the game rewards him for being something different. Richard Garriott directly attacked this dissonance when he designed ULTIMA IV, by making the game about achieving eight virtues instead of simply killing a "Foozle" at the end.

A PERFECT UNION

>> Sometimes a designer does achieve a perfect union of theme and mechanics. One example is Dani Bunten's SEVEN CITIES OF GOLD, the classic game of exploration. Bunten lost her way one day while hiking in the Ozarks and imagined a game in which the player struggles to keep her bearings in an unfamiliar landscape. From that seed, Bunten took the next step and chose a perfect theme—the age of the conquistadors, of Columbus, Cortez, and Pizarro, who were always partially lost—which provided wonderful raw background material with which to work.

Certain categories match theme and gameplay particularly well, including Wii games from Nintendo (WII SPORTS), music games (ROCK BAND), tycoon games (RAILROAD TYCOON), sports games (MADDEN), flight sims (WINGS), and racing games (GRAN TURISMO). Notice that while these examples are based on real-world activities, which helps keep the mechanics tied to the theme, a designer does not need to privilege verisimilitude above all else.

In fact, one could argue that MARIO KART is more truly about racing than GRAN TURISMO is—the former's rapid exchange of player position as shells fly around the track is perhaps closer to many players' ideal concept of racing than a stodgy simulation's more fixed positioning. Put another way, which object is more about Guernica—a photograph of the city's ruins or Picasso's masterpiece of anguish?

Further, great games can emerge when the theme simply provides an excuse to experiment with certain mechanics. LEFT 4 DEAD is not really a game about zombies, after all—it's a game about teamwork. The designers created each special zombie (those with special abilities, more powerful than the hoard) to encourage players to work together as a team—the Hunter punishes loners, the Tank requires concentrated fire, the Witch demands close player communication, and so on. The zombie theme simply gave the designers a plausible backdrop in which they could experiment with game mechanics that encouraged teamwork over solo play.

DOES CIVILIZATION FAIL?

>> The CIVILIZATION series provides an interesting study in the challenges inherent in trying to match theme with meaning. The games are purportedly about the sweep of world history, but one does not have to play long before cracks start to show.

To begin, societal progress is constant throughout the game—the player's civilization can never fall into a dark age or split apart in a civil war. The user community has dubbed this dynamic the "Eternal China Syndrome." The only entropy a player experiences comes from external invasion.

Indeed, the game actually provides a "Start a Revolution" button, so that the player can change government, but only when convenient. (I'm sure Louis XVI would have appreciated such a system!) Indeed, all actions in the game are conducted top-down—the player is some strange combination of king, general, tycoon, and god.

The source of these conflicts with real history is the problem of player agency. In order to be fun, the player needs to be in control. Moreover, the consequence of each decision needs to be fair and clear so that players can make informed choices, plan ahead, and understand their mistakes. Real history, of course, is much messier and more difficult to understand, let alone control.

In fact, the games mechanics of CIVILIZATION tell us less about world history than they do about what it would be like to be part of a league of ancient gods, who pit their subjects against each other for fun. These immortal opponents, after all, are the only characters that can destroy the player. The people themselves have little say in how history will develop.

But of course, player agency is actually a good thing; indeed, it is at the very center of what makes games so powerful. Perhaps some topics are simply too broad or vague or slippery to be addressed by a game's mechanics. And sometimes, themes can just be themes, with the player knowingly entering a fantasy space that speaks not directly to the topic but to some other need or desire.

In the case of CIVILIZATION, the desire is to control history, which may not teach us much about it, but

ILLUSTRATION BY JUAN RAMIREZ

it is not without value. Indeed, the game fares well when compared with other artistic disciplines. Few works of art tackle the sweep of world history, and the ones that exist (*Birth of a Nation*, for example) are often dangerous works of ideology.

Designers who care to make games that actually speak to us about history should focus on a specific era or event, such as Bunten's SEVEN CITIES OF GOLD, or SID MEIER'S RAILROAD TYCOON. Put the player in the shoes of a flesh-and-blood person. Let her explore the challenges and opportunities of the times but within mortal limits.

WHY THEME MATTERS

>> Although a game's theme and mechanics can tell different stories, society at large does not understand that there is a difference between the two, and if the theme is appalling to the mainstream, a good game can be unfairly tarred. For example, GRAND THEFT AUTO has a theme of crime and urban chaos, but the game is actually about freedom and consequence. Every crime increases the player's notoriety, which can end the game if the police send enough firepower.

Nonetheless, to the mainstream, GTA was simply about killing hookers and running over pedestrians—for outsiders, the game couldn't be "about" anything else. Players, however, understood that the game was giving them something different—an open-world in which their decisions actually mattered. Consequence was the true killer feature.

CRACKDOWN provides an interesting contrast in that it delivers the same open-world simulation with consequence as GTA but with a theme (fighting crime as a super-cop) much more palatable to the average person. Rockstar may have record sales to show for their work, but designers who believe they have a responsibility to society at large should take note that the criminal theme was not inevitable.

Today, many designers strive to achieve two worthy goals—reach a mass audience and create great art. However, both are at risk if theme and mechanics are in dissonance. The average consumer, who is not highly literate in the standard tropes of game design, expects video games to be about whatever is on the cover. Pulling a bait-and-switch—or simply not thinking critically about the lessons that a game actually teaches—will only turn new players away.

As for the question of art, one must first recognize that many great works of art are abstract. Lyrics may give some meaning to a song, but a symphony is generally meant to be interpreted and enjoyed however the listener prefers. Similarly, games can stand on their own without specific themes—TETRIS being the obvious example.

Furthermore, even a pasted-on theme can work if the designers are not promising more than the game can deliver—*San Juan* and *Race for the Galaxy* are both brilliant, yet similar, card-based adaptations of *Puerto Rico*. That one is set in the Caribbean and the other in outer space is not a problem, since the games are clearly not marketed as re-creations or simulations. The theme simply adds flavor.

However, great art never has theme and meaning in open conflict, in the way many games do. *Othello* is actually about the "green-eyed monster" of jealousy and not just the life of a Moor in the 16th-century Venetian military, but the latter does not detract from the former. Can the same be said about BIOSHOCK? About SPORE? About CIVILIZATION? These games do claim to be about something—do their mechanics tell the same story? To touch people, the play itself needs to deliver on the theme's promise.

SOREN JOHNSON *is a designer/programmer at EA, working on browser-based strategy games at www.strategystation.com. He was the lead designer of CIVILIZATION IV and the co-designer of CIVILIZATION III. Read more of his thoughts on game design at www.designer-notes.com.*

INFERNAL ENGINE®

www.infernalengine.com

The Infernal Engine, from Terminal Reality, is the most cross-platform foundation for building video games. Designed to solve today's real game development challenges, Infernal is a unified system, providing superior rendering, physics, sound, and AI in a single package. Built by the experienced engineers at Terminal Reality to optimize the game creation process and to take advantage of today's complex hardware – the Infernal Engine is the best choice for game developers.

Xbox 360 •PLAYSTATION 3 •PC •Wii • PLAYSTATION 2 • PSP

Industry Leading VELOCITY® Physics Engine

- Advanced collision system maintains thousands of simulating objects on next-gen hardware
- Advanced dynamic destruction for scenery & environmental objects

Next-Generation Rendering

- Multi-threaded
- Global illumination with real-time soft shadows
- Programmable material system

Cross Platform

- Full featured editor, with console emulation
- Scalable materials and dynamic lighting
- Configurable, platform specific texture settings

Contact us for an evaluation version or more information: enginesales@terminalreality.com

PLANTING THE SEED

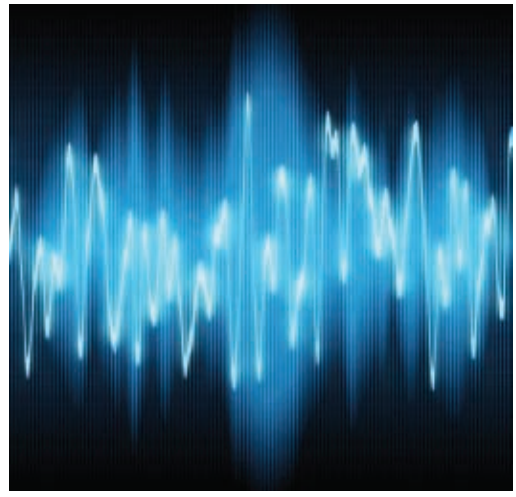
RETURNING TO SYNTHESIS WITH AUDIOKINETIC'S SOUNDSEED

THE DISCIPLINE OF GAME audio was born from synthesis. Chips generated synthetic waveforms, and the early pioneers of game audio shaped those waveforms into representations of every asset imaginable. From explosions to human speech, the limitations of synthesis-only sound design meant that most game audio was only an approximation of real-word sound, often iconic in its abstraction but ultimately unfulfilling in its aesthetic impact. The moment technology gave us the ability to create, capture, and play back real-world (or off-world, as the case may be) sounds in the form of digital audio files, synthesis fell by the wayside for all but the most limited game platforms.

Unfortunately, as soon as we abandoned synthetic sound for digital sound, we ran into another challenge: the ever-growing memory footprint of digital audio files. The pipe dream is realistic-sounding audio that doesn't take up memory or disc space, a solution that actually points backward, toward a rediscovery of the use of synthesis.

SOMETHING IN THE AIR

>> As recently as two years ago, real-time creation of convincingly realistic audio assets via synthesis was purely academic. Today, however, it's not only possible, it's also commercially available, as part of Audiokinetic's



Wwise toolset under the name SoundSeed. SoundSeed is a suite of two different plug-in modules that use synthesis to tackle real-time manipulation of synthetic assets.

The first module is called SoundSeed Air. The SoundSeed Air bundle is broken down into two different tools: Wind and Woosh. Wind, as the name implies, is used to synthesize environmental wind effects. Wind works by placing "wind deflectors" into a scene. These deflectors then react to synthetic wind (white noise) which is "blown" across them. Working in consort with the deflectors, the user has control over a host of real-time parameter controls that affect how the wind sounds. These controls are mostly a single-frequency resonant EQ. By controlling both the Q and the frequency of the wind, you have control over how tonal the wind sounds. More Q equals more tonality. Less Q equals more white noise.

Multiple deflectors can be added to a scene, and each one has its own parameter controls and user-definable positioning. Users have control over variables such as wind speed, wind direction, gustiness, volume, and variability. The latter is a parameter that works with gustiness in order to create more realistic behavior for the wind. All these parameters can have user-defined automation curves and contain randomization sliders.

The result is a dynamic wind environment without the need for long looping files that need to be streamed off the disc. In-game parameters such as changes in elevation or room size can be easily accommodated without the need for elaborate crossfade mechanisms between digital audio files.

The second part of Air, Woosh, also works by authoring deflectors. Unlike Wind, though, Woosh's deflectors aren't rooted to a specific location within the scene. Instead, Woosh's deflectors generate sound as they move through the air attached to objects, creating such sounds as bullet pass-bys, punches, or helicopter rotors. Just like Wind, the user has control over the noise's resonant center frequency as well as Q, and has access to multiple kinds of noise, including white, brown, and pink. The main parameters for Woosh are object speed, trajectory, and time. Object path defines distance-based volume attenuation. Just as with Wind, the user has control over definable automation curves and randomization sliders.

THE HIT PARADE

>> The second module in the SoundSeed bundle is called Impact. As the name implies, Impact is concerned with hit and collision sounds, such as footsteps or weapon clangs. The main focus is on resonant impacts, or impacts with some amount of tonality to them, and providing a smart solution to variability and variety while keeping the resident memory hit small.

Within Impact, each impact sound has two components: the initial sound made as soon as an impact occurs and the vibrations that occur after the initial point of contact. Impact takes a source file, analyzes it using its offline impact modeler, renders it into the initial impact sound and a text file describing the resonant content, and then re-synthesizes the resonant content in real time. This effectively shrinks the needed file size down to just the initial collision while simultaneously allowing for an increased number of variations without affecting resident memory. Within the Impact plug-in, the user then has control over magnitude, frequency, and bandwidth in order to shape and randomize their variations of the resonant content of the sound.

MORE TO COME

>> Undoubtedly, Audiokinetic's SoundSeed is merely the beginning of a new class of tools and assets. As the drive to increase variation while keeping resident memory requirements low persists, synthesis-based asset creation is going to be invaluable for elaborate physics systems, rich combat experiences, and detailed real-world simulations. By stepping backward and rediscovering synthesis, sound designers are reacquainting themselves with one of the oldest and most useful tools in the game audio arsenal.

JESSE HARLIN *has been composing music for games since 1999. He is currently the staff composer for LucasArts. You can email him at* ***jharlin@gdmag.com.***


MAKE AN IMPACT.
join our TEAM.
SLEDGEHAMMER
GAMES
Sledgehammer Games @ GDC March 11th-13th, 2010 : Activision Booth #2526 South Hall
www.sledgehammergames.com

ZOMBIE APOCALYPSE

IS THE JOB MARKET ON LIFE SUPPORT?

AN UNREAD POST-IT FLAPS LISTLESSLY ON an empty cubicle, a pathetic hint of color in a deserted wasteland. The scavengers have picked everything clean trying to stay alive. Outside: chaos. Devastation. Despair. A twilight struggle for survival between the living and the restless not-quite-dead.

Yup, it's tough times on the job market.

In 2009, mainline game sales in the U.S. contracted by about 8 percent, from $22.1 billion to $20.2 billion according to the year-end numbers from NPD. The NPD numbers don't count online sales such as Steam or the Apple App Store, so they may overstate the decline—but for an industry that roughly tripled in size in the last decade, even a little contraction can be as scary as anything George Romero ever filmed.

Shrinking sales and a gloomy economic background translate inevitably to tough times for us in the trenches. Last year brought the closure of many studios, including industry fixtures like 3D Realms and Pandemic. Research firm M2 estimates there were over 8,000 layoffs in U.S.-based game companies since the downturn started in 2008, with a further 4,000 or so around the world. The big layoffs—like those at EA in November—grab all the headlines, but even those who haven't lost their jobs can feel the pinch as projects are canceled, expansion plans shelved, and raises deferred. There's even a new "spouse" scandal, alleging overwork and underpay at Rockstar San Diego. There's not much fun to be had in the fun business these days. We were supposed to be "recession proof," dammit!

We've survived slowdowns before, and always came back as a bigger and more vibrant industry. Year-on-year sales fell in 2005 too, but the industry boomed once consumers warmed up to the new console generation. Still, with grim news all around, it's hard not to wonder: will this time be different? Are the boom times over?

~~28~~ 60 MONTHS LATER

» Our business has changed a lot over these last five years. The current console generation has inflated team sizes and budgets for traditional games enormously since the last downturn. A fully ZBrushed, normal-cast, ragdolled, shadered-up modern asset might demand five or even ten times as much work as a comparable asset from the PlayStation 2 days. Cheap storage space and the allure of open-world games have also driven content creep. It was once possible to build a triple-A title with a team of 30 or 40 people, but we now see teams of 300 or more. At the end of the last console cycle you could build a triple-A game for under $10 million; now they run close to $50 million. This is not a crazy, seat-of-the-pants startup business anymore. We've gone industrial scale.

Big teams and big money have changed the way we work. We all bitch about sequel-itis and me-too franchises, but we make sequels for the same reason Hollywood does: when you're on the hook for tens of millions of other people's dollars, you love safe bets. But more than the content has changed. Our huge teams generate more bureaucracy, more middle management, and more politics. The sense of personal ownership that used to distinguish games from film

Illustration by Mateja Petkovic.

BLIZZARD IS HIRING

We are actively recruiting across all disciplines for the following locations:

IRVINE, CALIFORNIA | AUSTIN, TEXAS | VELIZY, FRANCE | CORK, IRELAND

SEOUL, SOUTH KOREA | SHANGHAI, CHINA | TAIPEI, TAIWAN

jobs.blizzard.com

Visit us in the GDC Career Pavilion at Booth #2726.

production or conventional software development has eroded at least in the triple-A space.

The flip side of this new way of working—more hierarchical, more impersonal, and more expensive—is outsourcing. When you don't know three quarters of the people in the building, emailing that concept sketch to Budapest rather than down the hall to the modeling pod doesn't seem too bizarre. When the assets for GEARS OF GRAND THEFT TURISMO WARS IV total 450 years (!) of artist time, you can bet that somebody back in the accounting office is Googling the hourly rate for 3D modelers in Guangzhou.

LEFT 4 DEAD

>> Outsourcing isn't going to go away. It's a permanent fixture of the way games are made. It's important to be clear about this: outsourcing is more than just a cheap way to supply MMOs with tables and chairs. More than three quarters of studios surveyed last year have used some form of outsourcing, and that number is expected to rise to more than nine out of ten in the near future. Outsourcers around the world are gaining experience both in graphics tech and in dealing with international clients. They're smart, talented, and they cost a lot less than we do. Like it or not, outsourcing is part of the business.

It's easy to imagine the nightmare scenario: A Dickensian economy, development bloat, and low-cost competition push more and more production work to low wage countries. The artists that hang on in North America, western Europe, and Japan stratify: a handful of the lucky ones still do art-direction tasks, outsourcing management, or highly technical jobs like shader authoring and character setup, jobs which demand very close cooperation with engineering and design. The rest of the survivors get stuck with cleanup work. When the modeler in Latvia or the animator in Mumbai sends back a file, somebody on-site still has to vet it, add engine-specific markup, and make sure it hits technical specs.

It's hard to tell what's worse: downward pressure on wages and benefits, or watching what makes our jobs fun get shipped overseas along with the money. What do you prefer: bragging to your real-world friends about making ogres, robots, and smartass talking rodents for a living—or telling them how many texture sheets you resized to powers-of-two, or how many naming conventions you enforced today? Nightmare scenario, indeed.

Fortunately, nightmares don't always come true.

So far (knock on wood) the rise of outsourcing has not led to massive job losses. Studios continued to expand at home even as they turn to outsourcers for temporary extra capacity, rather than simply "exporting jobs" like textile mills or furniture factories do. Good talent is still hard to find, and it still commands a price.

Moreover, game studios have always done a bad job of keeping people busy. Most of us have polished our MINESWEEPER skills or Photoshopped LOLcats while waiting for the last game to pass cert or the next to emerge from pre-production. As pleasant as that may be for us, it's torture for our bosses. Outsourcing has often been a humane way for companies to respect the goat-in-the-python life cycle of production without a perpetual cycle of layoffs.

RE-ANIMATOR?

>> For an example of what an outsourcing-heavy future might look like, you can look at the film business. The so-called "Hollywood model" of game production has been fodder for innumerable GDC panels and articles in these pages over the years. Stripped of posturing, the idea is pretty simple: rather than maintaining a full-time infrastructure of production staff, a "studio" would consist of only a small core creative team that handles design, concepting, and management. Production muscle—i.e. the line art staff—gets hired on an as-needed basis, so there's no need to make work for hundreds of idle hands between crunches. Rigorous pre-production makes sure that every moment of the production run is tuned to maximum efficiency. The actual production hands are freelancers or outsourcers, handsomely compensated for the brief time they're working on the assumption that they'll have some downtime between projects.

The advent of this system has long been predicted, but so far the revolution has been postponed. However, the current industry squeeze adds urgency to those old GDC debates. Hollywood-model advocate Alex Seropian used to claim that distributed production could cut 35 percent off the cost of a title. In today's economy, that's a pretty attractive idea. Perhaps a different production system can breathe some new life into the business. What would life look like if we do, at long last, go Hollywood?

For a start, a lot more of us would become freelancers. Many of the rest would work for dedicated production firms rather than game studios. The ecosystem of production facilities would be very complex, including both general-purpose dev houses like Liquid Development and specialty firms devoted to very specific niches. For example, if you're a ragdoll specialist you might end up working for a small company of ragdoll ninjas who do setup and debugging for many different games at a time.

This scenario has both good and bad aspects for the average artist. The freelancer lifestyle typically alternates between periods of intense activity and long downtimes. The cash rewards are often quite high—but it takes a lot of discipline to plan for the fallow periods. It's great for young single types, but can be stressful for families or the naturally nervous; it might exacerbate our industry's existing tendency to alienate older artists. On the plus side, a proliferation of small and midsize outsourcing groups offers lots of entrepreneurial openings to ambitious artists—who wouldn't want to be a founding partner of an elite art-production house like Massive Black?

Of course, games aren't exactly like Hollywood. Whether we end up following in Hollywood's foot steps will depend in large part on how we address a few of the important differences between the operating practices of the film world and the game industry.

The first critical issue is geography. The sheer concentration of specialty resources in Los Angeles is essential to the way Hollywood works. Studios looking for production resources and production people looking for jobs are all part of the same social networks and economic environment. While the crew for each new production is unique, it always has a high percentage of familiar faces. This cuts the time needed to turn a cloud of freelances into a coordinated team; it also puts a damper on the export of work to lower-wage markets.

If we go Hollywood, expect to see more geographical concentration. It's impractical to build a career as a freelancer in a market that's too small for steady work, or to staff up quickly in a remote location. Even with Google Wave and a T-1, it's hard to build much of a business for yourself without a reliable network of social and business connections. Any successful freelancer will tell you that you get a lot more work from "friend-of-a-friend" tips than cold calls and web postings. If the Hollywood model takes over, it could be bad news for isolated outposts far from the centers of power.

The other thing Hollywood has that we don't is unions. Love 'em or hate 'em, they're a critical part of the landscape because they provide important cushioning for the ups and downs of the feast-and-famine cycle. Whether it's offering

The freelancer lifestyle typically alternates between periods of intense activity and long downtimes. The cash rewards are often quite high—but it takes a lot of discipline to plan for the fallow periods. It's great for young single types, but can be stressful for families or the naturally nervous.

health insurance or negotiating standard contract terms, unions are part of the grease for the Hollywood production machine.

If games go Hollywood we'll need something that plays the same role, whether we call it a "union" or something else. A standardized freelancing system might look more like the Graphic Artists Guild than the United Auto Workers, but it will still be an important mechanism for tamping down the volatility of the freelancer lifestyle. The critical question, of course, is how many concessions the labor—that's us—can extract without chasing too many jobs overseas. Maybe it's time to start trolling those IGDA forums in earnest.

Another important way unions make the Hollywood model work is by encouraging standardization. In L.A., if you hire a gaffer or a key grip you have a pretty good idea of what you're getting—even if nobody outside the film business knows why they get such odd job titles. In our business, on the other hand, roles and titles are fluid to the point of frustration. If we embrace a film-style production model, don't be surprised if standardized titles (and pay scales) emerge quickly. It might be startling to be reclassified, but HR managers all over the games world will heave a collective sigh of relief it this ever comes to pass.

DEAD RISING

» Of course, changes in the business model are not the only forces reshaping the foundation of the game business. While the modus operandi of "core game" development has been super-sizing, games as a whole have become a much more interesting and varied enterprise. From iPhones to Flash to Facebook to training cops and firefighters, games are now a fixture in settings undreamed-of in the days of the Dreamcast. This flourishing ecosystem of platforms, audiences, and businesses fosters a host of interesting new niches and alternative career paths. This new frontier offers at least a hint of reassurance to anxious artists who worry about the cost of keyframes in Calcutta or polygons from Prague. There are a lot of new opportunities out there—even in these grim economic times.

Unfortunately, the dynamism that makes it possible for an unknown group of amateurs to ship a million-selling iPhone app also makes it difficult to build and maintain a traditional studio environment. Life on the frontier will be more exciting but also riskier than our familiar studio system. A sector can flourish without a lot of ordinary devs benefitting: if one artist in ten makes a million on his or her iPhone app, and the other nine go broke, the average is a respectable hundred grand—but it's hardly a great job market. Nevertheless, if you need a reason to not ditch games and go to dental school, there's still excitement and adventure to be had on the new frontiers of gaming. It's a good time to think about re-examining your triple-A prejudices.

These are "interesting times," in the words of the old Chinese curse. There's no infallible way to predict what's going to happen, because the rapid fragmentation of the core business means many often- contradictory changes may happen simultaneously. The one thing that's safe to say is that the studio system we've known is in flux. Handhelds, web games, and digital distribution are reshaping the business landscape. Indie, casual, and serious games are changing the demographics of the audience. Equally radical changes in how we make games are inevitable.

It's the end of the world as we've known it. Stock up on supplies and learn how to survive.

STEVE THEODORE *has been pushing pixels for more than a dozen years. His credits include* Mech Commander, Half-Life, Team Fortress, Counter-Strike, *and* Halo 3. *He's been a modeler, animator, and technical artist, as well as a frequent speaker at industry conferences. He's currently a consultant helping game studios perfect their art tools and pipelines.*

GDC 10

Career Pavilion

Thursday, March 11 10:00am–6:00 pm
Friday, March 12 10:00am–6:00 pm
Saturday, March 13 10:00am–6:00 pm

Located in Aisles 2300–2799
(on the right side of the Expo Floor)

Stop by the Disney Career Pavilion Mixer

Thursday, March 11 4:00–6:00 pm
Enjoy free drinks, appetizers and a chance to win an All Access Pass to the GDC of your choice!

EXHIBITORS

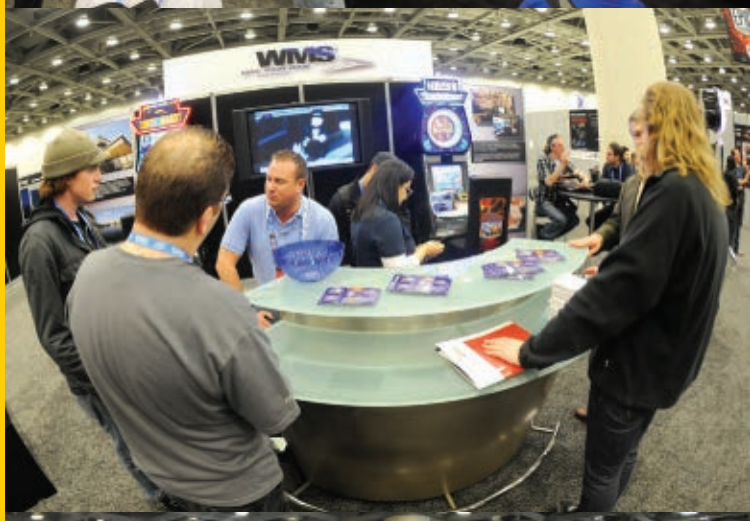
- Activision
- Bally Technologies
- Bigpoint Inc.
- Blizzard Entertainment
- Blue Castle Games
- CCP Games
- Contact Singapore
- Crytek Gmbh
- DeVry University
- Enjmin
- Flashpoint Academy
- Foundation 9 Entertainment
- Gameloft
- Gazillion Entertainment
- Gnomon School of Visual Effects
- High Voltage Software
- Insomniac Games
- International Game Technology
- Konami Digital Entertainment Co.
- Krome Studios
- LucasArts
- Masters of Digital Media
- Microsoft
- Obsidian Entertainment
- Playdom
- Premier-Search Inc.
- Realtime Worlds
- Shanghai F&Q Games Co., Ltd.
- Shanghai Hongli Digital Technology Co., Ltd
- Sony Computer Entertainment America
- TalentScotland
- Telltale Games
- Tencent Technology Company
- The Art Institutes
- The Guildhall at SMU
- THQ, Inc.
- Ubisoft
- WB Games
- Westwood University
- WMS
- Zenimax Media
- 38 Studios

SPONSORS

SONY

Disney Interactive Media Group

WESTWOOD COLLEGE



GOOD JOB!

Hired someone interesting? Let us know at editors@gdmag.com!

be prepared!

SCOUTING OUT THE GDC CAREER PAVILION AS A NEWER DEVELOPER

BY MATHEW KUMAR

2009 HAS BEEN TOUGH FOR THE games industry, with layoffs, consolidations and reorganizations meaning that not only have talented staff found themselves without jobs, but fresh graduates have found themselves entering into an uncertain future. The GDC Career Pavilion (Thursday, March 11th—Saturday March 13th in Moscone South Hall, accessible with all GDC passes, including Expo or Student passes) presents an opportunity to receive face time with recruiting studios and publishers, and with only three days to make an impression it's important to not waste any time. We've talked to some of the top recruiters from companies including Blizzard, Ubisoft, and Sony to ask them what they're looking for, and with that knowledge at hand you can ensure that every impression you make can be positive.

GDC PHOTOS BY VINCENT DIAMANTE

REALLY! BE PREPARED!

>> Not only the motto of the Boy Scouts, "Be prepared!" is also overwhelmingly the advice of every recruiter we talked to. Before even beginning to research the Career Pavilion, recruiters strongly advised that all job seekers have a specific position in mind, and organize their preparation toward getting it.

"Too many times folks come to a booth and say they want [work as] an artist, programmer, or a designer," said Maggie Bohlen for High Voltage Software (a multiplatform developer that recently developed THE CONDUIT for Sega). "Don't expect to leave your choice of career in our hands—know where your strengths lie and focus on that specific direction."

Once you know what you're looking for, tailor not only your resume to the position, but your portfolio (consisting of art, code samples, audio work, or whatever is most relevant to the discipline) and remember that even within disciplines it helps to be as specific as possible.

"All too often I'll have an artist walk up and say, 'I'm a concept, environment, and character artist, I can do it all!'," said Kraig Docherty for Blue Castle Games (currently working on DEAD RISING 2 for Capcom.) "This will not help your case."

Once you've got your tailored resume and portfolio, strip it down. Ensure your resume is as clear and as short as possible (recruiters advise it must always fit on two sides of one sheet of paper) with no spelling mistakes, typos, or formatting mistakes (every single recruiter told us this very important). Whittle down your portfolio to only the absolute peak of your output—your best work and the work you are most passionate about, and include descriptions. Then, though this may seem awkward, make sure that you are prepared to offer both your resume and portfolio to recruiters in a format that suits them—because if there's one thing our recruiters couldn't agree on, it was their preferred method to receive business cards, resumes, and portfolios.

All recruiters informed us they appreciated having a paper resume and portfolio to check out during any scheduled interviews, but business cards were considered unnecessary unless you are applying for a position of seniority ("They just get lost," confided an anonymous recruiter) and if you are unable to schedule an interview, recruiters were most positive about receiving a traditionally laid-out (8.5" x 11" paper) resume that included a link to a personal website that features your (tailored) portfolio ("put the link at the top of your resume in bold," begged one.)

However, a few recruiters stated they would happiest to receive CDs which contained both a resume and portfolio—as long as they were clearly labeled.

"If you're an artist, take two or three screenshots from your portfolio along with your resume verbiage and display them inside the clear cover of the CD/DVD case. It's very handy for instant recognition," added a recruiter. Whatever you do—ensure all the information a company needs about you is provided to them in one single submission.

RESEARCH IN DEVELOPMENT

>> So you know what position you want, and you've tailored your resume and portfolio to the position, with plenty of paper copies of your resume, a nice portfolio website, and a few CDs for those

ACTIVISION®

ARE YOU READY TO EXPLORE THE MOST SOUGHT AFTER JOBS IN ENTERTAINMENT?

VISIT US!
MOSCONE CAREER PAVILION AT GDC
BOOTH #2526

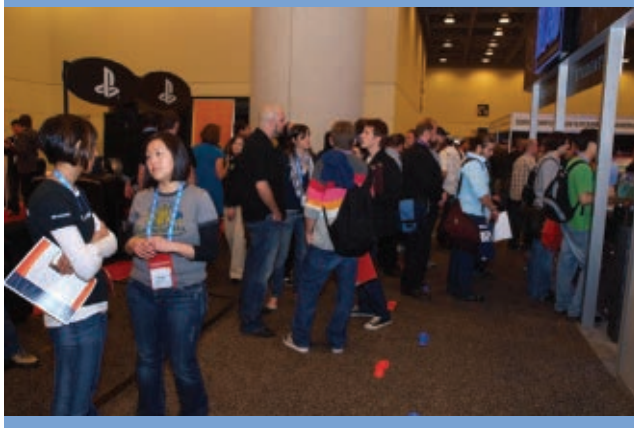
MARCH 11TH - 13TH

Activision Minneapolis · Bizarre Creations · Raven · Beenox · Underground · Vicarious Visions · Radical Entertainment · 7 · FreeStyleGames · Guitar Hero · treyarch · Luxoflux · Sledgehammer · Neversoft · Infinity Ward · Toys for Bob · DemonWare · High Moon · Budcat

Visit our site: www.activision.com

who might want them. You're ready to walk the Pavilion, right?

No matter how well you might know your own needs by this point, recruiters tell us they can spot someone a mile off that doesn't recognize their company's needs. Using GDC's published list of companies who will be represented at the Career Pavilion, research each company to find out



if they would be right for you—and you for them.

"Know the companies you are going to approach and do some research," said Blizzard Entertainment's Sumer Ortiz. "Know where they are located, the positions they have posted online, and their company culture. If you are armed with the basics, you can spend quality time talking to developers and recruiters about what it takes to get a job."

All recruiters surveyed agreed with this—"it's really as easy as checking out our website," said one—Ubisoft's Stéphanie Franco expanding that prospective employees should "Research recent job postings and the corporate website (including key titles)" using this information to prepare a "30 second pitch outlining what experience and skills you offer in relation to their current hiring needs. Be prepared to explain what special skills you can offer to their team."

A lack of research into a company was considered the most egregious error that an applicant could make, one that would absolutely destroy any potential for a good first impression (and woe betide if you've also managed to make spelling mistakes on your resume). Not only that, it pays to consider the recruiter's time just as valuable as yours.

"It's very important that job-seekers focus on companies that are located in areas in which they wish to live," said Bohlen (High Voltage). "It's deflating when you spend time with a candidate and then they ask you at the end of the conversation where the office is located, and when they realize we are in Chicagoland they shiver and say 'No thank you!'"

PRESENT YOURSELF PROFESSIONALLY

» With portfolio and resume in hand and an encyclopaedic knowledge of the companies you have focused on, the final thing to ensure before stepping into the Career Pavilion—or anywhere within San Francisco county during GDC week—is that in all cases you present yourself professionally.

"Never forget that you're interviewing—even if it's just a two minute conversation," said Sony's director of talent acquisition Karen Chellini. "You must always be ready to engage with a prospective employer."

While the game industry is generally casual, and during GDC week can get a little raucous, job-seekers have to hold themselves up to a slightly higher standard, and consider themselves to be under scrutiny even at the wildest GDC party—after all, you never know who you're going to bump into, and at what time.

Don't overcompensate, though. Game recruiters surveyed valued looking neat, clean, and presentable in smart casual clothes over wearing a suit—if you're not natural in a suit, or it's ill-fitting, it comes across as recognizably false.

First impressions were also strongly led by how confident, concise, and friendly applicants were according to recruiters. While that may seem less obvious to job seekers looking for work such as programming where you might imagine your communication skills are less valued, recruiters argued that video games are created by teams, not individuals. Remember to smile and look people in the eye, and if you're shy, practice what you think you want to say to potential employers (but not what you think they want to hear—be honest).

"Practice interviews with a friend beforehand," advised Renee Scott at Gazillion Entertainment (an MMO developer working on the upcoming LEGO UNIVERSE). "Try to eliminate all the 'ums' and 'uhs' and have answers for harder questions like 'why did you leave your last job?'"

But if you're a comfortable communicator, be wary of letting your mouth run off. "Treat any discussion like a professional interview," continued Scott. "Be respectful, don't swear or trash talk, and while being chatty is fine, remember that everyone at GDC is on a schedule and has more to do there than there is time for."

DON'T PANIC—NETWORK

» Recruiters will be there for three days—you don't have to talk to everyone on the first day. Taking your time to scout out the pavilion and talk to people at booths that may not have been your first choice during your initial research period can be invaluable, and at those booths you were interested in, there is often non-recruiter company staff available to answer more in-depth questions before you make a serious attempt to apply for a job. Remember—you don't want to waste anyone's time, especially not your own, so take the time to make sure a company is right for you, and perhaps you'll find out new things while networking.

Whether you're new to the industry or a well-established veteran, networking is important—try and hook up with people in the industry you know and people you want to know for casual discussions that don't need to relate to getting a job—a good impression in a chat about a GDC seminar you both attended is still a good impression. Don't be shy to talk to companies or individuals that you respect or admire—just don't fawn or ramble!

THE MORNING AFTER

» Once GDC is over, even if you've had a good experience at the Career Pavilion with plenty of new connections via savvy networking, lots of resumes distributed and maybe a few interviews, don't expect to be able to sit back and wait for the job offers to roll in. With thousands of others applicants in the same market, "relying on a resume that may have been buried under 200 others in a pile can mean never getting a call back," said Gazillion's Scott. So be sure that if you were asked to fill out an online application, you do, and if you feel you made a connection with a recruiter or another GDC attendee (and were given their contact details) send them a polite follow-up e-mail to solidify that connection. Preparing for the Career Pavilion is all about giving a great first impression, but it's equally important that you keep it up—that way you'll be sure to stand out and get the job you worked so hard for.

MATHEW KUMAR *is a freelance journalist based Toronto and a contributing editor at Gamasutra.com. Email him at* ***mkumar@gdmag.com****.*

New IP
New Challenges
New Production Values
all require a

New Attitude.

Are you right for the
New THQ?

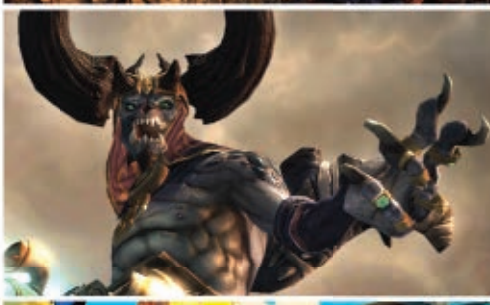
Darksiders. Red Faction. deBlob. Digital.
We're just getting started.

Curious?

Come see us at GDC.

THQ®







IF YOU'RE LOOKING TO WORK ON THE NEXT BIG THING

WE'RE HIRING.

id Software is now hiring for:

Artists
Animators
Designers
Programmers
Producers
Scripters
Marketing/PR
QA
IT

STOP BY TO SEE US AT GDC!
March 11th – 13th
Booth 2316

E-mail us your resume and portfolio if you're interested in setting up an on-site interview with one of our studios at GDC.
gdc2010@zenimax.com

www.idsoftware.com/makedoom

38 STUDIOS

A FAMILY OF PEOPLE

CREATING A FAMILY OF PRODUCTS

NOW HIRING

LEARN MORE AT 38STUDIOS.COM

TODD MCFARLANE

R. A. SALVATORE

KEN ROLSTON

CURT SCHILLING

MEET WITH US IN THE CAREER PAVILION AT GDC (BOOTH #2438)
OR SUBMIT YOUR APPLICATION TO JOBS@38STUDIOS.COM

38 STUDIOS

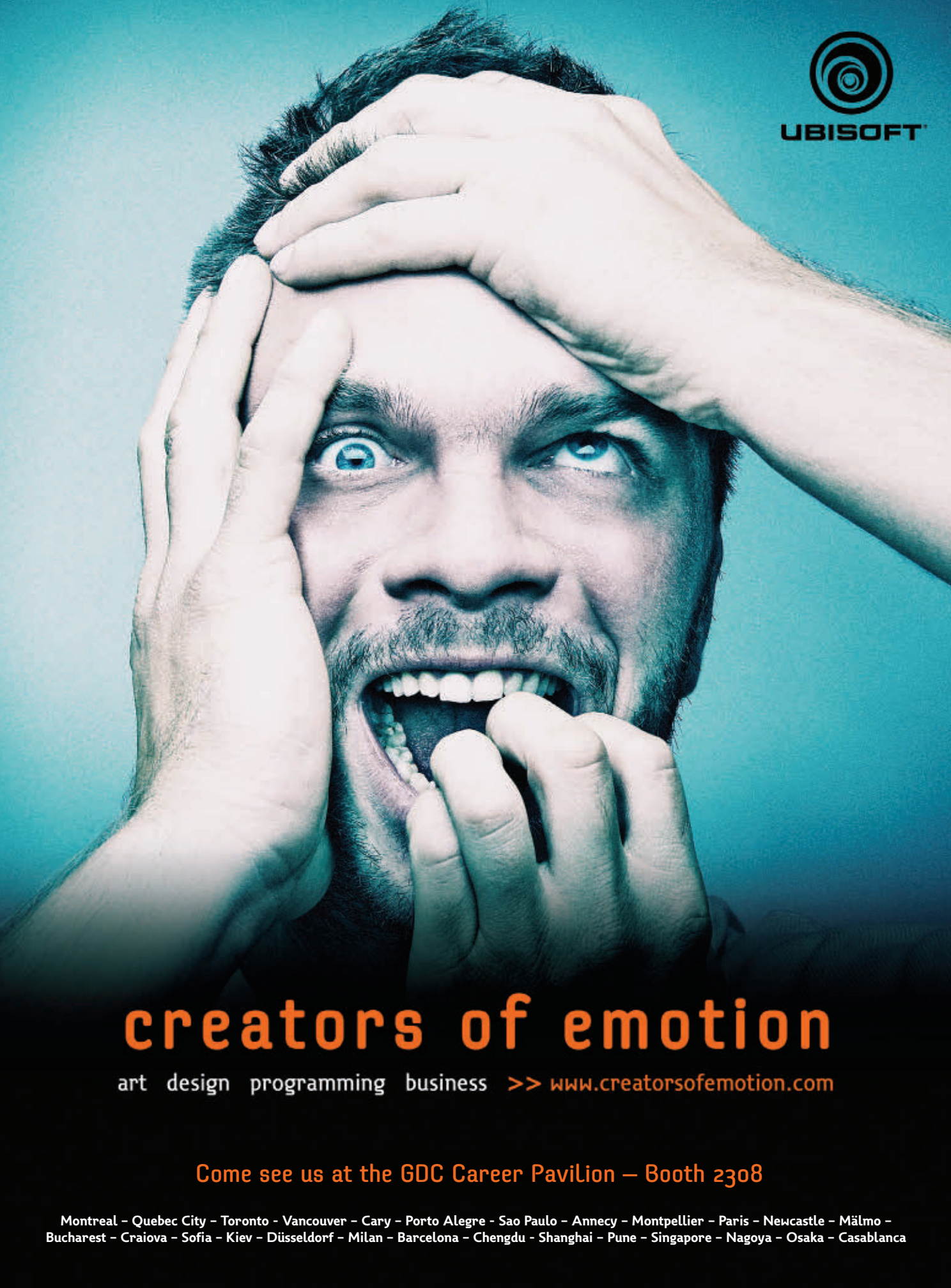
BIG HUGE GAMES

Apply today

Now hiring at all locations

Greater Seattle/ Greater Chicago

wbgamesjobs.com


UBISOFT
creators of emotion
art design programming business >> www.creatorsofemotion.com
Come see us at the GDC Career Pavilion – Booth 2308
Montreal – Quebec City – Toronto - Vancouver – Cary – Porto Alegre - Sao Paulo – Annecy – Montpellier – Paris – Newcastle – Mälmo – Bucharest – Craiova – Sofia – Kiev – Düsseldorf – Milan – Barcelona – Chengdu - Shanghai – Pune – Singapore – Nagoya – Osaka – Casablanca

FROM THE CREATORS OF **"SCRIBBLENAUTS"**, **"DRAWN TO LIFE"** & **"LOCK'S QUEST"**

WORK ON SOMETHING

TRULY ORIGINAL

Now hiring for Xbox 360:

- **Lead Environment Artist**
- **Senior Environment Artists**
- **Lead 3D Level Designer**
- **Senior 3D Level Designers**
- **Lead Animator**

5THCELL™

Visit 5thcell.com/jobs

VISIT US AT:

HTTP://JOBS.LUCASFILM.COM

OR VISIT US IN:

SAN FRANCISCO AT GDC 2010
IN THE CAREER PAVILLION

LUCASARTS

THE 10TH ANNUAL

game developers CHOICE awards

The Game Developers Choice Awards are the premier accolades for peer-recognition in the digital games industry. Every year at GDC, the Choice Awards recognize and celebrate the creativity, artistry and technological genius of the finest developers and games created in the last year. Congratulations to all the nominees! View the full list at **www.gamechoiceawards.com**.

AWARDS ARE PRESENTED IN THE FOLLOWING CATEGORIES:

2009 AWARD CATEGORIES

- Best Audio
- Best Debut
- Best Downloadable Game
- Best Game Design
- Best Handheld Game
- Best Technology
- Best Visual Arts
- Best Writing
- Best New Social/Online Game
- Innovation
- Game of the Year

SPECIAL AWARDS CATEGORIES

- Lifetime Achievement
- Pioneer
- Ambassador

PRESENTED BY GameDevelopers Conference

PRODUCED AND HOSTED BY GAMASUTRA gamedeveloper

GOLD SPONSOR Vicious Engine

THE 12TH ANNUAL INDEPENDENT GAMES FESTIVAL

The Independent Games Festival was established in 1998 to encourage innovation in game development and to recognize the best of the best of independent game developers. Just as the Sundance Film Festival benefits the independent film community, the IGF creates a similar event for independent game developers in including the student population of game developers. Now in its twelfth year, the IGF continues to evolve and bring to the community a sense of achievement for those in the indie scene. Congratulations to all the finalists! View a full list at **www.igf.com**.

IGF AWARD CATEGORIES:

IGF MAIN COMPETITION

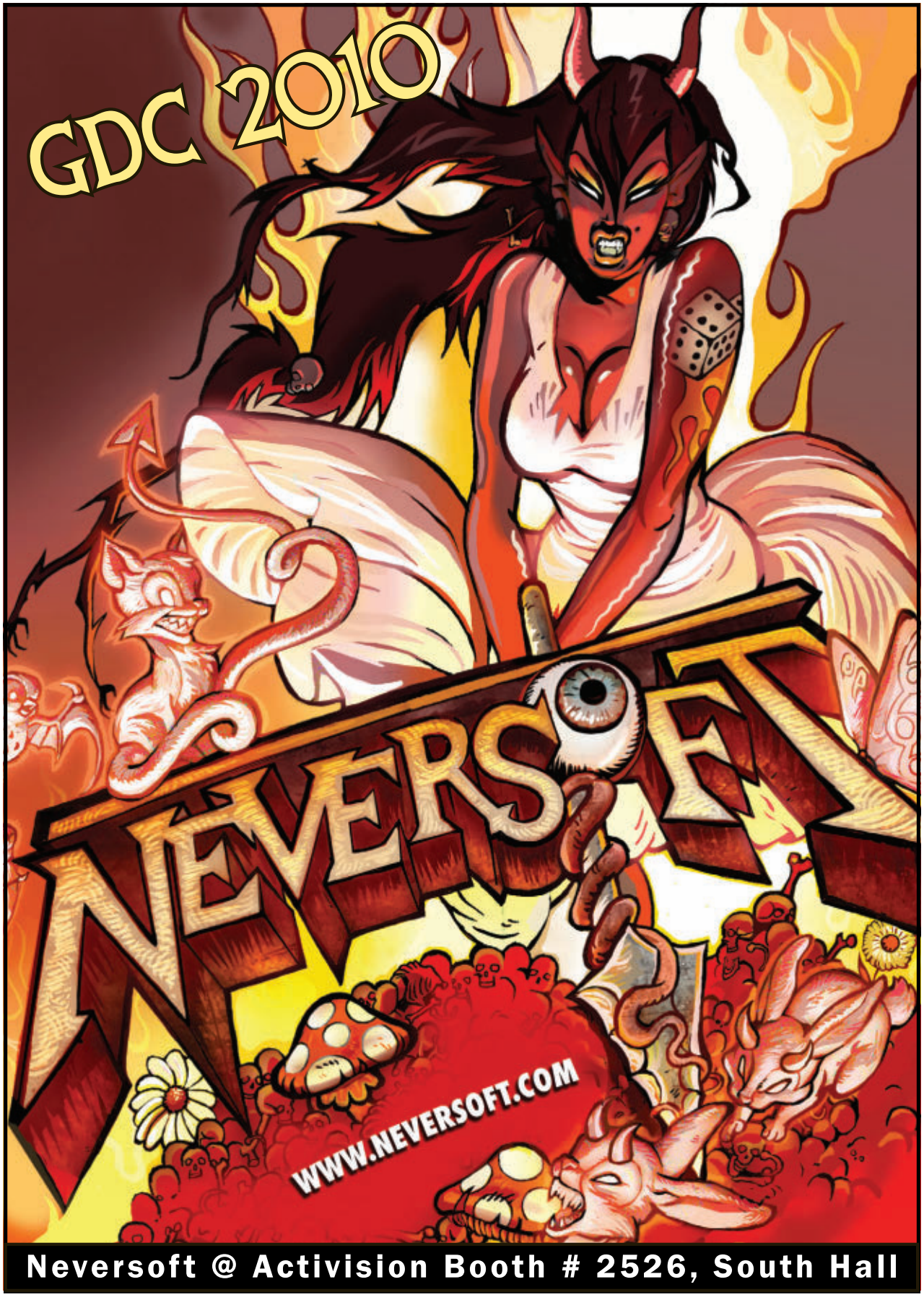
» Seamus McNally Grand Prize
» Excellence In Design
» Excellence In Visual Art
» Excellence In Audio
» Technical Excellence
» Nuovo Award
» Audience Award

IGF STUDENT SHOWCASE

» Student Showcase Finalist
» Best Student Game

C2D Direct2Drive — DIGITAL DISTRIBUTION SPONSOR
unity — GOLD SPONSOR
Microsoft — SILVER SPONSOR
DigiPen Institute of Technology — STUDENT SHOWCASE PLATINUM SPONSOR
ENGMIN — GOLD SPONSOR

THINK Services


GDC 2010
NEVERSOFT
WWW.NEVERSOFT.COM
Neversoft @ Activision Booth # 2526, South Hall

sidhe
WE MAKE GAMES

RECRUITING NOW

We are looking for creative and talented developers who will contribute to innovative titles for console, PC and handheld. Challenge yourself, embrace the New Zealand lifestyle, and join a studio with a fresh perspective!

ARTISTS | PROGRAMMERS | ANIMATORS | GAME DESIGNERS | PRODUCERS

www.sidhe.co.nz/liveworkplaynz (ask about our ability to fast track your work visa)

SCORE YOUR DREAM JOB AT GAMELOFT.

VISIT US AT BOOTH # 2429 AT GDC.

AT GAMELOFT, WE ALWAYS KEEP OUR EYES OPEN FOR FRESH TALENT.

WE EMPLOY CREATIVE AND AMBITIOUS:

* 2D & 3D ARTISTS
* GAME DESIGNERS
* PROGRAMMERS
* PRODUCERS
* AND MORE!

(GARGANTIAN GAMERS, DARING DAY DREAMERS, HILARIOUS HIPSTERS, SUPER-DUPER SMARTY PANTS, WONDROUS WHIZ KIDS, BRILLIANT BRAINIACS AND CLOSET SUPER HEROES ALL NATURALLY, WELCOME)

For more info about specific job opportunities in Barcelona, Buenos Aires, Montreal, New York City and Shanghai (Redsteam), please visit the career section at www.gameloft.com OR come to talk to us at booth #2429 at GDC.

We will contact you if we believe you experience is a good fit for an open position, but if you don't hear from us right away, don't fret. You might be our next hire!

gameloft
www.gameloft.com

gearbox software CAREERS

Join the Team!*

HAPPINESS

Be part of a Passionate Team
• Career Not a Job • Make Your Path
• Achieve Your Goals
• Free of Earthquakes and Wildfires!

CREATIVITY

• Independent and Unbound
• Dream Projects & Original Games
• Collaborate with Top Talent
• Make Meaningful Contributions

MONEY

• We Ship Bestsellers & Award Winners • Over $14 Million in Profits Paid Directly To Our Talent
• Exceptionally Low Cost of Living •
Get Paid to be Passionate about Work

Gearbox Brands
Brothers in Arms • Borderlands

Brands† we Rocked and Play in
Halo • Half-Life • Aliens: Colonial Marines
Tony Hawk • Samba de Amigo • 007

*The Awesome are welcome here.

Visit gearboxsoftware.com/jobs

†All games and franchises listed are the property of their respective owners and may be trademarked in the United States and other countries.
All inquiries regarding such trademarks should be made to their respective owner. All inquiries regarding Skagzilla specimens and Rakk Hive husbandry should be directed to the appropriate zoological authorities.

'TEAM UP'

GUERRILLA IS NOW RECRUITING FOR ITS UPCOMING PROJECTS

Senior **Build Engineers**

Senior **Game Designers**

Senior **Environment Artists**

Senior **Level Designers**

Senior **Multiplayer Designers**

Senior **Tech Programmers**

VFX/Tech Art Producers

gamesjobs@guerrilla-games.com | www.guerrilla-games.com

GUERRILLA

EDUCATED PLAY!

cave

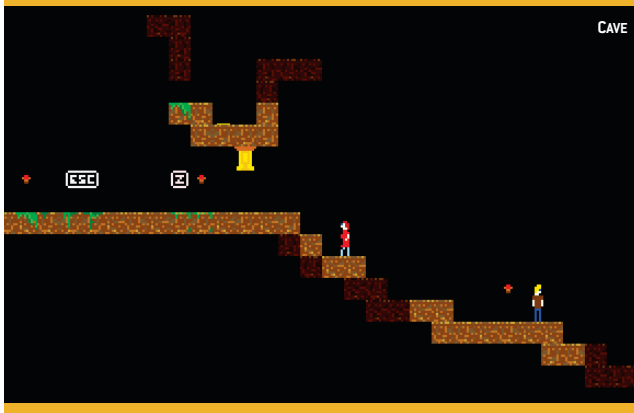
EXPLORING THE DARK SPACES BETWEEN TWO HEARTS

CAVE
http://users.design.ucla.edu/~chippermonky/cave

CREATED IN THE UCLA GAME Lab, Peter Lu's CAVE is a twisty little game about a boy and a girl exploring a cave. The characters have slightly different abilities and finding passage through the dark, maze-like environment requires the player to swap back and forth between the two, who reveal their thoughts about each other as they become increasingly separated. CAVE's simple, pixilated exterior belies an inner complexity that connects players to the deeper mystery of isolation and the struggle to connect with another.

Can you tell us a bit about the UCLA Game Lab?

Peter Lu: The game lab here opened up pretty recently. I did most of the development for CAVE in the game lab as I find it a very productive environment. The game lab here is a joint effort between the Design / Media Arts, Film TV, and Computer Science departments at UCLA to provide a resource for students interested in game development. Among other things, the Game Lab aims to encourage experimental design that can extend outside of just software. By this I mean incorporation of physical elements to games, be it costumes, flashy lights, or those crazy EEG helmets they have out there. One of the Design Media Arts professors, Eddo Stern, runs the lab. You might recognize him as a Nuovo award judge for this year's IGF.



What tools did you use to create CAVE?

PL: CAVE was written entirely in Python using the Pygame library for graphics and no additional dependencies. I used Graphics Gale and SFXR for art and sound respectively. In the interest of getting the project done for the IGF, I chose fairly simple technology.

Had you worked with other scripting systems before, such as ActionScript?

PL: I did some work with Flash back when it was still owned by Macromedia. I didn't really know what I was doing back then though. That was in High School. We didn't have any programming classes and I was too lazy to actually do examples in a programming book so I was pretty terrible at programming back then. Later on I took a class and learned C++ the right way and that put me in good standing to learn any other language. I've worked on unfinished games in both C++ using SDL and Java using processing. Python is particularly nice as a scripting language because it has a wide array of libraries and is not tied to a specific application.

What did you like about working with Python? Did you run into any limitations with Python or the Pygame library?

PL: A lot of people I've talked to won't look twice at Python because it's a scripting language with no parenthesis and has no immediate application like PHP and Flash do for web. It's really been the most enjoyable language I've programmed in. The syntax and duck typing took a while to get used to but now I save so much time with Python. As far as limitations go, unless the game is really complex, there's no need to use a powerhouse language like C++. For CAVE, I ran into a few efficiency issues but that was due to inexperience in writing a game engine. For games that are more graphically heavy or even 3D, one can use libraries like Pyglet, an OpenGL wrapper designed for 2D games or to make OpenGL calls directly. Certainly Python has its limits in efficiency as a scripting language but I see few independent games out there right now that could not have been written in Python. Also, using py2exe and py2app, it's almost hassle free to build a game for both Mac and Windows although I do most my development on Linux.

Was working on CAVE largely a solo project? Were online resources like TIGSource and others useful to you?

PL: CAVE was a solo project I decided to do because I was fed up with never finishing any of my games. I used few outside resources other than the Python and Pygame documentation and most of my feedback was from my friends and this other smaller forum that I've been posting at for a while. I like to keep a development blog on a forum. It's easy to lose motivation on a project without feedback every now and then.

There's a great expressive piece of animation in CAVE whenever the girl climbs over a ledge. Can you tell us about creating this simple, but very effective moment?

PL: The animations for CAVE were inspired by games like OUT OF THIS WORLD and PRINCE OF PERSIA. Whereas both those games used the simple but elegant solution of rotoscoping real actors, it would be pretty pointless for a 10x20 sprite. For CAVE, I did all the animations pixel by pixel. The animations were originally going to be maybe four frames but somehow ended up being 12. I really do enjoy the way smooth animations look at really low resolutions. The process of making all the animations was pretty quick though since the resolution was so low. In total I spent no more than five hours making the art and an additional 15 trying to export it to the right format. I must brag a little here though; it's quite a challenge when a pixel is the difference between a flat face and a massive nose.

What are your plans for your next game?

PL: Funny you should mention. I'm just about to finish my entry for this year's GAMMA IV. Hopefully my game will be accepted and then I'll be at this year's GDC. The name of the game is called ROULETTE, and it's about Russian Roulette.

I'm also working on a series of Tower Defense games with my roommate. We're doing it just for fun. I think that's the best way to develop a game. That's a feeling I might have forgotten about while working on CAVE, especially late in the development. Fun or not though, the work pays off in the end with the exhilaration of being done.

—Jeffrey Fleming

CALL FOR PARTICIPATION

FDG

June 19 - 21
2010

the International Conference on the
Foundations of Digital Games

Asilomar Conference Grounds
Monterey, California, USA.

Keynote Speakers

Markus Gross, Disney Research Zurich and ETH Zurich
Jane McGonigal, Institute for the Future
James Gee, Mary Lou Fulton College of Education

Workshops (June 18th)

Intelligent Narrative Technologies III
Procedural Content Generation in Games
Teaching Aesthetics in Game Design

Graphics & Interfaces
Game Studies
Learning in Games
Game Design
Infrastructure
(Databases, Networks, Security)
Computer Science & Games Education
Artificial Intelligence

FDG 2010, the International Conference on the Foundations of Digital Games, is a focal point for academic efforts in all areas of research and education involving games, game technologies, gameplay and game design. The goal of the conference is the advancement of the study of digital games, including new game technologies, capabilities, designs, applications, educational uses, and modes of play.

Find out more at **http://fdg2010.org**

An Official Conference of the
Society for the Advancement of the Science of Digital Games

SASDG

In Cooperation with ACM

SIGART

acm SIGCSE

Supported by

Microsoft Research

Photo credits:
1. (cc) (by) Andrew Fitzhugh - www.flickr.com/people/fitzhugh
2. (cc) (by) Steve Schnwarz - www.flickr.com/people/vasculata
3. © Asilomar Conference Grounds - www.visitasilomar.com
4. (cc) (by) Veronica Vale - www.flickr.com/people/vbv

Hire education

Our 142 alumni are working at 21 companies around the world making some of the best games and simulations on the market.

These grads are working on everything from the latest Call of Duty and Rock Band to the next Madden Football. And the list keeps growing.

So if you are looking to move from playing games to making them, FIEA's curriculum and faculty can teach you how to become a successful programmer, producer or artist.

And since it only takes 16 months to get your master's degree, the first day of your dream job could be sooner than you think.

Florida Interactive Entertainment Academy • www.fiea.ucf.edu • 407.823.2121

FIEA where active learning

UCF Center for Emerging Media

Bachelor of Science > Game Programming
Bachelor of Arts > Game Art and Animation, Game Design, Serious Game and Simulation
Master of Science > Game Production and Management

LEARN:

- Advancing Computer Science
- Artificial Life Programming
- Digital Media
- Digital Video
- Enterprise Software Development
- Network Engineering
- Network Security
- Open Source Technologies
- Robotics and Embedded Systems
- Technology Forensics
- Virtual Modeling and Design
- Web and Social Media Technologies

University of Advancing Technology
UAT
Learn. Experience. Innovate.

You can talk the talk. Can you walk the walk? Here's a chance to prove it. Please geek responsibly. www.uat.edu > 877.UAT.GEEK

DNA Plug (OC)

ACADEMY *of* ART UNIVERSITY

FOUNDED IN SAN FRANCISCO 1929 BY ARTISTS FOR ARTISTS

Gaming Artwork by Rob Arnold

Gaming Artwork Collaborative Student Project

Gaming Artwork Collaborative Student Project

TAKE CLASSES ONLINE OR IN SAN FRANCISCO

Advertising
Animation & Visual Effects
Architecture*
Art Education*
Fashion
Fine Art
Game Design
Graphic Design
Illustration
Industrial Design
Interior Architecture & Design
Motion Pictures & Television
Multimedia Communications
Music for Visual Media
Photography
Web Design & New Media

ENROLL NOW

EARN
YOUR AA, BA, BFA, MA, MFA OR M-ARCH ACCREDITED DEGREE

ENGAGE
IN CONTINUING ART EDUCATION COURSES

EXPLORE
PRE-COLLEGE SCHOLARSHIP PROGRAMS

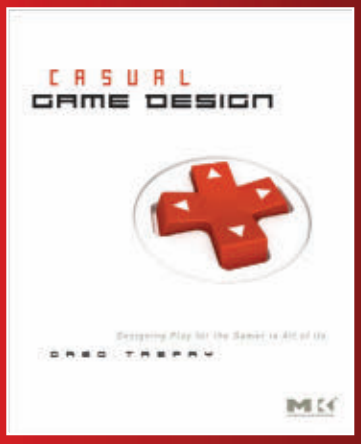
WWW.ACADEMYART.EDU
800.544.2787

79 NEW MONTGOMERY ST, SAN FRANCISCO, CA 94105

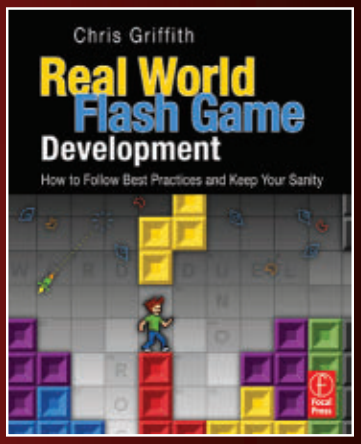
Accredited member WASC, NASAD, Council for Interior Design Accreditation (BFA-IAD), NAAB (M-ARCH)

**Degree program not yet available online.*

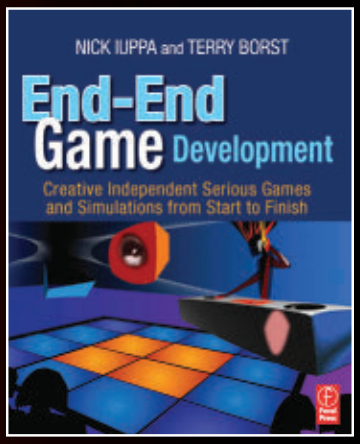
Focal Press & Morgan Kaufmann BOOKS



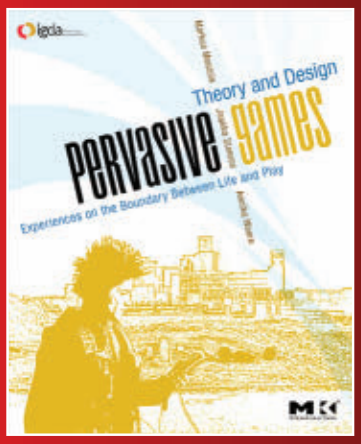
Casual Game Design
By Greg Trefry
9780123749536



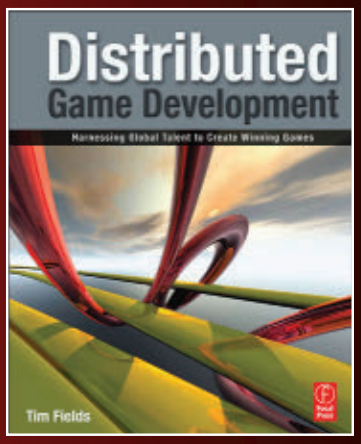
Real-World Flash Game Development
By Christopher Griffith
9780240811789



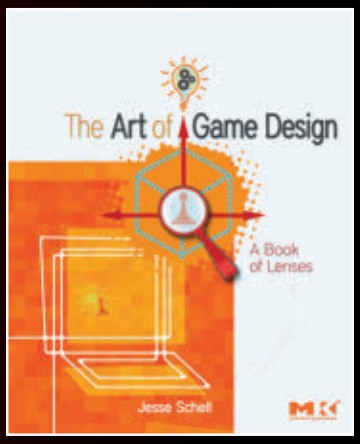
End-to-End Game Development
By Nick Iuppa & Terry Borst
9780240811796



Pervasive Games
By Markus Montola, Jaakko Stenros & Annika Waern
9780123748539



Distributed Game Development
By Tim Fields
9780240812717



The Art of Game Design
By Jesse Schell
9780123694966

>> Find us on YouTube facebook twitter

learn · master · create >> www.focalpress.com

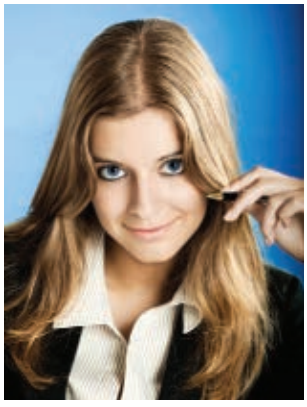
Technical information resources for computer professionals >> www.mkp.com

ADVERTISER INDEX

Game Developer (ISSN 1073-922X) is published monthly by United Business Media LLC, 600 Harriso ...L, 6th Fl., San Francisco, CA 94107. (415) 947-6000. Please direct advertising and editorial inquiries ...o this address. Canadian Registered for GST as United Business Media LLC, GST No. R1328807 ...Customer No. 2116057, Agreement No. 40011901. **SUBSCRIPTION RATES:** Subscription rate for the U.S ...s $49.95 for twelve issues. Countries outside the U.S. must be prepaid in U.S. funds drawn on a U.S ...ank or via credit card. Canada/Mexico: $69.95; all other countries: $99.95 (issues shipped via a ...delivery). Periodical postage paid at San Francisco, CA and additional mailing offices. **POSTMASTE** ...Send address changes to Game Developer, P.O. Box 1274, Skokie, IL 60076-8274. **CUSTOMER SERVIC** ...or subscription orders and changes of address, call toll-free in the U.S. (800) 250-2429 or fax (84 ...47-5972. All other countries call (1) (847) 647-5928 or fax (1) (847) 647-5972. Send payments to *Gam* ...*Developer*, P.O. Box 1274, Skokie, IL 60076-8274. For back issues write to *Game Developer*, 4601 V ...th St. Suite B, Lawrence, KS 66049. Call toll-free in the U.S./Canada (800) 444-4881 or fax (785) 83 ...566. All other countries call (1) (785) 841-1631 or fax (1) (785) 841-2624. Please remember to indicat ...*Game Developer* on any correspondence. All content, copyright *Game Developer* magazine/Unite ...Business Media LLC, unless otherwise indicated. Don't steal any of it.

THIS JUST IN!

FLASH UPDATES FROM THE DEVELOPMENT FLOOR



INTERN GIVEN SUSPICIOUSLY BIG MONITOR

>> A production intern was issued an unusually large monitor earlier this week, prompting some lunchtime grumbling and questions about IT department policy.

"I'm just saying, I've been here for almost a year now, and this intern comes in, and she gets a bigger monitor than me. It's a little weird, you know?" said one employee, who spoke on the condition of anonymity. "At least I still have better computer speakers than her, but I honestly don't know how long that will last—I noticed she got one of those Aeron chairs, too."

The intern, who was found familiarizing herself with a stack of obsolete design documents, appeared to be unaware of the controversy. An IT worker was seen peering around the corner at her, before rapidly returning to his department. He was unavailable for comment as of press time, but was heard remarking positively about "his chances."

DESIGNER NOT A REGULAR PERSON AFTER ALL

>> Late Friday afternoon, designer Todd Coleman destroyed the elaborate charade of his normalcy by inadvertently mentioning his *Inu Yasha* costume project. In answer to a query about his plans for the upcoming weekend, Coleman mentioned he would be "sewing like a total madman" in order to finish the ensemble in time for the local anime convention.

After an uncomfortable silence from the rest of the group, Coleman is reported to have said, "What? It's not like you guys don't cosplay too, right?"

Plans to move Coleman into a different bullpen toward the back of the building with the conspiracy theorist, the vampire guy, and the furry are moving forward with high priority.

EYE TWITCHES FOR A REALLY LONG TIME

>> Rumors from within the programming team indicate that build engineer Grant Hathaway's eye has been twitching for at least three days.

"Would you stop talking about my eye already? It's fine. I'm completely fine," Hathaway said when asked about the state of his eye. "I just need to do a few more code merges and just get the banana on the phone with the walrusface—sorry, what? I didn't hear what you said just then."

SNACK LEVELS ACCEPTABLE, SAY PRODUCERS

>> A team from the production department, acting on a tip that the kitchen was low on beef jerky, recently completed an inspection of the break room and pronounced the snack situation "under control." The impromptu task force, which consisted of one senior producer, one producer, and one associate producer, comprehensively surveyed kitchen stock levels and determined them to be satisfactory.

"We noticed that, yes, the beef jerky was a bit low, but it always goes fast. Plus, we have plenty of Slim Jims still left in there," the senior producer said. "Really, I don't like the employees having such a tremendous sense of entitlement. They forget just how good they have it here already."

The associate producer added, "That said, we are definitely going to talk to the admin and ask her to add Nutter Butters to her list."

ANIMATOR COMPARES EVERYTHING TO PIXAR

>> Allegations that Joe Gardner, an animator, constantly compares the studio's work to Pixar's appeared to be confirmed earlier today in a meeting to discuss motion capture plans.

"That's not how Pixar would do it," Gardner is reported to have said. "Pixar never uses motion capture."

According to eyewitness accounts, a long discussion about how the company is not Pixar then ensued, with Gardner eventually backing down. All seemed well until later that evening, when cleaning staff found a long, rambling and partially illegible love letter to Brad Bird on the floor of the conference room, and posted it on the corkboard, presuming it an important document.

CREATIVE DIRECTOR TO LOOK AT BUILD OF GAME

>> The team's senior creative director, Randy Gallione, made waves today with the announcement that he is totally ready to "check out" a recent build of the project that his team has been hard at work creating.

"I'm really excited to see the progress this talented group of people has made," said Gallione. "Just as soon as someone can burn a disc for me, I'll pop it in my dev kit here and give it a go. Would someone remind me how to turn on invincibility again, please?"

"NO MUSIC IN LEVEL 3" BUG WRITTEN UP FOR 5TH TIME

>> Chad Markham of Calabasas, California has contributed another bug detailing the lack of music in level 3, bringing the total number of active bugs about the issue in the database to five. "Steps to reproduce: play level 3. Notice that there is no music," Markham wrote.

As with the other bugs on this issue, Markham's write-up was sent to the developers, passed through the level designer, to the audio lead and eventually landed on the staff composer's queue.

"Thanks, guys," the composer wrote in a note to the bug. "I totally wouldn't have written any music for that level unless I got five bugs on it."

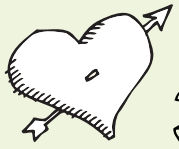
MATTHEW WASTELAND *writes about games and game development at his blog, Magical Wasteland (www.magicalwasteland.com).*

THERE'S NOTHING LIKE BEING A

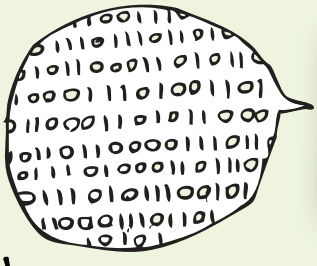
RAD GAME DEVELOPER

BINK VIDEO

It'll feel like you have SUPER POWERS when you use Bink, our amazing, super FAST video and audio codec - all in a simple, clean API.



You'll fall in LOVE with Granny, our run-time dynamic 3D ANIMATION SYSTEM with AMAZING content exporters.



GRANNY 3D

And with like Miles, our multichannel sound system - you get the world's FASTEST MP3 and Ogg DECODERS + way cool NEW high-level audio tools.

So it's obvious, that there's nothing like using RAD tools

Well, nothing except having a killer MUSTACHE

RAD GAME TOOLS ©

www.radgametools.com
(425) 893-4300