

god

DEV
EX
POSTMODERN
HUMAN
REVOLUTION



Change Your World

DOWNLOAD FREE* 2012 AUTODESK SOFTWARE

The future is closer than you think. Join a whole new class of artists in the Autodesk Education Community, and receive free* access to the same software and tools used by professional digital artists.

autodesk.com/freesoftware

Autodesk®



*Free products are subject to the terms and conditions of the end-user license agreement that accompanies download of the software. The software is for personal use for education purposes and is not intended for classroom or lab use.

Autodesk is a registered trademark of Autodesk, Inc., and/or its subsidiaries and/or affiliates in the USA and/or other countries. All other brand names, product names, or trademarks belong to their respective holders. Autodesk reserves the right to alter product and services offerings, and specifications and pricing at any time without notice, and is not responsible for typographical or graphical errors that may appear in this document. © 2011 Autodesk, Inc. All rights reserved.



AMG

CONTENTS.0112
VOLUME 19 NUMBER 1

POST MORTEM

20 DEUS EX
DEUS EX: HUMAN REVOLUTION is a huge project undertaken by Eidos Montreal, to revive the DEUS EX franchise. The team managed to turn a lengthy pre-production process into later successes through the power of planning. *By Jean-François Dugas, Martin Dubeau, David Anfossi, and Mary DeMarle*

FEATURES

7 14TH ANNUAL FRONT LINE AWARDS
In our annual awards that honor the best tools in the business, we show the best game development tools and software across art, audio, engines, networking, middleware, and programming/production tools, as voted by you, the readers. We also place one special tool in our Hall of Fame. *By Steve Theodore, Damian Kastbauer, Nathan Fouts, Eric Undersander, Zach Lehman, Noel Llopis, Mark Cooke, and Brandon Sheffield*

13 BUILDING A FREE-TO-PLAY BACK END
For independent companies that want to break into social or networked games but don't know what to do about the back end work, the makers of CREATURES 4 have an open source solution, which they outline here. *By Julien Hamaide*

31 GDC PREVIEW GUIDE
GDC 2012 approaches! This preview guide showcases some of the many talks from the show. *By Staff*

46 NEVER ENOUGH: AN INTERVIEW WITH DENA'S KENJI KOBAYASHI
DeNA is one of the largest social game companies in the world, operating its moogle platform across the world. We spoke with director Kenji Kobayashi about profitsharing, and paying for items that enhance player ability. *By Brandon Sheffield and Christian Nutt*

DEPARTMENTS

2 GAME PLAN *By Brandon Sheffield* [EDITORIAL]
The Generous and The Stingy

4 HEADS UP DISPLAY [NEWS]
Front Line Award finalists in quotes, MindCandy 3 released, and Indie Royale debuts

36 TOOL BOX *By Brandon Sheffield* [TOOL INTERVIEW]
An interview with Havok managing director David Coghlan

39 THE INNER PRODUCT *By John McDonald* [PROGRAMMING]
Practical Ptex for Games

48 PIXEL PUSHER *By Steve Theodore* [ART]
Hope and Change

51 AURAL FIXATION *By Jesse Harlin* [SOUND]
What's Next for the Next NextGen?

52 DESIGN OF THE TIMES *By Jason Vandenberghe* [DESIGN]
Write For The Player

55 THE BUSINESS *By Kim Pallister* [BUSINESS]
Business Books for a Turbulent Time

56 GDC NEWS *By Staff* [NEWS]
GDC China 2011 Closes With Record Attendance, 2011 Independent Games Festival China Winners Announced

57 GOOD JOB *By Brandon Sheffield* [CAREER]
Jaime Griesemer moves to Sucker Punch, who went where, and new studios

58 EDUCATED PLAY *By Tom Curtis* [EDUCATION]
A CLOSED WORLD

64 ARRESTED DEVELOPMENT *By Matthew Wasteland* [HUMOR]
HelpDev, the Friendly Publisher



THE GENEROUS AND THE STINGY

A GROWING TREND IN GAME DIRECTION

THERE WAS A TIME WHEN THE BACKS of game boxes had a slew of bullet points printed on them, illustrating numbers of enemies, features, and hours of gameplay. We decried this, at the time. "Games are more than a series of features!" we said. "Games are interactive experiences, and can't be reduced to a simple list of numbers of items and maps!"

Nevermind the fact there aren't nearly as many game boxes anymore—I'm starting to actually miss those bullet points, as well. Or at least I miss what they represented—because game designs are getting stingy.

GENEROSITY

In the past, many of the best games were (and a few of the current best games still are) generously designed. By this I mean in some games a lot of the content will not be appreciated or experienced by most players, but it's in there beneath the layers, because the developers felt it should be, and because they wanted to make a vibrant, living world. This allows players to keep discovering new ways to interact with and enjoy the game, even after playing it for hours.

I'll use a recent example: BEJEWELED 3. BEJEWELED is a proven property that's remarkably popular. You could probably spruce up the graphics, add some nice filters, and be done with it. But BEJEWELED 3 has a whole lot of interesting, weird ideas. It's got explosions, particle effects, and lush sound that would please any FPS fan, on top of eerie fantasy novel backgrounds that are clearly meant to appeal to the more casual. The music is a fantastic take on classic '90s PC games, with a bizarrely compelling MORTAL KOMBAT-style deep voice over. There are 8 modes to play which all use the same mechanics in clever ways, to form a very curious and very compelling amalgam that, ultimately, Popcap didn't need to go out of its way to create. The game is very generous to me as

a player. It keeps giving up little nuggets of enjoyment when I pay attention to this or that element of the design, art, or sound.

SCROOGED

Microtransactions and downloadable content are making their way into everything. Until recently, TETRIS was held up as one of those classic, pure examples of straightforward, fun-oriented game design. But now, a new iOS TETRIS has launched with a paid subscription. The game is 99 cents, and you can pay \$2.99 per month to get access to exclusive content, and most importantly, a booster that lets you increase your TETRIS rank faster. You get a core game for one price, then you get the "extra bits" for an additional fee. Features that might be generous in the design are sold at a premium. This is decidedly stingy, and almost every corner of the industry is trending this way.

CHOP AND CROP

Let's be honest about what we're doing here in the freemium space. We're taking what would traditionally have been a whole game, and we're chopping parts of it up to sell off individually. In ZELDA, you never paid for a sword that was slightly more powerful, you found it in the game after a long and arduous journey.

Most folks will tell you their free-to-play game is fully-featured, and all the microtransaction-purchasable items are unnecessary for full enjoyment of the game. But that sort of design is inherently stingy. Even if you design to compartmentalize, you're separating something from the whole, and the capacity for generosity to players is diminished.

A lot of downloadable content for triple-A games is similarly compartmentalized through DLC. I do understand it—these models can extend the life of games well past their "shelf" life. In fact, in a game that's inherently generous in its depth of content, like FALLOUT: NEW VEGAS or SKYRIM, DLC is almost

a welcome departure: A sidequest can be a breath of fresh air. But if you try to integrate that content into the core game, like the already generous DRAGON AGE: ORIGINS did by making an entire character and story arc downloadable, the overall feeling of generosity is diminished. Ultimately, while I do not think DLC or freemium games are inherently bad, I believe that these types of models have changed the way games are envisioned for the worse.

HOLISTIC DESIGN

In most parts of the industry, I'm seeing less and less interest in creating a full game that's finished in one go. This is a money issue, of course. Everyone needs money. But what about the love of the craft? What about the care put into making a game with an authorial vision, or an overall "feeling?" The feeling from games nowadays often comes from the community as much as (or more than) the structure and design. That's all well and good for some, but that way of doing things won't yield you a SHADOW OF THE COLOSSUS or a FAR CRY 2. If you chop off part of the game's "feel" into DLC, doesn't this inherently change how you treat it as a creator?

I do believe there are genuine ways to go about freemium models or DLC which are not so stingy. Consider the model of DEAD PIXELS on XBLIG, which is quite a generously designed game for a dollar, with hours of randomized gameplay (and a co-op mode). This game uses a neat model where the developer will begin work on free downloadable content if the game reaches a certain sales target. This in turn can incentivize more people to play the game, thus more sales, and thus more free DLC. But a large company would never take this risk—there's little guarantee this will work. But it's done for the love of the game, and as a player, that's what matters.

—Brandon Sheffield
twitter: @necrosfty



UBM LLC.
303 Second Street, Suite 900, South Tower
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

FOR DIGITAL SUBSCRIPTION INFORMATION
www.gdmag.com/digital

EDITORIAL

PUBLISHER
Simon Carless | scarless@gdmag.com
EDITOR-IN-CHIEF
Brandon Sheffield | bsheffield@gdmag.com
PRODUCTION EDITOR
Jade Kraus | jkraus@gdmag.com
ART DIRECTOR
Joseph Mitch | jmitch@gdmag.com
DESIGNER
Cliff Scorso
CONTRIBUTING WRITERS

Tom Curtis
Jesse Harlin
John McDonald
Jason Vandenberghe
Steve Theodore
Kim Pallister
Matthew Wasteland

ADVISORY BOARD
Hal Barwood Designer-at-Large
Mick West Independent
Brad Bulkley Microsoft
Clinton Keith Independent
Brenda Brathwaite Loot Drop
Bijan Forutanpour Sony Online Entertainment
Mark DeLoura THQ
Carey Chico Independent
Mike Acton Insomniac

ADVERTISING SALES

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

ADVERTISING PRODUCTION

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

REPRINTS

WRIGHT'S MEDIA
Jason Pampell e: jpampell@wrightsmedia.com
t: 877-652-5295

AUDIENCE DEVELOPMENT

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



UBM

WHERE YOUR GAME MEETS GAMERS

GAME.minder

GAME.minder is designed to bring developers and gamers together. With GAME.minder, you can keep your audience updated so they **never miss a game release.**

GAME.minder is the perfect platform for you to quickly and easily inform fans about release dates and availability of all your games, making it easy to build a fan base and excitement around your next game release.

We welcome EVERY game – indie games and major releases. All platforms and systems covered!



To get your game into GAME.minder for free, contact us at:
newgame@minder-app.com



Download GAME.minder today



Available on the
App Store

Visit us at GDC San Francisco,
booth 416 and see how GAME.minder
can change your business
and promotional model



gd

FRONT LINE AWARDS

finalists in quotes

WE'VE ANNOUNCED THIS YEAR'S FRONT LINE AWARD WINNERS (STARTING ON PG. 7), BUT THE FINALISTS ARE NO LESS NOTABLE. The winners were voted on in an online survey, but nominations were previously open to anyone willing to submit a new version of a tool. We allowed for comments in our survey, and in some cases, a large number of voters called for a tool that was not nominated. More interesting than that were the positive comments for tools that didn't actually win. Here, we've collected a few responses in each category, to highlight the runners-up. Unsurprisingly, the tools most folks requested that weren't on the list were open source or free.

ART

There was a huge clamoring for Blender to be on our list. One enthusiastic commenter stated that "Blender is getting very good. Some development BMesh builds are enough to pull me from Maya. Not ready this year of course, but check them out in 1-3 years and they probably will have a very compelling option."

Praise came in for Sculpttris, but even more so for the mainline product ZBrush, which one voter called "An absolute must for next-gen graphics. ZBrush offers the best means to acquire detailed meshes and normal maps via a high poly to low poly workflow."

AUDIO

A number of folks requested Audacity, which is free, and Sony's Vegas, which is not. But the vast majority of comments were actually about winner Pro Tools, with one person saying "Pro Tools = Audio power, fidelity, and now flexibility (since PT9). PT10 enables you to orchestrate your own score with its amazing synthesizers and plug-ins, so you now have a complete, believable-sounding warehouse of instruments at your fingertips to convey the emotion and feeling we strive to install in the player as they play."

ENGINE

By and large, there were few requests for any engines that weren't on the list. But there was a lot of praise for C4 and Unity.

>>PRAISE FOR C4:

"C4 has a clean C++ code base that is both easy to use and to extend. It is priced for indie developers to afford. And the community is very supportive and responsive."

"Terathon's C4 Engine continues to improve at an amazing pace, with ground-breaking

features and AAA support. The full inclusion of all source code for regular licensees is epic—any developer with the time and initiative to do things 'their way' is in for a treat."

"I will only vote for C4. As an aspiring engine programmer myself, I've done some considerably deep research on other game engines. C4 is much more versatile; it's incredibly well designed, comparably powerful, and very cheap. Realistically benchmarking higher than Unity, and designed with such great flexibility to rival (or surpass) even the budget-needy engines such as Unreal Engine or CryEngine; all while requiring much less human resources. It would be a shame to see it honored for less than its real greatness."

>>PRAISE FOR UNITY:

"While other engines are maybe more powerful in rendering and in entities management, Unity gives the power that Photoshop gave artists 20 years ago with [its] superb editor."

"As much as I love UDK, I'm wanting to make games, but don't have a coder. Unity has tools that allow me to create a game with node-based coding throughout the entire game (not just some parts as with Kismet). CryEngine and UDK (and the other engines listed) have some great features, but if I can't make anything with them, what's the point?"

"Unity already leads the pack in elegance and small team productivity—and with the forthcoming 3.5 release, it will offer a AAA-quality rendering pipeline as well."

MIDDLEWARE

Most non-Havok praise was aimed toward Scaleform, though Kynapse got a nod or two. One voter said that "Scaleform provides easy to use tools for creation of in-game HUDs and

menus. Saves lots of time in development." Another effused, "Scaleform provides you with incredibly nice and detailed UIs with many possibilities; what could you possibly ask for other than 3D UIs in an FPS game? Nerdgasm, at its highest."

NETWORKING

The biggest call here was for Steamworks, which many independent developers use to get multiplayer games going on Steam, but one voter called out OpenFeint, saying "OpenFeint / GREE are doing BIG things in the F2P space."

PROGRAMMING

Programmers are a vocal bunch, with strong opinions about their software tools and languages. Perforce got some callouts, with one commenter saying, "Source control is very important and Perforce is fully integrated into UDK. It's the obvious choice for a UDK developer." Another said, more boldly, "Where would we be without Perforce or SVN?"

FlashDevelop got some serious love as well. One voter said, "FlashDevelop is an alternative IDE to Visual Studio. It offers a similar, easy-to-use interface from Visual Studio. Plus, like Visual Studio, you can debug your code using breakpoints. Overall, it's a great editor to use if you do not, or do not want to have access to Visual Studio." But that's nothing compared to this absolute statement: "Sometimes I choose to write a game in ActionScript just so I can program it in FlashDevelop. By far my favorite IDE."

Congratulations again to the winners and finalists—and don't forget to nominate your favorite tools next year!

—Brandon Sheffield

mindcandy volume 3 released

The demo scene has long been a testing (and proving) ground for new technologies and techniques, and MindCandy has been one of the best anthologies of these around. The MindCandy series of DVDs (and now Blu-rays) presents the best of PC demos, running on high-end machines and captured at maximum capacity. While many folks like to run these demos themselves, there's something cathartic about just watching them unfold before you. This third edition of the series brings 40 of the best PC demos released from 2003–2010, running at their best in the high-definition Blu-ray version.

Demos range from 64k marvels of compression to technical monsters, with a host of extras on the disc. Bonus content includes seven hours of demo-oriented talks from the NVScene 2008 seminars, and commentary from the demos' makers. Jim Leonard of Oldskool.org (and author of

the MindCandy 3 release) told us that "over 95% of the demo content has commentary from the original authors," adding that "The commentary mix is roughly 25% technical talk and 75% design talk," which may be of interest to game developers, as well as demo-sceners (and there is certainly some crossover between those two camps).

In addition to being a fun curiosity, while emulation of old computers and operating systems is improving, there is no sure fire method of preserving these important parts of our digital culture. At the very least, the demos in the MindCandy series will be preserved at optimum speed and display. MindCandy Volume 3 is available now at www.mindcandydvd.com.

—Brandon Sheffield



indie royale bundles launch

Newly launched game bundle Indie Royale (owned by UBM TechWeb and Australian download site Desura) groups together four downloadable, independently developed PC titles every few weeks, using a unique pricing system that changes as time passes and more people purchase the games.

While bundles will start at a heavily discounted rate (usually \$2.99), that price will automatically increase as time goes on. Generous purchasers, however, can help lower the price for everyone else by choosing to pay more than the current minimum, earning a mention on the front page of the site in the process.

The first few bundles have included Wadjet Eye Games' 2D

adventure game GEMINI RUE, indie RPG CTHULHU SAVES THE WORLD, Nicalis' arcade puzzler NIGHTSKY, and many others, with recent bundles sporting a bonus prize as well.

Titles offered in Indie Royale bundles are available in various configurations including direct downloads for Windows PC, as well as redeemable PC keys for Steam and indie-focused download service Desura, with new formats forthcoming.

"When we first launched the service, we had no idea that we'd sell almost 70,000 bundles and over 350,000 games in under one month!" explain organizers. In order to keep things running smoothly, and add a "mystery box"-style approach to bundles,



organizers are now opening pre-orders for future releases at slightly higher than launch price. Organizers say pre-orders are "a simple way to guarantee you get the bundle, albeit sight unseen. This means that you won't know what games are in the bundle until it launches. But you are

guaranteed to get the bundle at a low minimum price, and you will be supporting handpicked, high quality independent games along the way."

Find out more at the official site, www.indieroyale.com.

—Staff



Taking

Game Development

to the next level



DevTest Studio

The 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 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. Watch a recorded overview. Request an online demo.

www.techexcel.com
1.800.439.7782



FRONT LINE AWARDS 2011

//////// Every year, we at *Game Developer* honor the best tools in the game business with our Front Line Awards, which are now completing their 14th year. But games are rapidly changing beasts. The industry seems to be evolving before our very eyes. While middleware and game tool use has risen dramatically in recent years, this was traditionally the domain of middle-sized teams that either couldn't afford to roll their own engines, or needed to get up and running faster to stay competitive. As more small teams have made strides in the mobile and social spaces, middle-sized developers have grown, or shrank, or fallen by the wayside.

What does this mean for game tools? One might have expected a lot of small tools to crop up to meet the demands of small studios. But what we've seen has been more that the larger tools are finding ways to scale small, and stay relevant across a wide variety of team sizes. Many of these tools have indie licenses, browser and mobile versions, or special versions for smaller teams.

As such, this year's Front Line Awards don't look amazingly different from last year's. But that shouldn't be taken as stagnation. It took a lot of hard work for all these companies to remain relevant in such a rapidly changing space, and it's to their credit that over 11,000 developers weighed in and chose the winners.

Here's how our process works: Products were nominated in all categories by the readers of *Game Developer* magazine and Gamasutra. Once all nominations came in, the editors of *Game Developer* consulted with other industry experts to select finalists, and finally, readers determined the winners in each category through an online survey.

Nominations were open to all new products (and new versions of products) related to game development released between September 1, 2010 and August 31, 2011. Awards are given in the fields of art, audio, game engine, middleware, networking, and programming tools. We also honor one venerable tool with a Hall of Fame Award.

Congratulations to all the winners and finalists that help us make great games!

—Brandon Sheffield

HALL OF FAME

XNA GAME STUDIO

MICROSOFT [[HTTP://CREATE.MSDN.COM](http://create.msdn.com)]



////////// Somewhere between programming language and game engine resides the full-featured, free, professionally-developed programming suite known as XNA Game Studio. Allowing game developers to create their own engines and worlds while taking care of the lowest level and most commonly-shared issues is where XNA shines.

Flash back to 2006, when I'm working for Insomniac Games on RESISTANCE FALL OF MAN for PlayStation 3 using proprietary technology. Gameplay work is rewarding, but with dozens of engine programmers constantly shifting the programming landscape beneath, it can also be very frustrating. After years in the industry, I wanted to work on my own creations. Something burns inside me, a glowing pixelated mass, convincing me to make my own games for high-definition consoles. Flash forward to 2008, I've left Insomniac Games and WEAPON OF CHOICE—my own company's first console release—hits the Xbox 360, delighting and surprising run-n-gun fans the world over, even being recognized by the Guinness World Records. The dream has been realized.

XNA Game Studio made it possible—you can release any game you devise onto the Xbox 360. A current-generation game console has been opened up to developers everywhere.

Xbox Live Indie Games is the perfect home to modern classics such as MINER DIG DEEP,

LEAVE HOME, PROTECT ME KNIGHT, the SOULCASTER series, EPIC DUNGEON, and the radiangames series, all console-designed games to their cores. The best indie games have a lot to contribute to the industry in terms of design and artistry despite being available for bottom-level prices. Yet without the intrinsic connection of XNA to the Xbox 360 and smooth facilitation of the Creators Club peer system, the widespread console release of these games would still be a dream.

The breadth of technology and design on display is staggering, with games ranging from slick, high-definition 3D (some with the stereoscopic 3D visuals), to 2D pixel adventures straight out of Game Boy or Genesis alternate histories, to flights of fancy using Xbox avatars, to experimental games eschewing visuals altogether in favor of audio and rumble support. With only the lowest levels of programming supported, there is no "feel" or "look" to an XNA game—it simply gets to exist as the developer intended, and they are free to design in nearly any direction they can imagine.

The programming suite in XNA Game Studio is elegant, robust, and expansive. All low-level game programming such as rendering, audio control, graphics shaders, controller input, storage—nearly everything—is handled in a well-documented and fully-designed manner. Several programming issues have filled me

with anxiety as I've approached them (such as dealing with memory cards or implementing online leaderboards), and I'm consistently impressed with the ease and ability with which XNA has afforded me to meet those challenges. Not only are the systems well designed and documentation-ready, but many, many helpful and immediately useable tutorials and full games with source code have been created to make the task of making your own engine or game even easier.

It's truly a triumph of development that XNA Game Studio allows for unrestrained games on the Xbox 360 via Indie Games. And I'm happy to say it's also easy to develop for the PC and even the burgeoning Windows Phone 7 market. Mommy's Best Games' own XNA-developed SERIOUS SAM DOUBLE D released recently via Steam and other PC portals, having the way paved by excellence such as TERRARIA and BLUEBERRY GARDEN.

Having developed for three separate platforms all using the underlying XNA technology has allowed me amazing creative and engineering freedom, while providing me the leg-up needed to get my games going in the first place. I can't wait to see where XNA Game Studio is heading next after leading the charge to open gaming console for all developers!

NATHAN FOUTS is president of Mommy's Best Games.

3DS MAX

AUTODESK [[HTTP://USA.AUTODESK.COM/3DS-MAX](http://usa.autodesk.com/3ds-max)]

////////// Perennial artist favorite 3DS Max is one of the workhorses of game development. Since its introduction in 1995, Max has been used in literally thousands of titles. Max pioneered the plugin architecture that all modern graphics packages use, and has historically set the standard for efficient polygon modeling. With its powerful scripting and support for Microsoft .NET architecture, Max is the wellspring of hundreds of game studio pipelines. It's hard to imagine the game industry without 3DS Max.

This year the long-awaited XBR rendering engine, now dubbed "Nitrous," finally brings the full power of modern graphics hardware and multicore computing to the process of modeling. With effects like real-time shadows and ambient occlusion, it makes for more accurate previews without exporting or offline rendering. More importantly, though, Nitrous handles complex modern scenes with aplomb, rendering millions of polys interactively. It allows real-time editing without the need for tedious viewport management.

Finalists

- / Modo 501 [Luxology]
- / MotionScan [Depth Analysis]
- / Substance Designer 2 [Allegorithmic]
- / ZBrush 4R2 [Pixologic]

The new version of Max also tackles one of the suite's enduring weaknesses. For more than a decade, Max users have envied Maya's node-based procedural texturing system, known as Hypershade. This year's Max integrates Substance, the procedural

texturing engine from Allegorithmic. With Substance, Max users can create complex effects by combining and compositing noise functions (or, if they're less technically inclined, by picking from an extensive library of existing procedures).

Finally, this year's Max introduces a new renderer from the makers of Mental Ray. While Mental Ray is famous for producing rich, realistic imagery, it's also notorious for exposing a nearly infinite variety of incomprehensible parameters. Any artist who has spent hours trolling the net in search of the right numbers for things like final gather point interpolation, caustic filter kernels, or dielectric outside refraction will be pleased to hear about IRay, the new rendering engine which is intended to produce Mental Ray quality with point-and-shoot simplicity.

Instead of providing programmable shaders with lots of sliders to twiddle, IRay uses a physics-based rendering system with a limited set of predefined material types. This provides a consistent look and realistic results, though many veterans will find the lack of fine control and artistic exaggeration unnerving. They may not mind so much, however, when they see how much faster the renders are; like the Nitrous viewport renderer, IRay makes good use of modern graphics cards.

Naturally, there are also a host of minor improvements in Max 2012. From better-looking icons to more logical organization of UV editing tools, the package continues to improve with age. Artists tend to be ferociously loyal to their software (and, in any case, the same company owns all three of the big 3D packages) so the new Max may not trigger a tidal wave of conversions. Nonetheless, this year's iteration certainly cements Max's well-earned reputation as a key pillar of the game industry.

STEVE THEODORE is art director at Undead Labs.



PRO TOOLS

AVID [[WWW.AVID.COM](http://www.avid.com)]

////////// The manipulation of sound is a delicate art that carries with it enough science to explain the complex relationship between vibrating molecules of air through the atmosphere. On one hand an ephemeral sonic dance, on the other a wholly quantifiable puzzle of frequencies and amplitudes. Today's digital audio workstation (DAW), equipped with a microphone to record and the power to blend, bend, and shape recordings at your whim, is a sonic architect's finest building block.

What separates Avid's (formerly Digidesign's) Pro Tools from other audio sculpting utilities is the firm foothold across film, game, and music

studios, rooted in the company's combination of proprietary hardware and software products. At an early stage in the development of digital audio, Pro Tools was among the first to offer a unified solution to the question of how to get audio in, out, and through our wildly varying desktop computers. Once firmly embedded in control rooms throughout the world, Pro Tools became the standard against which all other tools would be measured.

Finalists

- / Fmod Designer 3.4.8 [Firelight Technologies]
- / Miles Sound System 9 [RAD Game Tools]
- / Soundminer HDv4.3 [Soundminer]
- / Wwise 2011.2 [Audiokinetic]

Behind every DAW user interface lies a staggering amount of code that has been arranged to support the import, modification, and export of high-quality digital audio. While some might argue that you can hear differences between these solutions, they all rely on a backbone of hard science to push sound in and out of the digital domain. It's often said that "it's not about the tool, it's how you use it," and how you interface with the software sets Pro Tools apart.

Locked in the muscle memory of every Pro Tools user is a set of hot-keys and shortcuts that enable the swift execution of design without thinking. Coupled with a clean and simple visual representation, it's all about familiarity. While the lower level audio engine remains mostly transparent, new features often end up being UI related, unlocking workflow speedups through the addition of draggable, clickable-sound shaping tools, or navigating quickly between views. It turns out that working with sound is also a very visually intensive discipline, and Pro Tools succeeds by delivering an experience that facilitates the mastery of flow for those who have been working with it for years.

Love to love it, hate to love it, or love to hate it, Pro Tools is a vital part of the creation of sound for games and deserves a victory lap for enabling efficient and accessible design for all.

DAMIAN KASTBAUER is a freelance audio professional at Lost Chocolate Lab.

M I D D L E W A R E

HAVOK PHYSICS

HAVOK [WWW.HAVOK.COM]

//////// My first experience with Havok Physics came while working on a driving prototype at Electronic Arts in 2002. We were experimenting with curved surfaces for rendering terrain, but we hadn't really thought about how such an unusual representation would work as collision geometry. The integration with Havok was surprisingly easy, though, and we quickly had vehicles speeding across the rolling landscape.

Later, in 2007, I was tasked with adding Havok Physics to the early game engine would become 38 Studio's action RPG, KINGDOMS OF AMALUR: RECKONING. At that time, the player character could run around in an empty 2D plane, pursued by simple, AI-controlled enemies. We could also load and render buildings and other models. As a first pass at physics, I fed the triangles of our environment models into Havok as collision geometry and hooked Havok's character-proxy class into our 2D character movement.

At the following milestone review, I cobbled some models together to make a little dungeon, and during my presentation, my player character hiked across uneven terrain, climbed stairs, and dropped off ledges. A band of enemies pursued me through the dungeon, comically bumping against obstacles and each other. It was the first time that parts of our engine had come together to resemble an actual game.

Finalists

- / Kontagent kSuite User Analytics platform (Kontagent)
- / Kynapse 2012 (Autodesk)
- / Scaleform Gfx 3.2 (Scaleform)
- / XaitControl 3.4 (Xaitment)

Havok Physics documentation includes illustrated help files, detailed comments embedded directly in each C++ class's header file, and an invaluable, comprehensive set of sample projects. Despite these excellent resources, I expect that teams looking to get the most out of Havok Physics will enlist Havok's support services. For RECKONING, our account manager regularly arranged phone chats to discuss the direction of our project and warn us of potential pitfalls ahead. For specific questions

and problems, we used the support web site, and Havok support engineers were quick to respond.

RECKONING will ship very soon with Havok Physics supporting character movement, ragdolls (including some that explode!), inverse kinematics, a variety of collision queries, and more. The development team chose it and stuck with it because it's a mature, well-designed API with powerful tools and high performance across multiple platforms. Indeed, I've witnessed the product improve continuously since its inception over a decade ago. At the same time, Havok has branched out from core physics to offer character animation, destructible meshes, cloth simulation, and pathfinding. The most recent addition is Havok Script—a virtual machine for Lua script. I've talked to industry colleagues who've used this, and they rave about the script debugger integrated into Visual Studio ("it has conditional breakpoints!").

This year, Havok cements its number-one position among game physics providers. As a gamer, I have no doubt I'll be seeing that buzz saw logo gracing splash screens for many years to come.

ERIC UNDERSANDER is a freelance game programmer based in Austin, Texas.

E N G I N E

UNREAL ENGINE 3

EPIC GAMES [WWW.UNREALTECHNOLOGY.COM]

//////// Epic's Unreal Engine 3 has proven itself yet again as the most popular licensable engine for AAA console game development. It's not hard to see why. Developers all around the world continue to choose the Unreal Engine for its impressive and powerful suite of content creation tools and a frequently updated engine core that supports the latest and greatest technology. From the engine's original focus on first-person

Finalists

- / C4 Engine 2.6 (Terathon Software)
- / CryEngine 3 (Crytek)
- / Unity 3.4.1 (Unity Technologies)
- / VisionEngine (Havok)

shooters, we're now seeing games ranging from third-person action games, to RPGs, to one-on-one fighting games. The variety of genres of games built with the engine is proof that the technology is extensible and powerful enough to support many use-cases.

For developers in the trenches there are numerous popular engine features that make day-to-day life easier and more efficient. Designers

love the power that UnrealEd gives them. The editor environment gives the design team the power to craft gameplay scenarios and prototypes rapidly via its intuitive interface and robust Kismet visual scripting system. Artists have an extensive suite of tools at their fingertips to make characters and game scenes shine. From the visual material editor to the integrated Lightmass static light-map generator, Unreal gives artists the ability to focus on making great art with limited technical fuss. Programmers have access to numerous built-in systems for rendering, AI, physics, and networking that help get the team up and running quickly. These systems have been debugged extensively in the 50+ games that have shipped using the engine, so coders can rest easier knowing that the foundation they are building on top of is proven.

In the last two years, what started as an engine focused entirely on the high-end PC and console space has moved into two new arenas: mobile and small-scale development. As mobile devices continue to make more of an impact on our industry, Epic has further extended the engine's iOS capabilities. In addition, through the UDK program, which puts the engine in the hands of any developer with an internet connection, individuals and less-established teams can harness the Unreal Engine's power without a large license fee. Combined with the popularity of mobile development (and now supporting browsers), UDK offers a compelling engine platform to build smaller-scale games on.

With so many developers now familiar with the engine and tools, the talent pool of experienced developers who are familiar with Unreal continues to grow. UDK is now putting the engine in the hands of students who can learn the engine in school, then smoothly transition into a job at an Unreal licensee post graduation. Unreal experience has become a marketable skill for employees. And on the other side of the fence, for employers, licensing the engine brings the benefit of getting new hires up to speed quickly. If a new employee is already experienced with the engine they can become productive much more quickly on a new game because they don't have to learn the intricacies of a new engine and tool chain from scratch.

The core technology, tools, and business benefits have made the Unreal Engine *Game Developer's* winner for best engine. In 2011, for games big and small, Unreal Engine 3 is a fantastic choice.

MARK COOKE is cofounder of Shiny Shoe, a mobile game company.

LUA

[WWW.LUA.ORG]

////////// Lua is a programming language invented in 1993 by Robert Ierusalimsky (and others) at the Pontifical Catholic University of Rio de Janeiro in Brazil. Initially designed from necessity as a data programming language, it quickly grew into a powerful general purpose language. Lua is very popular in the game development community as an embedded scripting language. There are several reasons why developers pick Lua over other scripting languages such as Ruby, Java, or Python.

POPULARITY. Lua is popular in part because it is popular! It's achieved a critical mass of experienced developers in the game industry, which means Lua skills are transferable from company to company. The use of Lua as a scripting language has also meant there is significant familiarity with it in the mod community, particularly with WORLD OF WARCRAFT and CRYISIS. Many level designers start out with a background in mods.

EMBEDDABLE. Lua is designed to embed easily in a game engine. The Lua interpreter is supplied as a relatively small set of C files which can be very easily incorporated into the source code for your game or tools. The Lua script files can then either be loaded directly, or as precompiled byte code. This flexibility makes it very easy to allow for run-time editing of script files.

Finalists

- / FlashDevelop 4.0.0
[FlashDevelop Project]
- / Hansoft 6.6 [Hansoft]
- / Perforce 2011.1
[Perforce Software]
- / RAD Telemetry [RAD Game Tools]

LIGHTWEIGHT. Lua is relatively small and simple, both in terms of the source files, and the resultant code and run-time memory usage. So it does not take up very much space, and does not add much time to a full re-compile of your game. Since it's so simple, there's less to go wrong when compiling, and you have fewer problems with dependencies.

SPEED. Lua is a relatively fast programming language. The stock version of Lua is usually more than twice as fast as Ruby. Mike Pall's excellent LuaJIT implementation has a bytecode interpreter that is several times faster than stock Lua. If you are working on a platform that allows it, then the full JIT compilation of LuaJIT adds another order of magnitude in speed, often rivaling the raw power of C/C++. LuaJIT also comes with several extensions that are useful to developers, such as bitwise operations and simplified binding with C/C++.

EXTENSIBLE. It's very straightforward to call a Lua program from C++ and equally easy to call C++ from Lua. But you can also very easily add custom data types (such as matrices and vectors) to Lua. It's also quite straightforward to bind your C++ data types and functions with Lua. This can be automated with some pre-processing (and there are products such as LuaBind, SWIG, and several others to help you with this), or you can fairly easily roll your own if you have specific needs. It's also simple to extend the functionality of Lua with libraries either written in Lua, or as extensions in other languages. While not as ubiquitous as Perl or php, there are numerous useful Lua libraries available.

NOEL LLOPIS runs Snappy Touch, an independent iOS game developer.

GAMESPY

GAMESPY TECHNOLOGY
[WWW.POWEREDBYGAMESPY.COM]

////////// 1997 marked the start of a new breed of video game enthusiast: the online gamer. Moving away from single-player games and leading the way into the world of multiplayer online play, GameSpy saw a gap in technology and stepped onto the scene. At the time, connecting players all over the world was just a theory. The bold steps taken by GameSpy forever changed the industry. For 15 years GameSpy has been a leader in providing online services to all gamers, including resources such as data tracking, user generated cloud storage, in-game commerce, and of course matchmaking services. The company's robust services have been dominant in the field of game networking technologies.

Finalists

- / DeNA Mobage [DeNA]
- / Open Feint [Aurora Feint]
- / Photon 3 [ExitGames]
- / ReplicaNet 7.0 [Replica Software]

I work with Emotional Robots, Inc., creator of WARM GUN, which uses a ranking system for its players. GameSpy provides the framework for this to happen. The technology is also helping us provide users with the ability to track their settings, scores, and stats, synced through GameSpy's intuitive cloud solutions. The central factor to any multiplayer game is the matchmaking system, which without the support and tools from GameSpy would simply not be possible. Not only were we able to implement the functionality of multiplayer matchmaking, but also tailor the data to fit the needs of a unique mobile product.

GameSpy Industries has fostered a reputation for providing great tools that help developers build great games. But I want to take a minute to speak to the key reason why we chose GameSpy ourselves. The Open program, started February 24th of this year, was created to provide independent developers with the same level of access to GameSpy software as AAA studios. Lowering the barrier of entry on middleware is quickly becoming an industry trend, but GameSpy takes it a step further by providing direct interaction with its engineer team. GameSpy also offers on-site services for some developers living in San Francisco. In addition to unrestricted access to industry-leading multiplayer software, independent developers can hone their business and marketing skills via communication with GameSpy and IGN. It's no wonder the program has since grown to host over 1,200 developers!

Bringing the entire package together at GameSpy, as with any company that wishes to be successful, is the people. The Open program simply would not be possible without the wonderful people who bring GameSpy to life. Over the past four months of development, there have been difficulties that nearly caused WARM GUN to sacrifice features for functionality. GameSpy was always there to assist in development and provide critical feedback. GameSpy's commitment to delivering a combination of tools and service makes them deserving of this Front Line Award.



ZACH LEHMAN is executive producer at Emotional Robots.



Foundations of Digital Games 2012

Workshops: 29th May 2012

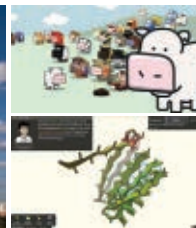
Conference: May 30th to June 1st, 2012



Raleigh, North Carolina



www.fdg2012.org



FDG 2012, the International Conference on the Foundations of Digital Games, first held in 2006, brings together experts and researchers in all areas of game design in one place to share new ideas and contributions to their respective fields, including:

Game Studies - studying games, players, and their role in society and culture

Game Design - methods, techniques, studies

Serious Games - building and evaluating games for a purpose; learning in games

Games Education - preparing students to design and develop games

Artificial Intelligence - agents, motion and camera planning, navigation, adaptivity, dialog

Game development - tools, frameworks, networking, databases

Graphics and Interaction - rendering, modeling, animation, understanding players and interfaces, interaction techniques

FDG promotes direct dialogue and information exchange between industry and academia concerning the scientific foundations of digital games, technology used to develop digital games, and the study of digital games and their design. This year **FDG** is hosting for the first time an experimental game festival to feature new ideas in game design. These will be playable games judged to have design innovations by game design judges.



Sponsors: **Microsoft Research**

Microsoft Studios

grano
Graphisme, animation et nouveaux médias
Graphics, Animation and New Media

Organized by: Society for the Advancement of the Science of Digital Games (SASDG)

JULIEN HAMAIDE

3 solution for indies

building a free-to-play web back end

✗ Most free-to-play games alternate between player engagement and player frustration. The latter is used to push the player to open his purse. Cheaters can easily overcome the frustration by hacking the game and bypassing the developers' restrictions. Storing all important data such as in-game credit counts on an online system can limit the risk of exploits.

With free to play, you need as many people playing your game as possible. Usually, the game rewards players that spread the word and invite friends into the game. Once again, this can only be accomplished with a back end that stores all that information.

At the same time, the back end allows players to access your game anywhere. By way of example, I'm currently working on the virtual life game CREATURES 4, which is going to be available on mobile and PC/Mac platforms. You can play your game on PC, then switch to your mobile device to continue playing on the road. This article will guide you through the creation of a back end that will do all this. Extensive code resources are available for this article at www.gdmag.com/resources.

WHAT THIS BACK END IS AND ISN'T

✗ The presented back end is a message exchange system. The game produces events and ask for validation on the server: Is this action valid? What's the experience value after I complete this quest? Am I allowed to buy this item? It is not a multiplayer server such as those found in first-person shooters. Nor is it a virtual world server like you find in MMORPGs. The back end stores the persistent state of the game, e.g., the player's info, the experience level, quests completed, amount of currency, and all social interactions. This facilitates, for example, visiting a friend's world and exchanging Norms (one of the titular creatures in our game).

We designed the back end to scale, both up and down, quickly and easily. If active user numbers change frequently, it can adapt and save resources. To limit the up front cost of server installation, the software is ready to run on cloud services.

Finally, all in-app purchases are validated on the server, preventing cheaters from accessing paid items. It validates the purchase receipts and delivers required data only if valid. So even if a hacker succeeds in unlocking items, they won't be usable. For example, an object might miss a script only delivered by the server at purchase time.

ARCHITECTURE

✗ The back end is designed as a grid of servers. Figure 1 illustrates the grid setup. The responsibilities are dispatched, and each node only handles a single type of task. The node types are:

- » **Front-end nodes:** Servers that have contact with the player's client
- » **Database nodes**
- » **Processor nodes:** Nodes that are used to apply application-specific processes to data

The system is built using open source software, permitting a fast but robust implementation. Those pieces of software are also maintained and checked for security holes. Focus can then be applied to application-specific software pieces.

THE OPERATING SYSTEM

✗ The nodes run under Linux. This decision was driven by different criteria:

EASE OF CUSTOMIZATION The system can be tailored to your needs. The drivers and components can be kept to a minimum. It's also easy to create a default installation DVD/ISO that contains the customized system.

PRICE Windows licenses cost lots of money. If the nodes run on a cloud system, the license

is included in running costs. In a server farm containing tens of machines, the price can quickly make a difference.

EASE OF INSTALLATION AND UPDATING Most Linux distribution comes with a package manager that handles installation and updates. The installation is done with a single command. For example, on a Debian-like distribution, the installation of our front end is done by running this command:

```
apt-get install fishingcactus-frontend
```

The dependencies are defined in the package, and the missing software is automatically installed and configured. We run a private package repository where all our updates are pushed. The servers are easily upgraded using the package manager. Manual steps are limited to the minimum, decreasing maintenance time.

Linux comes in various different flavors. There is no universally "best" distribution—every one comes with its pros and cons. It really depends on your level of knowledge and your personal taste. At Fishing Cactus, we use Debian-like distribution for all our internal servers, so the choice was easy.

THE FRONT END

✗ The front end is the only server at war. It is exposed to the real world. Attacks and exploit tentatives are inevitable. The software must be able to resist to most problems. This reality brings us to use the Apache web server. It is actively maintained and is a proven piece of software, running millions of web sites. The communication with the client is therefore done using the HTTP protocol. The application is developed in PHP. This choice was driven primarily by team experience. Nowadays, many languages are supported for a web application, such as Ruby, Python, and Lua. Choose what you feel comfortable with.

As a web server is designed to handle lots of short requests, the application uses a REST-like approach. The client does not maintain a connection to the front-end node and the connection has no state. The communication is made of small independent connections. User information is transmitted as a session token with each message. The client is therefore not "attached" to any front-end node. If a node crashes, the client can switch to another one. The session token is requested at session start-up using the user authentication information (more on this connection in the security section).

Each node is completely independent and is an interface to the inner network. Adding and removing nodes does not require any setup. The client follows the algorithm shown in Listing 1 to decide which node it will connect to. Two things can force a client to change its server node: either

the node fails or the node is overloaded. As the algorithm shows, the server can at any moment ask the client to move to another server. Load balancing is achieved directly by the nodes. But if all servers are at the limit, the clients will spend their time jumping from one node to another. For security and quality of service, new nodes must be added when the charge of the servers reach 90%.

Listing 1.

```
At init :
server_ip = dns_get( "api.example.com" );

When a request is sent :

send_request( server_ip, request );

if( server returns a new server ip )
{
    server_ip = ip from request result;
}
else if( server is down )
{
    server_ip = dns_get( "api.example.com" )
}
else
{
    update variable from request result
}
```

The messages are constructed as Remote Procedure Calls (RPC). Arguments are encoded as JSON objects (JavaScript Object Notation). Arguments contain the action (or function), the session token, and the hash of all the parameters. The result is also encoded as a JSON object. It contains an error code (if any) the result to the function, and all the updated values. A hash is also added to prevent packet forging. Listing 2 shows a typical request/result pair.

Listing 2.

```
Example of message sent on quest completed:
{
    "session_token" : 'some token',
    "action" : "quest_complete",
    "quest_idenfifer": 12,
    "hash" : '...'
}

Typical response:
{
    "error" : 0,
    "xp" : 1234,
    "norm_capacity" : 6,
    "coins_won" : +20,
    "hash", '....'
}
```

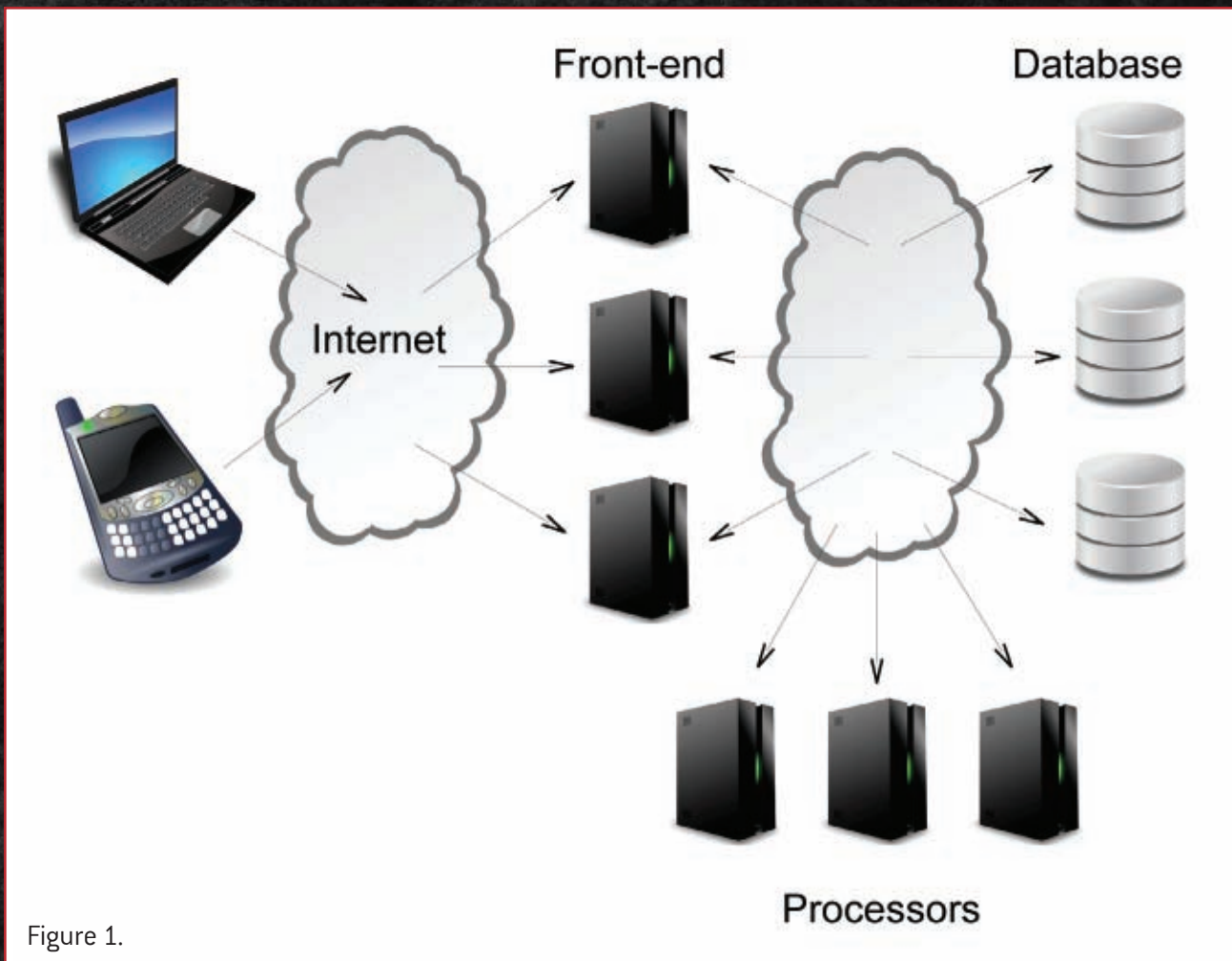



Figure 1.

THE DATABASES

X Databases contain two types of information: the gameplay data and user-specific data

The former is almost static. It only changes when new content is added or tweaked. We chose an SQL database. The tables are designed to ease content addition. Quest, shop items, and levels are described in this database. Adding new content does not require a software update (which may take more than a week to be approved). This allows fast iteration and tuning. If a quest is too hard, we can quickly adjust it. Even if the system does not contain user information, a fail-over system should be installed. Otherwise the game will stop working as soon as the database fails.

The user specifics are excessively modified. Every user action changes the profile data. The amount of data is also proportional to the number of users. If you have a million users online, with each of them making 0.5 requests per second, the database must handle 500,000 requests per second. The database must thus be fast and

scalable in size. These constraints exclude all SQL servers. Our choice turned to Redis, a key-value store. In such a store, there is no table nor relation between data. It's just a map of pair. Below, you'll find typical keys in our system for a CREATURES 4 user.

```
user:john_doe:id -> 1234
user:1234:username -> john_doe
user:1234:level -> 15
user:1234:experience -> 23765
```

The user profile contains several values that should be stored using different keys. Those values are only linked semantically. In an SQL database, the relationship between data is stored in the table structure. In a key/value store, no structural relationship exists. To bind variables, two keys are necessary. The example below shows how to link the username to its user number, and shows how a SQL request is converted to a Redis get. The command here shows the key layout must be designed to allow requests.

```
SQL: select name from user where id=$id;
Redis command: GET user:$id:username
```

The save data files are usually too big to be stored directly in Redis. The files are saved to network storage, and the path is stored in the Redis cluster.

CONSISTENT HASHING

X To achieve scalability, we may need to add new database nodes. But how to include a new server without changing the distribution of existing keys? If the distribution is made with a modulo, the index of used servers is defined by $\text{hash}[\text{key}] \% N$ where N is the number of servers. But if N changes, every key hash needs to be recomputed and redistributed. This may take a while, and the system is unusable during the redistribution. Consistent hashing solves this problem, distributing hashes onto a ring.

To choose on which server we store the key,

the key hash is applied on the ring. From this point, we search for the next server in a clockwise order. To ensure uniform repartition, each server will have a different spot on the ring. Figure 2 shows how the servers spread around the hash ring. When a new server is added to the system, the keys that lie within the portion dispatched to the new server are not available anymore. To handle the problem, we use the following algorithm: If the key is not found (either when reading or when writing) on the expected server, we continue on the ring to the next server. If the key is found here, we copy it to the expected server and delete it from its old position. If the key is not there yet, we continue to the next server. If the key is not found at all, an error is raised. This technique is really efficient, but in some cases, all servers will be queried for a specific key. When we know a key is new (such as when creating a new user profile), we skip the key search. The new node is then populated gradually, cleaning other nodes at the same time.

If nothing specific is done to the keys, a user profile might be spread among all database servers. To prevent this phenomenon, we limit hashing to a specific part of the key. For user keys, "user:\$id" is only used for the hashing. Below, you'll see how we specify the part of the key to be used in node selection. We use {} to specify which part of the key to hash. Each key is stored on the same node.

```
{user:1234}:username -> john_doe  
{user:1234}:level -> 15  
{user:1234}:experience -> 23765
```

REDUNDANCY

Redis servers store our user data. This must be protected against destruction and failures. In classical backups, the servers can fail over to redundant servers. But Redis has an internal mechanism to replicate itself. It can accept another instance as a slave, duplicating the data. The slave is an exact copy of its master. Every time the master changes, it sends the modified values to its slaves. If the master fails or gets disconnected, the application detects it. The slave is promoted to master, and the application can start using it. When the old master recovers, it detects its old slave has become a master, and sets itself up as the new slave. These operations require no human intervention. If a master is unable to recover itself in a reasonable time, a message is sent to the IT staff. This is important, as the server might require human action to recover. Figure 3 shows how the switch between master and slave happens.

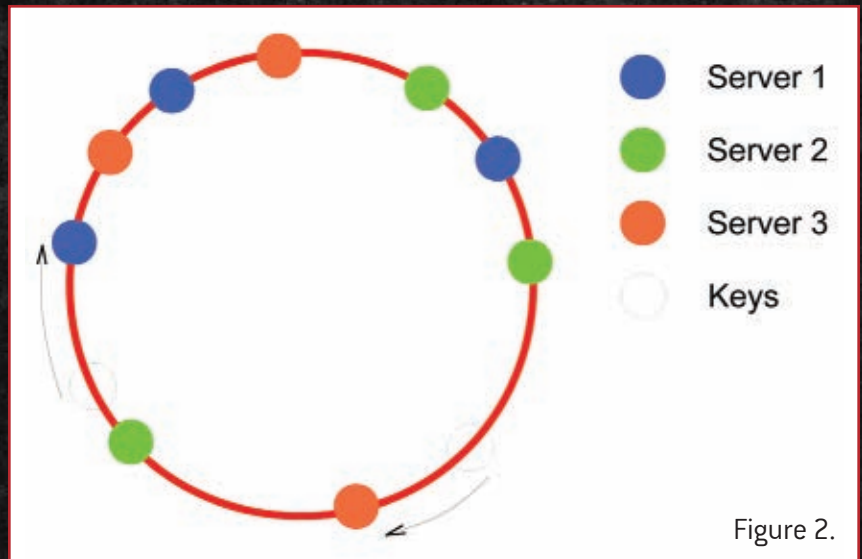


Figure 2.

PROCESSOR NODES

Processor nodes are application-specific. They take care of all intensive computation. In CREATURES 4, they are used for three main purposes. The first is to provide a validation of sensible data. Creatures are exchanged by players. The exchanged file must be checked against malicious modifications. We therefore prevent those files from crashing their destination clients.

The second purpose is to detect abnormal behavior. The user database is scanned to find impossible player situations. It checks for abnormal player progression and anomalous currency amounts. Application-specific rules are applied to detect such situations. If a rule is violated, the team is contacted and the case is investigated. If a cheat is detected, the team contacts the player to solve the problem.

The third purpose is player conquest and player rewards. If a player hasn't played for a while, the system sends her some reward by mail to bring her back into the game.

SECURITY

In multiplayer games, cheating and hacking are a big problem, impacting the experience of other players. Also, as the connection is in clear text, players' personal information should not be transferred. The security of the system is based on a simple but efficient concept.

As explained before, the connection is not permanent. The user information should be added to each message. When the user connects at the start of the play session, the client asks the server for a challenge. The player returns the answer to the challenge by using his password. Below you'll find a typical challenge computation:

```
Challenge answer = md5( text( challenge ) + text( md5( password ) ) );
```

The server validates the challenge using the same formula. The password is never transmitted, and the server only stores its md5 hash. If somebody eavesdrops on the connection, he will never get access to the password nor its hash. If the challenge is answered correctly, a session token is provided. This token will be added in each message, anonymizing the message (no user name is sent after the password challenge has succeeded). After session initialization, a session key is created in the Redis database. This key allows the server to recover the user from the session key. This set of keys binds the session to the user, as below. To confirm the session is valid, we check that both keys match.

```
user:1234:session_token ->  
123456789ABCDEF  
session:123456789ABCDEF:user_id ->  
1234
```

To prevent message forging, all messages contain a hash of all arguments. The password hash is added to the arguments create the final hash. If you change any of them, the hash is going to be completely different. So, changing the message requires knowledge of the algorithm and the password. If this is not enough, a message number can be added. All the responses also contain the hash of all returned values. If it were easy to modify those values, the cheater could induce incorrect values to the client, such as an incorrect money amount.

In-app purchases are also validated on the servers. Whenever a player buys an item,

the receipt is sent to the back-end, which can validate this receipt with the provider's servers. If it is correct, the bought item is added to the user profile and the server returns the updated value.

If you still need more security, the web server can communicate through a secure connection using SSL (Secure Socket Layer). But this encryption will require extra processing power on both client and server sides. Also remember that any software that uses encryption in the U.S. must be authorized for export.

SCALABILITY

✗ When the number of users playing the game increases, the back end should be able to adapt. Our system has 3 axes of scalability: bandwidth, database storage size, and processing power. Those categories match exactly one type of node. If bandwidth becomes too small, we simply add a front end node. The node is added to a database of online nodes. The load balancing algorithm contained in the node will automatically forward clients to the new node. Clients continue their communication with the new node with no requirements other than their session token. The DNS server is also adapted to serve the new IP address to the client.

Increasing the database storage size is trivial, but removing the server requires more work. The key/value contained in the removed server needs to be assigned to active nodes. To limit the impact and maintenance time needed to perform the node removal, the data is separated in several small database clusters. User data is stored on one cluster, items owned by users on another one. Finally, a third cluster contains all exchanged data to be processed. Processing nodes are the easiest to add. When the node is up, it automatically connects the database to get some jobs. There's nothing else to be done.

Unfortunately internal network bandwidth can quickly become a limitation. If you use hardware in a data center, you need to pay attention to the network connections. Processor and front-end nodes should not communicate with each other. The database can be connected to those nodes using two separate physical networks. In a cloud system, there is little you can do about it.

ADDITIONAL TOOLS

✗ As the game already requires an internet connection, an analytic system can easily be added to the game to gather information about the player. What's the mean time spent to finish a given quest? How long does a user play each day? What is the behavior of buyers? Every question is useful in tuning your game. If a quest's success rate is really low, it may frustrate the player and make her quit. If an item is never

bought, maybe it's too expensive. Such a system really helps you tune and tweak your game, increasing player retention.

When all this analysis leads to a gameplay decision, the data must be changed. If the game is huge, the database might be hard to change by hand. It also might lead to bogus data. To prevent those problems, a dashboard can be developed. In addition to facilitating game edits, it also provides a layer of verification. This interface is of course application-specific and should be specialized for each game.

Monitoring the back end is unavoidable. If you want to assure quality of service, an alert should be sent every time a problematic situation is detected. There are several monitoring systems out there, including open source ones like Nagios. These systems monitor the services on the servers and report problems as soon as they occur. In our office, a dashboard screen is hung in the central room, viewable by everybody. It summarizes the current memory, disk, and power usage of all servers.

CONCLUSIONS

✗ Building a back end for a free-to-play game is accessible to any team with some server administration knowledge. The architecture is common to all games, and only the gameplay rules must be integrated in the system. As it uses open source software for most critical parts, your time can be spent on the game itself. The security is based on existing techniques, which are simple but robust. The investment is minimal and can be reused easily. At Fishing Cactus, we built this system in two months, and deployed it within a single month. With the advent of cloud services, a company can start its own cluster without buying and managing real hardware. Now your only limits are how the game uses it. 🎮

JULIEN HAMAIDE is technical director of Fishing Cactus. He was previously senior programmer in a scrum-based team at 10Tacle Studio Belgium. By working in small studios, he gets to work on everything from low-level console programming to AI passing through multithreading. He has contributed to the Game Programming Gems series, AI Programming Wisdom 4, and Game Engine Gems 2.

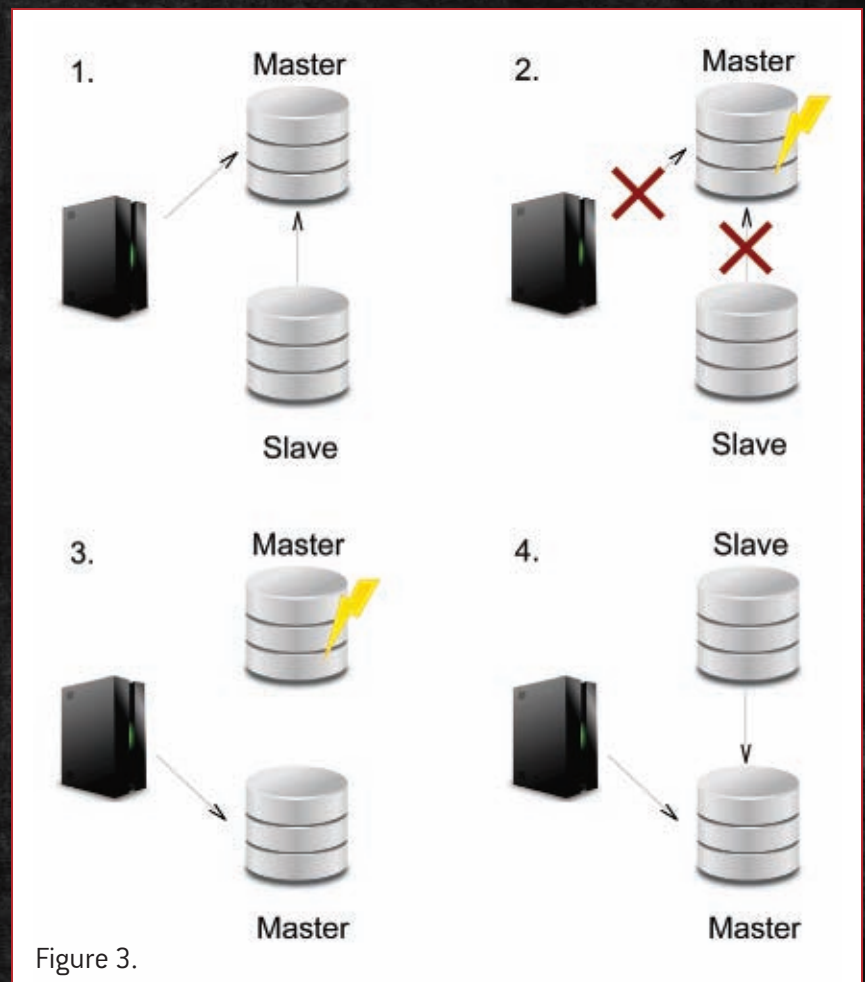


Figure 3.

GDC 12

learn / network / inspire

GAME DEVELOPERS CONFERENCE[®]

SAN FRANCISCO, CA
MARCH 5-9, 2012
EXPO DATES: MARCH 7-9

2012

Register before February 2nd and save up to 35%!

VISIT WWW.GDCONF.COM FOR MORE INFORMATION.



The **Game Developers Conference**[®] is the world's largest and longest-running professionals-only game industry event. Join us at GDC, March 5-9, 2012, for five days of learning led by some of the biggest names in the industry.

Learn

SUMMITS AND TUTORIALS

[MONDAY & TUESDAY]

- AI** AI
- IT** Game IT
- ✦** Localization
- ▶** Games for Change
- 🏠** GDC Education
- 🎮** Independent Games
- 📱** Smartphone & Tablet Games
- 🌐** Social & Online Games

MAIN CONFERENCE SESSIONS

[WEDNESDAY-FRIDAY]

- 🔊** Audio
- 💰** Business, Marketing & Management
- 🎮** Game Design
- 🎬** Production
- 💻** Programming
- 🎨** Visual Arts
- 📊** Monetization [SPONSORED]
- 🎓** Game Career Seminar [FRIDAY]

network

**GDC
Play**

[TUESDAY-WEDNESDAY]

**Business
Center**

[WEDNESDAY-FRIDAY]

**Career
Pavillion**

[WEDNESDAY-FRIDAY]

inspire



**INDEPENDENT
GAMES FESTIVAL**

[MONDAY-WEDNESDAY]



[WEDNESDAY]

Expo Floor

[WEDNESDAY-FRIDAY]







DEUS

EX

HUMAN REVOLUTION

J E A N - F R A N Ç O I S D U G A S

//// In 2007, when David Anfossi, Jacques-Belletête, and I [Jean-François Dugas] joined Eidos-Montreal, we were given the task of reviving the DEUS EX franchise. This project was a dream come true for us. It was the chance to work on one of the most acclaimed video game franchises out there, reviving its lore, its unique blend of action RPG, and its epic scale. For four years, with a relatively small team, we went through the ups and downs of crafting one of the most ambitious titles of 2011. Building DEUS EX: HUMAN REVOLUTION was one of the most challenging projects for all of us. It's not only been one of the most challenging games we have ever worked on, but also the most memorable one!

WHAT WENT RIGHT

1/ conception.

In early 2007, we started to work on DEUS EX: HUMAN REVOLUTION with a very small group of people. The core creative team was composed of David Anfossi (producer), François Lapikas (senior game designer), Jonathan Jacques-Belletête (art director), and Jean-François Dugas (game director).

We started slowly by replaying the original DEUS EX games, trying to understand what worked and what didn't. We needed to remind ourselves of what made this franchise so special in the first place before we made any attempt to revive it. We also spent a lot of time reading books, watching movies, and playing other games, always taking notes and sharing our ideas along the way.

Gradually, we began holding the first brainstorming sessions for what would become DEUS EX: HUMAN REVOLUTION. Our process was totally organic, as every individual in the core team presented their ideas to the rest of us. We would then debate the ideas until we found what we were looking for.

Then, at the end of every week, we'd present the week's results to other team members who were not part of the core creative unit. These people then had their chance to comment on and challenge our ideas, and express their concerns

about what we presented while also offering up their own ideas. Our goal was to get buy-in from the entire team. We knew DEUS EX: HUMAN REVOLUTION would be an ambitious game, and that it was critical to get people from all departments on board as early as possible.

For the first few months we wrote down all our high-level intentions for every aspect of the game on large sheets of paper. Every time we nailed something, a sheet explaining it would be hung on the wall where the core team was located for everyone to see. It felt like we were in our garage, or something like that. We even had a fridge full of beer there most of the time!

By the end of the concept phase, the walls were covered with our design intentions. We didn't have a single design document outside of those sheets of paper. If the building were to unexpectedly catch fire and burn down, there would have been no DEUS EX: HUMAN REVOLUTION! The concept phase lasted five months, more or less; and for us, it was the best experience in our respective careers, because we worked hard in a very friendly environment, using a freeform process in which everyone could participate. It was a true democracy. Our days were organized with objectives to meet, but rigidity mostly stopped there.

The first time we had to present our ideas to Eidos management, we sat them in the middle of

the room and presented one sheet after another until we had covered all major points. Along with a few concept art pieces that showed off our early ideas, that's all we had ready at the time. After two days of this odd, informal style of presentation, Eidos's feedback was short but full of meaning: "DEUS EX is back!"

In retrospect, our concept phase was built around fostering the creativity of the people involved in it. We created a stimulating environment that allowed their creative ideas to bloom, without making them feel the pressure of having to be formal in their approach. It was more important that we were efficient at what we needed to do, rather than looking good for the higher-ups. After this experience, I'd find it hard to go through another concept phase without employing these successful conditions.

2/ the blueprint process.

Pre-production is a crucial phase. By the time it ends, you need to have figured out and answered most of the questions that will arise during the next phase of your game's development: production. The better equipped you are, the better the chances are of running production smoothly.

Our production ended up being turbulent at times for various reasons, but because we were able to nail down the experience of the game on paper first, before production started, we were able to remove many of the risks that come from not knowing where you're going. We owe this foresight to what we called the "blueprint process."

In a nutshell, the blueprint process was our way of building the game on paper before starting any real production. It was a process we used to merge gameplay and story together into one coherent experience. It was also a tool we used daily during production to help us maintain our goals and the experience we wanted to deliver. By the time the concept phase for DEUS EX: HUMAN REVOLUTION was finished, we had a strong game design concept to work with. Having already brought in our lead writer, Mary DeMarle, we also had a one-page pitch document that expressed our basic story idea, but that was pretty much it. Before starting the blueprint sessions, we needed to build six or seven major plot points. With these in hand, Mary's team could continue to flesh out the story. Now, we were ready to start the blueprint. For three months, we held meetings from 10 a.m. to 5 p.m. every day. Meeting attendees included the writers, level designers, level artists, game designers, art director, and game director. We spent each day building mission details focused around the story points. We adapted gameplay when the story was more important, and revised the story when gameplay needed to prevail. We also broke down the enemy lists, environments, number and types of dialogues, cutscenes, and the like. Those three months of meetings turned out to be really successful in anticipating and answering many of



DEUS EX
HUMAN REVOLUTION™

PUBLISHER Square Enix
DEVELOPER Eidos-Montreal

RELEASE DATE
August 23, 2011 (North America)

PLATFORM
Xbox 360, PS3, PC, OnLive

DEVELOPMENT TOOLS
Visual Studio, Perforce, CDC engine, Maya, 3DSMAX, Photoshop, first-party development tools

BABIES BORN TO TEAM MEMBERS DURING DEVELOPMENT 23

TIMES THE PS3 DEV KIT TRAVELED FOR PRESS OPPORTUNITIES IN NORTH AMERICA, ASIA, AND EUROPE 28



the questions that arose during production. They succeeded because we went into each meeting keeping in mind the following key points:

- ✗ The blueprint process is a creative, organic process that gives a true sense of ownership to the team by involving key members from several departments. It's not one or two people leading the show; it's a true group effort, and all the players must be fully committed.
- ✗ The process is meant to give us a clear vision of what we're trying to build. By the end of it all members of production know not only what they need to create to support the game experience but also why it's needed.
- ✗ The blueprint itself allows us to plan what needs to be built during production with a great deal of precision (roughly 80% of the content). This means how many maps, scripted events, cutscenes, dialogues, challenges, exotic gameplay elements, and other assets we will need. It also helps us see whether we can deliver them on time.
- ✗ Having a blueprint makes it easier to cut without damaging the game's cohesiveness. Since we've got visibility into all of the game's content, we can rapidly pinpoint what is important and what is not as critical and make important decisions accordingly.
- ✗ The process forces us to focus on the really important aspects of the game, decreasing the amount of time spent on things that will be marginally important or sparsely used in

the final product. For example...

- ✗ If the design calls for friendly AI and we discover during the blueprint that it's really used only once or twice, we can determine if it's worth having or if we can cut it sooner rather than later.
- ✗ If we discover through the blueprint that we have a lot of scripted events, we can begin to ask questions like, "Do we build a specific team to take charge of this? Or, do we reduce the number of events instead, and if so, what is the impact?"

The blueprint process is not magic! It's a creative and organic process, so by definition it's hard and exhausting. To make it work, you need a committed team, otherwise it's pointless. While it doesn't answer all the technical and creative questions you'll have, it does give you quite a lot of answers beforehand. It also helps you make a realistic production plan and schedule!

In the end, what we have in our blueprint document is very similar to what's in our final product. We did have to cut some things along the way, but because we had a clear vision from the start, it was easier to identify the best places to cut and fix any narrative/mission problems that our cuts might have created.

3/ cutting early and often.

Even though it was emotionally hard on several occasions, we had to make cuts. We didn't shy away from cutting a feature or a map that we

were attached to when we knew we wouldn't be able to carry it through properly.

For example, at some point before production kicked in, we had three city hubs planned: Detroit, Hengsha, and Bangalore. Fortunately, during pre-production, we chose to build our first city hub, Detroit, as a demo, because we knew that kind of semi-open map would be the most challenging to create.

This strategy worked out well and led us to realize that three city hubs were too many for our production capabilities. One had to go. By looking at the blueprint, we were able to identify Bangalore as the least explored in terms of gameplay and story. Because it was fairly behind on a conceptual/artistic standpoint as well, it became the obvious choice to cut. So, we reworked its story and gameplay bits so they could be folded into the Hengsha map.

Later, during production, we realized we still hadn't cut enough maps. A huge chunk of Hengsha (the upper city HUB part) would also have to be cut, along with a few compounds (smaller maps such as Sarif Manufacturing Plant and Picus) that had been planned—one in Hengsha and one in Utah. Again, we looked at the Blueprint and figured out a way to fit some parts of the story into the lower part of Hengsha, while fitting the other parts into a revisit of Detroit later in the game. It took us about three weeks to rework the story and Blueprint, but doing so allowed us to ship the game with the desired level of overall quality.

Meanwhile, in the features department, we had about a thousand content-related things we wanted to create. I'm serious: one thousand. With our production manager, Martin Dubeau, we spent weeks prioritizing, cutting, and making sure we were putting the right people in the right places to get things done in accordance with our newly identified priorities.

In the end, we managed to maintain the integrity of our design, but only because we did what needed to be done even when it hurt. In fact, the key to our success might have been just that.

4/ team experience and fearlessness.

When we started the project, we knew we were facing a huge challenge. For us, the only way to meet the mandate was to build the most experienced team we could, and for a long time, the average team member's experience level was around 10 years!

Now, working with an experienced team is not necessarily a recipe for success. It doesn't automatically mean things will go smoothly all the time. Over the course of the project, we faced all the problems a team can face: ambitious scope vs. unrealistic deadlines, tough recruitment in key areas, inefficient tools, sudden, and unexpected departures. Having an experienced bunch of directors and other key people on staff made it easier to not panic or lose focus.

Every time we faced a crisis, we took the time to sit down and really examine the state of things. Several times during the project, our directors and leads would head off-site for several days of meetings, just to look at the current problems and brainstorm possible solutions, as well as examine all the pros and cons that came with each one. We talked (and sometimes fought) about real issues that touched upon sensitive, personal subjects, but I always felt we did it with the utmost respect.

In the end, we came back with strong contingency plans that helped us overcome whatever obstacles we were facing. Our experience gave us the means to structure and challenge ourselves, establish strong processes, make tough decisions, and defend the project when there were doubts from others. I think we also really became a team in the process.

Having an experienced team is only one part of the equation, however; the other is having project leaders who possess courage and strength of will.

It's easy to report the state of things to your publisher when everything is going smoothly, but what happens when you have mostly bad news to communicate? Oftentimes, it's tempting to hide as much as

you can in hopes of toning down the negative reactions.

Our leaders were never afraid to—pardon my [Quebecois] French—"have the balls" to address these situations in an honest way. Every three months or so, we had to present the progress of our project to the execs in order to receive clearance to continue moving forward. We decided to approach these meetings with total transparency. I remember flying to a meeting where we didn't have much good news to share. I said to David Anfossi, "We either get out of here alive and continue the project, or we die with it during the meeting. At least there won't be any confusion!"

We wanted to have a mature and professional relationship with our publisher. By exposing the ugly truth as problems came our way, and strongly defending our points with solid plans on how to mitigate the issues when challenged, we were building a relationship based on trust and commitment. We showed that we were in control and believed in what we were doing.

I think the publisher respected our dedication and passion, and in the end, despite all their concerns, they supported us in a very professional way. It has been a fantastic collaboration!

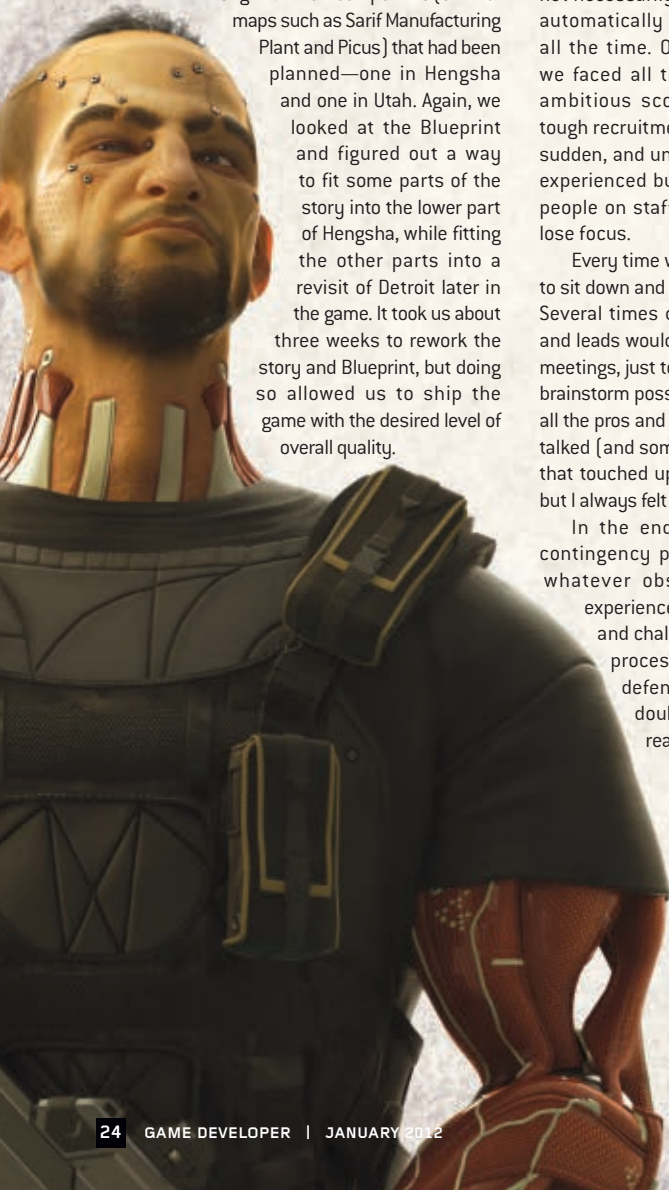
5/ internal playtest department.

Developing a game such as DEUS EX: HUMAN REVOLUTION is really complex, and from time to time, being close to it makes you lose perspective on what is or is not important. This is where having an internal playtest infrastructure becomes really useful, by helping you validate where your design is or isn't working. User feedback is one of the strongest reality checks you can get.

Unfortunately, at the beginning of the project, Eidos-Montréal did not have an internal playtest department. We had to travel to Europe and the U.S.'s West Coast to observe players. Not only did this make playtests expensive, but they were logistically more complicated to organize: we needed to plan way ahead to book the locations and playtesters, often long before we were sure we'd have a robust version of the game ready to make the playtests worth our time. Because of all this, it was hard to conduct them on a regular basis. We definitely lacked the flexibility needed to make the playtest process valuable in the long run.

However, as most developers know, a game only comes to life in the last year or so of development (presuming your development lasts more than a year). At the beginning, we had managed to do a few playtests to validate the high-level intentions and basic gameplay mechanics that were in place. In the last year, the game became fully playable, so it was finally possible to do an entire walkthrough. This was also when we started to really balance the game.

Luckily, the Eidos-Montréal studio had recognized the inherent value of building an internal playtest department early on. They'd



begun hiring experienced people in the field so that, by the time the HUMAN REVOLUTION team reached its last year of development, we were able to use the new department to its fullest. We were able to schedule several playtests and, if I remember correctly, conducted over 15 of them.

The important thing about playtests is to know your objectives ahead of time. We weren't interested in subjective stuff like "I want more guns." We were interested in seeing where people would get stuck. Were they able to navigate the game easily? Did they understand the story, and were they getting involved in it? Were there any balancing issues? Did they understand the gameplay's flexibility?

We did have some issues with most of the above topics in the first few playtests. However, because the team could watch players struggling with one aspect or another in real time, we could quickly take in the problems at hand, thus reducing the number of arguments and enabling us to focus more on solving the issues.

Having a playtest department in-house also allowed us to book it often with builds that contained the fixes of prior sessions to see if our changes worked. We balanced the game and tested the full walkthrough with players more than once, too. We even conducted a playtest at the end of May 2011, resulting in one final balancing pass a few days before locking the game down to prepare our Gold Master candidate.

This kind of last-minute fine-tuning would not have been possible if we didn't have an internal playtest department. Of course, playtests don't give us answers for everything (see Boss Fights under "What Went Wrong"). Make no mistake: we as developers need to maintain our vision and judgment to determine when to listen to consumers. We do not want to end up with the Homermobile.

Playtests can really help with feedback on important details here and there. They can show us details and issues that can be addressed easily, making a world of difference to users without compromising our vision.

WHAT WENT WRONG

1/ boss fights.

When we started the project, we knew quite early on that we would include boss fights. The goal was to spice up the rhythm of the game with more exotic sequences. We also wanted these encounters to celebrate and expand on some of the game's core gameplay pillars: choices and consequences, combat vs. stealth, and lethal vs. non-lethal play styles.

Our ambitions were sky high, but the reality didn't match what we were trying to put in motion. Although we did raise the flag early on that a team dedicated to bosses would be needed, the project as a whole was so ambitious that all resources were focused on establishing the core gameplay mechanics and underlying systems that would

serve as a foundation for the entire experience. We didn't have a huge team, and this fact caused headaches on several occasions, especially in the first two years of the project.

So, for a while, boss fight designs didn't advance. We had identified our enemies. We had determined the philosophy behind each encounter. Nevertheless, about a year into the project, no progress beyond these design points had been made.

When we finally realized our lack of progress, we were too caught up in other aspects of the game to give boss fights the attention they needed. We shuffled people around and named a "boss fight" owner. Unfortunately, we were still being more reactive than actually thinking things through at this point. As a result, we put an inexperienced level designer in the position without giving him the programming and animation support he needed. He also wasn't given the proper support from management. So basically, we had created very unsteady groundwork for success.

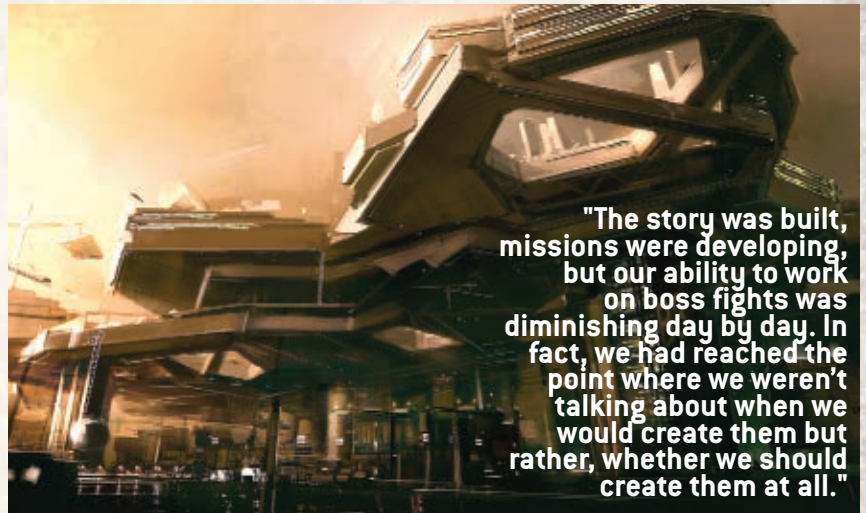
It took us a few months to realize things

scenario and needed other options.

We managed to find a company based in Montreal that specialized in AI and was willing to take on the contract to create our boss fights. Still, since we were already so late, our schedule was very aggressive, forcing us to "re-scope" the boss fights into something way more typical and inconsistent with the rest of the game. It was a tough decision, but we decided to move forward with it anyway. The company we hired would produce two of our boss fights while the other two would be completed in-house.

We weren't at the end of our problems, however. Internally, we had a new AI team. The new team redesigned several fundamental systems, making the outsourced bosses mostly incompatible with what was running in the game. When we decided to approve the new AI philosophy, we hadn't investigated the impact it would have on the outsourced bosses. Consequently, we were forced to scale back the scope of our boss fights (once more), until they became somewhat glorified versions of our new enemy AI system.

Toward the end of development, we had some



"The story was built, missions were developing, but our ability to work on boss fights was diminishing day by day. In fact, we had reached the point where we weren't talking about when we would create them but rather, whether we should create them at all."

weren't progressing the way they needed to, and that the problem was the result of several factors. When we did, we removed the person in charge, but didn't assign anyone else to take over. We were too occupied with other important aspects of the game and simply didn't have anyone to spare. Thus, after two years in production, we only had some of the basics for one out of the four boss fights designed.

Time passed, the story was built, missions were developing, but our ability to work on boss fights was diminishing day by day. In fact, we had reached the point where we weren't talking about when we would create them but rather whether we should create them at all. The problem was that cutting the fights would have impacted other aspects of the story, essentially putting the risk somewhere else. So, cutting them didn't make any sense. We were stuck in a "lose-lose"

free time to work on bosses and managed to still make them fun; they changed the pace of the game experience and forced players to use the systems and tricks they learned while playing the game. Playtests also confirmed they were still fun.

(As an aside, when I say bosses were still fun, I'm referring to the fact that the feedback we received from playtests before the game's release indicated this. At that point, we knew that we were inconsistent design-wise, but we didn't anticipate the upcoming backlash we would get upon the game's release, which stated the bosses were frustrating.)

In the end, we didn't address the boss issue early enough. Then, when we tried to fix things, we were pulled in so many other directions that we couldn't clearly see the full impact of our decisions. Had we managed to think through all those steps,



bosses would have been way more consistent with the game experience and, more importantly, would have brought a level of novelty to the game I feel is missing.

2/ cutscenes.

DEUS EX: HUMAN REVOLUTION is a conspiracy-laden, story-driven game. Early on, we knew we would have to rely on several story delivery methods to impart what we had in mind. One of those was the use of cutscenes. After completing the story and blueprint, it was estimated that we could afford close to an hour of cutscenes. That really forced us to make sure the right narrative pieces were being created in each scene.

One thing was really clear: we wanted cutscenes to be rendered in real time in the game engine. We wouldn't rely on pre-rendered stuff because it creates a clash with the actual in-game rendering, and we wanted cutscenes to flow seamlessly with actual gameplay sequences.

Despite the fact that we were, overall, an experienced team, we lacked the experience to realize that the number of people we had planned to take care of cutscenes was insufficient. As we moved forward, we began to understand that our tools didn't allow us to work efficiently, but we didn't have enough experience creating real-time cutscenes to understand what tools we would need. We also didn't have the internal resources

to allocate to the creation of efficient tools that might help the cutscene department.

Two and a half years into the project, we still had no "final quality" cutscenes implemented in the game. All we had were some early versions of the least complex scenes, such as choppers landing and taking off.

We knew for a while that things weren't going in the right direction, but I guess our lack of experience on that level made it harder for us to recognize how much trouble we were in.

The undisputed truth about our situation was about to be revealed to all as we were racing to deliver our 2010 E3 demo. We had three cutscenes planned for that demo. In terms of length, those three scenes counted for about 1/10th of what we needed to produce for the entire game. It took us roughly four months with everyone working in the same direction to complete them, and since a lot of the team was dedicated to creating the E3 demo, working on the cutscenes gave us a good sense of what it would take to ship the game with everything in place.

The hard reality we faced was that while we were able to produce really high-quality in-game cutscenes, we had less than a year to produce the rest of them. It was just impossible.

It was the 11th hour and we had to knock on the doors of our good friends at Vancouver's Goldtooth studio (the company behind the CGI

trailer concept). We built a small internal team to evaluate everything it would take to have Goldtooth produce our material, and we discovered that the only way to complete the cutscenes in time would be to drop the in-game cutscene philosophy and go back to classic pre-rendered ones. It was a disappointment for us, but we were at a point where it was necessary to make concessions.

In the end, even with this new approach, our schedule was more than aggressive, forcing us to compromise on some aspects. Ignoring the fact that outsourcing poses its own challenges, it was the best decision we could've made. We learned quite a lot, and I'm confident that we won't make the same mistakes again in the future!

3/ shared technology and an initially unrealistic development cycle.

When we started Eidos-Montreal, our original mandate was to revive the DEUS EX franchise in 24 months. We had never made a DEUS EX game back then, but we knew it wouldn't be finished in two years. Even so, we were making technological decisions based on that initial ambition. At the time, Eidos's philosophy was to support shared technology. Our best choice was to make a partnership with Crystal Dynamics. The plan was to have Crystal make improvements on the engine that would fit DEUS EX: HUMAN REVOLUTION's needs.

Basically, they would be the providers and we would be the clients. At the time, it seemed like a smart plan that would allow us to focus on content instead of focusing on developing the technology.

We built our first technical demo during the collaboration between the two studios, and things went rather nicely. However, despite the fact that the initial demo helped us validate the foundations of our pipeline and our collaboration, due to our ambitious vision, specific needs quickly started to pile up.

The problem was that Crystal Dynamics was also working on a project. Eventually, their needs and ours started to veer off in different directions, making it harder and harder to get the features we needed completed in time, since they had to split their efforts between the projects.

We were almost two years into DEUS EX: HUMAN REVOLUTION and were losing so much time because our tools and the shared technology pipeline weren't keeping up with the challenges we faced. We had to admit the obvious: we couldn't continue the way we were without compromising the delivery of the game.

Fortunately, Eidos was really committed to reviving the HUMAN REVOLUTION franchise and backed our decision to branch off from Crystal and go on our own with the engine. By then, it was clear for all of us that we would never ship the game in a two-year time frame. We were too far behind.

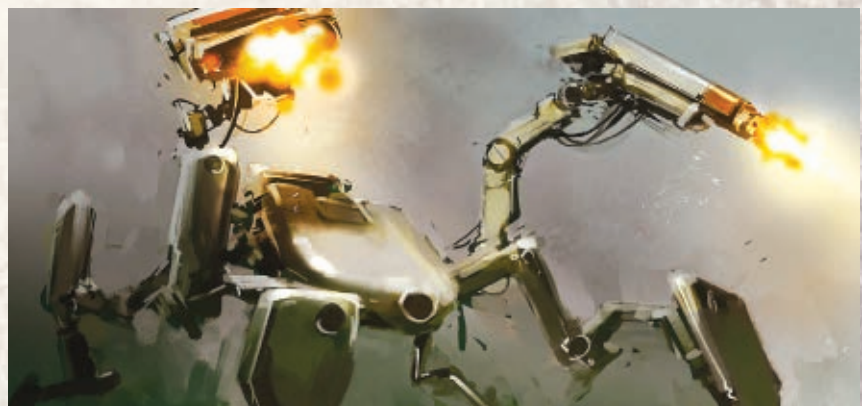
By branching off, we were able to improve several tools for the production floor, adding systems such as deferred lighting to the engine, which allowed us to push things in the direction that fit our ambitions and needs. Shared technology is a beautiful idea, but when it's built around projects that are so different in nature, it's hard not to penalize all projects involved.

We started in good faith with good intentions. We suspected that what we were doing might have been based on unrealistic deadlines, and had we voiced those concerns more loudly at the time, we might have avoided headaches. At the same time, had we built a less ambitious concept to fit the original timeline constraints, we might have been able to make the shared technology work and ship the game on time, but it wouldn't have been the DEUS EX we know today.

4/ harder than anticipated to recruit the right experts.

Even with a generally experienced team, we sometimes struggled to find the right people, especially in key areas like animation and UI. Unfortunately, our inability to find the right people at the start of the project to some extent impacted the quality of these areas in the final product.

Early on, we knew that DEUS EX: HUMAN REVOLUTION would be a game about scope and ambition in all respects, and animation was no different. We wanted to have living, breathing city hubs populated with a lot of non-playable





characters. We also wanted to have a very advanced dialog system, which eventually became known as our “social boss fights.” (For those who played the game, talking to Wayne Haas in the police station is a prime example.) On top of all this, we wanted to have fully fleshed-out combat and stealth experiences.

To sum it all up: we wanted a bit of everything.

For the first part of the production, we did have a lead animator taking charge of these things, but as we moved forward with very ambitious goals, it became clear that he didn't share the same vision for the project. Eventually, both sides agreed to peacefully part ways.

Then we started looking for the best animation director we could find. We had very high requirements, however, and didn't find someone right away. In the meantime, we had an animation team working without a director. The art director and I tried to fill in and compensate as much as we could, but we couldn't really replace an animation director. In the end, having a lack of clear direction in that department for a long time really hurt our progress.

With less than two years to go on the project, we finally found the right guy. Everything landed on his shoulders at that point, and given that he couldn't save everything, I must say he did everything he could to fix the situation by nailing Adam Jensen's takedowns, the social boss fights, and the AI patrol cycles, among other things.

In our long quest to find all the right people, we also had a hard time finding the right UI artist for the game. We knew we'd want to push augmented reality through the HUD and we'd have tons of menus. We didn't find that many candidates, and to make matters more challenging, we were very picky when evaluating resumes.

We eventually found a very talented artist who blew us away. His only drawback was his total lack of experience leading a UI team. But it was hard to find quality candidates, so we decided to move forward with him anyway, well

aware that it was a risk.

The project was now well underway, and we had a more inexperienced lead in the middle of a strong-headed team of old-timers. In retrospect, I think he was intimidated, and since we were quite busy—a lame, but accurate excuse—we failed to support him properly. He eventually left, leaving us with a new problem: after roughly two years, we were still struggling to nail our UI direction, and now we had to look for a new UI director. Again, it wasn't easy.

Sometimes you look too far when the answer is right next to you. We eventually found a guy we knew in Montreal who accepted the challenge. Similar to the animation director, the new UI director had to deal with a lot of pressure, but he had the experience needed to give a clear direction. I must say, what ended up in the final game is pretty good.

In a sense, finding the right experts turned out to be a search for ourselves. It made us realize what kind of team we were and what was important to us. We were a very demanding team and not all profiles fit in with us. We were lucky enough to find the right warriors when we still had some time. The good news is that at that point, we knew what we were looking for.

5/ outsourcing.

Early in the game's development, we knew we would have a lot of content to produce and that we would have to explore the possibility of outsourcing. This was new to us, but luckily Eidos had a studio dedicated to subcontracting: Eidos Shanghai. After discussions with our friends in Shanghai, everything seemed much clearer: we would not be able to proceed with the same type of structure we had been using for in-house production. Even so, we were forced to move forward and try to do our best.

We proceeded by following standard steps to first determine what kind of content we needed to produce externally. This was based on

availability of resources versus the overall scope of our game. It appeared that we had to outsource the modeling and textures of thousands of objects and characters. Later in development, for various reasons, we also had to subcontract animations, boss fights, cutscenes, and some code.

Thinking back, I wonder how we managed to handle it all...

Our approach was fairly standard, though. We determined approval and communication workflows, organized internal reviews, and so forth. I won't go into all the details because I want to jump directly to where outsourcing hurt us the most: setting up an internal team structure to oversee it.

If outsourcing content can reduce risks and/or problems of recruitment and studio space, it also brings a lot of tasks and actions that are essential to complete. In other words, it's not free of challenges.


People without much knowledge of production think of subcontracting as a miracle that magically “gets stuff done,” enabling their team to concentrate exclusively on internal work. WRONG!!!!

Subcontracting involves setting up a team that's dedicated to monitoring tasks, ensuring approvals, planning deadlines, and carrying out constant communication with the outsource studio. You also need to consider sending internal teams off-site to train staff, maintain quality, and filter work that is sent back to your studio for approval.

We were never able to establish a structured team that would allow us to manage the work of 30 external artists, 20 external animators, and other external programmers. We were never able to give the Shanghai team a clear mandate because we lacked the bandwidth to manage them properly, and we had to conduct additional overtime to compensate for the work that didn't meet our expectations. For these reasons, outsourcing was not what we hoped for in the development of DEUS EX: HUMAN REVOLUTION.

revolving humans

In conclusion, building DEUS EX: HUMAN REVOLUTION was not always a smooth ride—far from it. We hit walls, we learned, and we bounced back several times. Our ability to adapt to ever-changing situations made us see the project through to the end. This is the most ambitious project that most of us have ever embarked on, and it made us grow as a team, as people, and as professionals who take their craft seriously.

Most importantly, three constants remained throughout development: our passion for this project, the love we put into it, and our commitment to make the best possible game. Despite the fact that DEUS EX: HUMAN REVOLUTION is not a perfect game by any means, we like to believe it has a soul. I hope you will feel this too when playing the game. 

JEAN-FRANÇOIS DUGAS was the game director on DEUS EX: HUMAN REVOLUTION. This post-mortem was written in collaboration with Martin Dubeau (project manager), David Anfossi (producer), and Mary DeMarle (lead writer).




UNREAL ENGINE NEWS

UNREAL ENGINE 3 ON iOS: A RISING TIDE LIFTS ALL THE BOATS

When Epic Games began licensing Unreal Engine 3, the thought of mobile hardware running such advanced technology was merely a dream. Over the past year, the reality of this aspiration has been demonstrated over and over again. The release of the engine's best-in-class OS features, especially through the wide availability of the Unreal Development Kit, has enabled developers – including our own studio, ChAIR Entertainment – to reap the rewards of shipping high-quality games for Apple's world-class gaming platform.

In December, ChAIR Entertainment released *Infinity Blade II*, the successor to the bestselling App Store mega hit. At the time of this writing, *Infinity Blade II* is enjoying critical acclaim, with more than a dozen perfect review scores from select media. Within hours of its release, *Infinity Blade II* became the number one paid and number one grossing App in the App Store worldwide. The first *Infinity Blade* game has enjoyed long-lived success, even reappearing on the list of the top 10 grossing Apps a full year after launch.

In addition to *Infinity Blade's* success, full source UE3 licensees and UDK developers have recently emerged victorious on the App Store with commercially successful games for iOS.

Fans of *Batman: Arkham City* hoping to experience its stellar visuals and gameplay on iOS need to look no further than *Arkham City Lockdown*. Published by Warner Broth-

ers and developed by NetherRealm, the same talented UE3 studio behind *Mortal Kombat*, *Arkham City Lockdown* is an incredible-looking brawler in which players deliver powerful combat moves via touch and swipe controls.

Phosphor recently released *The Dark Meadow*, which immediately cracked the top 10 iPhone and iPad sales charts, launching as an App Store Game of the Week. Developed with UE3, *The Dark Meadow* is a gorgeous action game that combines exploration, dark humor and horror.

In addition, UpperCut Games has captured the fun of robot combat on touch-screen devices with *EPOCH*, which launched in November as Game of the Week on the App Store, as well. Media outlet SlideToPlay.com even crowned *EPOCH* its Game of the Month, saying, "UpperCut Games has delivered a stylish, original shooter, starring a nimble, lovable robot that's armed to the teeth."

Praised for its steampunk visuals and fun multiplayer options, Emotional Robots' *Warm Gun* was developed entirely with UDK and is available on the App Store.

Another gorgeous game, *Desert Zombie: Last Stand* for iOS, is quickly gaining popularity. Crystallised Production Manager Cam Phillips remarked of the free UE3 toolset, "UDK is an industry leading, AAA games engine that's been made accessible to smaller development houses with a royalty-based licensing deal. No other engine on the market can compete with UDK on mobile platforms."

UE3 CONTINUES TO EVOLVE

Even in the midst of exciting product launches such as *Infinity Blade II*, Epic's engineering team continues to

make strides with enhancements to the Unreal Engine.

UE3 now ships with Scaleform 4.0, and includes support for both ActionScript 2 and ActionScript 3.

Scaleform 4.0 makes it easy to create menus, UI elements and 2D graphics for modern 3D games. UI designers can quickly become productive using familiar and mature tools such as Adobe Creative Suite.

Full source licensees have access to Scaleform 4.0 for PC, Xbox 360 and PlayStation 3, and Scaleform 4.0 features are available for PC to all UDK users as well.

In addition, Epic is bringing Scaleform 4.0 functionality to iOS, which will equip UE3 developers with powerful SWF rendering for their iPhone, iPad and iPod touch games.

WWW.UNREAL.COM



Canadian-born Mark Rein is vice president and co-founder of Epic Games based in Cary, NC. Epic's Unreal Engine 3 has won Game Developer magazine's Best Engine Front Line Award seven times, including entry into the Hall of Fame. UE3 has won four consecutive Develop Industry Excellence Awards.

Epic is the creator of the mega-hit "Unreal" series of games and the blockbuster "Gears of War" franchise. Follow @MarkRein on Twitter.

UPCOMING EPIC ATTENDED EVENTS

D.I.C.E. Summit
Las Vegas, NV
Feb. 8-10, 2012

GDC 2012
San Francisco, CA
March 5-9, 2012

Gadget Show Live
Birmingham, UK
April 11-15, 2012



Please email licensing@epicgames.com for appointments

WE'VE GOT PHYSICS BY THE BALLS



havok™ Physics

The Proven Leader in Middleware Technology.



2010 2009 2008

Three-time consecutive winner of the Game Developer Front Line Award for best middleware.

Thank you to all our customers and fans who voted for Havok Physics and Havok Vision Engine for the 2011 Game Developer Front Line Awards!

Learn more: www.havok.com

Havok™ Middleware Solutions Include

Havok Physics • Havok AI • Havok Animation • Havok Behavior • Havok Cloth • Havok Destruction • Havok Script • Havok Vision Engine

GDC 12

P R E V I E W

The Game Developers Conference in San Francisco (March 5–9, 2012) reflects not only the current state of the industry, but also shows hints of where it's going. While the main conference showcases the best of the more traditional industry—big-budget console and PC games, MMOs, and high-end social—the summits dive deep into specific areas of game development that might not otherwise get enough attention.

The big trend this year seems to be a familiar one for later in the cycle of game platforms. Now that the PlayStation 3, Wii, Xbox 360, iOS, motion control, and social platforms have been around for several years, talks seem to be more about extracting the absolute optimal performance from each, and solving problems that were previously brushed under the rug in a mad dash to break into the various spaces now available to developers.

Meanwhile, on the business side, the idea of free-to-play and subscription models for console and PC games is no longer a “what if.” The reality of the changing business is now more about best practices than new scenarios, though it does seem as though a variation on popular business schemes crops up every other month.

As a teaser before the show, we've highlighted a number of interesting talks from both the mainline GDC and the summits (see page 34), with more talks to be added frequently at www.gdconf.com. See you at GDC!



AI-DRIVEN DYNAMIC DIALOG THROUGH FUZZY PATTERN MATCHING. EMPOWER YOUR WRITERS!

ELAN RUSKIN (VALVE)

/// "Pills here!" "There's a zombie behind you, Bill!" Characters that understand and remark on their circumstances add a lot to a game, but as state and dialog possibilities grow, selecting one line out of thousands with an immense if/else script becomes daunting. This talk details Valve's system for using hundreds of facts about its game worlds in a fuzzy pattern match against a database of thousands of possible lines to create responsive, dynamic dialog in LEFT4DEAD and other titles. The talk will show a simple, uniform mechanism for tracking thousands of facts and possibilities, allowing intelligent characters to remember history, cascade from special to general cases, and select the optimal dialog, script, behavior, or animation for every situation. Most importantly, a friendly interface provides writers creative freedom to make special cases, running gags, and track additional facts without forcing programmers to change thousands of lines of code.

ARRRGHH!!!—BLENDING QUANTITATIVE AND QUALITATIVE METHODS TO DETECT PLAYER FRUSTRATION

JANUS RAU SORENSEN (CRYSTAL DYNAMICS / IO INTERACTIVE)

/// Frustration is an integral part of game experiences (and life in general). But some types of player frustration are just plain bad, and can lead to players feeling powerless and even angry. This talk describes a computational model that can identify patterns of behavior indicating unwanted player frustration. The model is based on player behavior logged through gameplay metrics from the game KANE & LYNNCH 2: DOG DAYS. The framework for the method used is at least as interesting as the results: It was obtained by methodological triangulation, using both qualitative and quantitative methods (e.g., observation, interviews, and data mining). Apart from game developers being able to establish automated frustration detection systems, the model could also theoretically be used as an element in adaptive gameplay design.

ATTENTION AND AGENCY: MAKING YOUR GAMES BETTER USING METRICS, THE UNCHARTED WAY

RICHARD LEMARCHAND (NAUGHTY DOG)

/// We all know metrics can be useful in social games, but what about a triple-A title like UNCHARTED 3? Lead game designer Richard Lemarchand will lead us through the use of metrics in the latest UNCHARTED title, showing how the team mined valuable data from playtests, and applied that to a game with a huge world, above and beyond the simple "users will buy this or that" tactics we've seen in the social realm.

THE AUTOMATION TRAP AND HOW BIOWARE ENGINEERS QUALITY

ALEXANDER LUCAS (BIOWARE)

/// Test automation is received with great enthusiasm by many in the game industry. Yet, given many of the unique challenges faced in game development, is it really a useful tool for creating quality games? Using real-world examples from the development of the DRAGON AGE and MASS EFFECT series of games, this talk examines the perceived vs. actual return on investment of test automation, as well as its potential pitfalls. More specifically, this talk hopes to show that many of our base assumptions, such as the idea that manual testing can be effectively reproduced with automation, aren't necessarily true.

The lecture will also look at some of the sensible approaches to automation and test-technology that BioWare has employed. This includes

tools-assisted-testing that augments the testing process and turns all testers into rock stars, and applying automation at earlier stages of development to prevent defects outright, rather than just detecting them.

CINEMATIC GAME DESIGN IV: CHARACTER & EMPATHY

RICHARD ROUSE III (UBISOFT MONTREAL) MARTY STOLTZ (BIG HUGE GAMES/38 STUDIOS)

/// More than any other medium, games have the potential to let players walk a mile in someone else's life. But for players to be truly immersed and see through someone else's eyes, players must empathize with the player character. Films faced a similar challenge, and over the last hundred years have developed numerous cinematic techniques that help define characters and make audiences empathize with even the most unlikable personalities.

In the latest installment of the popular Cinematic Game Design GDC lecture series, the focus is on the issue of defining character and strengthening audience/player empathy for that character. A series of film clips that demonstrate specific cinematic devices for manipulating audience empathy will be shown, with each technique analyzed and assessed to see how it could be applied to a game. Clips from games that have used variations on these techniques will also be shown and discussed. The talk is not just about cutscenes, but how these cinematic techniques can be applied to gameplay.

COMMUNICATION AND ITERATION OF VISUAL EFFECTS IN RED FACTION: ARMAGEDDON

DAVE SAMUEL (VOLITION)

/// When we think of visual effects in games, we tend to think of fire and pretty explosions. In this talk, Volition's senior visual effects artist Dave Samuel explains the communication and process of iteration of visual effects. The talk goes over specific art and design problems in RED FACTION: ARMAGEDDON that were aided by visual effects, and offers a look into the iterative process that led to the game's effects looking as good as they did.

DEDICATED SERVERS IN GEARS OF WAR 3: SCALING TO MILLIONS OF PLAYERS

MICHAEL WEILBACHER (MICROSOFT STUDIOS)

/// If you've ever wondered why games use dedicated servers for FPSs, or how much client and non-client code is needed to achieve such a goal, or what the infrastructure requirements are, this talk is your chance to see how GEARS OF WAR 3 conquered the problem.

The presentation will give an overview of the dedicated servers that were hosted in multiple data centers. The talk will touch on factors that impact engineering decisions, like cost, latency, and what changes are necessary in terms of client and code infrastructure for hosting over a thousand servers.

DEVELOPING IMPERFECT SOFTWARE: HOW TO PREPARE FOR DEVELOPMENT PIPELINE FAILURE

RONALD PIEKET (INSOMNIAC GAMES)

/// Computer game software development, more than any other kind of software development, relies heavily on the usage of software that is itself in development. Software that is in development is inherently unstable. Often the entire content creation team depends on this imperfect software for their daily workflow. And when it fails, people cannot do their work. The larger the team, the greater the impact on productivity.

Much effort goes into preventing software failure, especially if a large team relies on it. But developers must understand that failure is a natural

part of software development, which means they cannot rely on prevention alone. We must have a strategy to manage software failure. This session will discuss what happens once a failure has occurred, and how developers can make sure that everyone can keep working. How can we make our development software "fail well?"

THE FIVE DOMAINS OF PLAY: APPLYING PSYCHOLOGY'S BIG FIVE MOTIVATION DOMAINS TO GAMES

JASON VANDENBERGHE (UBISOFT)

/// Over the last 20 years, modern motivational psychology has coalesced around a system called "The Big 5" (or OCEAN). Why should we care? Because unlike its predecessors, this idea has a titanic landslide of repeatable, scientific evidence behind it. But how does it apply to games?

Jason VandenBerghe, an Ubisoft creative director, been collaborating with academics and industry colleagues to try to answer that question—they have found that several specific game elements are directly to measurable Big 5 personality facets. By doing this, designers can uncover a completely new way of looking at the motivations of play.

THE GAMIFICATION OF DEATH: HOW THE HARDEST GAME DESIGN CHALLENGE EVER DEMONSTRATES THE LIMITS OF GAMING

MARGARET ROBINSON (HIDE&SEEK)

/// GDC is the place we come to get excited about the potential of games. But what are the hard limits to how far games can go? In what might be the hardest game design challenge ever, UK studio Hide&Seek was asked to make the game of a documentary about the discovery of a corpse in a busy London block of flats: a woman had lain dead for three years without being found, without ever having been reported missing. She was 37 years old.

How does someone fall through the cracks, despite having family, friends, and neighbors? And how do you make a game about a real woman's death without producing something crass, simplistic, or libelous? This is the story of how the team tried and failed, and of the fundamental contradictions that they discovered at the heart of gamification. (The team promises it won't be as grim as it sounds.)

GIANT TOADS AND ZOMBIE BEARS: TECHNICAL ART RE-ENVISIONED FOR DIABLO 3

JULIAN LOVE (BLIZZARD)

/// From start to finish, DIABLO 3's technical art team challenged the notions of the traditional technical art role. This talk focuses on the motivations, philosophies and values that drive the DIABLO 3 technical art team. Examples of how this approach impacted DIABLO 3 will be covered, including key contributions to combat responsiveness, the game's over-the-top feel, the D3 class skill design process, and how the team anticipated the project's scripting and special effect needs, then transformed itself to meet them.

HOW I GOT MY MOM TO PLAY THROUGH PLANTS VS. ZOMBIES

GEORGE FAN (POPCAP)

/// The final boss of PLANTS VS. ZOMBIES has been conquered by gamers of all ages as well as people with no gaming background whatsoever. In this lecture, George Fan, designer of PLANTS VS. ZOMBIES, will walk you through how he managed to get his mom to play through a full-blown strategy game. He will present 10 techniques he used to better teach game mechanics to players, using specific examples from PLANTS VS. ZOMBIES to illustrate these concepts.

INTRINSIC & EXTRINSIC PLAYER MOTIVATION: IMPLICATIONS FOR DESIGN AND PLAYER RETENTION

SCOTT RIGBY (IMMERSYVE)

/// Based on years of research, this session explains in detail how to sustain engagement with players in order to drive longer and more profitable relationships with them. To design for this kind of engagement, data shows it is important to have a clear knowledge of a player's specific psychological needs and understand the various categories of motivation that lead to sustaining (or thwarting) customer relationships. This talk will outline principles of both intrinsic and extrinsic motivation, and how both can be applied effectively when they satisfy specific psychological needs, such as autonomy, relatedness, and mastery. Many game examples will illustrate some surprising truths about motivation, including how certain reward mechanisms can actually hinder a long-term relationship with games, as well as how various forms of "extrinsic" motivation can be good for the player relationship, while others can foster a hasty exit by creating feelings of control and manipulation.

NO BROKEN BUILDS: INCREASING TEAM VELOCITY WITH PREFLIGHT

DAVE SCHAEFER (BIOWARE)

/// Decrease your broken builds by 99%! Submit your work and go home! Through this talk, you'll learn how automated preflight tests can help your team iterate quickly and with confidence by preventing errors. This presentation details how BioWare revamped its pipeline for DRAGON AGE ORIGINS and DRAGON AGE 2, letting programmers easily and automatically test their work to keep progress moving and builds flowing. Learn how to build your own preflight system to help your team's project take off!

PRODUCTION CULTURE: HOW TWO AND A HALF ARTISTS COMPLETED THE FX FOR GEARS OF WAR 3

FRANCOISE ANTOINE (EPIC GAMES)

/// In an time of ever-growing development teams, Epic prides itself on keeping its team as small as possible. In this presentation they will look at how two and a half artists managed to create FX for an entire AAA game, including cinematics. The focus will be on how Epic Games' "production culture" made this possible, and how deeply it reaches into all aspects of the company, from the hiring process to the software tools.

STABLE SSAO IN BATTLEFIELD 3 WITH SELECTIVE TEMPORAL FILTERING

LOUIS BA VOIL (NVIDIA)

/// With the highest-quality video options enabled, BATTLEFIELD 3 renders its Screen-Space Ambient Occlusion (SSAO) using the Horizon-Based Ambient Occlusion (HBAO) algorithm. For performance reasons, the HBAO is rendered in half resolution using half-resolution input depths. The HBAO is then blurred in full resolution using a depth-aware blur. The main issue with such low-resolution SSAO rendering is that it produces objectionable flickering for thin objects (such as alpha-tested foliage) when the camera and/or the geometry are moving.

After a brief recap of the original HBAO pipeline, this talk describes a novel temporal filtering algorithm that fixed the HBAO flickering problem in BATTLEFIELD 3 with a 1–2% performance hit in 1920x1200 on PC (DX10 or DX11). The talk includes algorithm and implementation details on the temporal filtering part, as well as generic optimizations for SSAO blur pixel shaders.

GDC hosts a variety of summits, which delve into specialized and emerging markets in the game industry. This year, the conference introduces the Game IT Summit and Games for Change @ GDC to the lineup, totaling eight summits in all. Below, we've outlined each summit and given you a taste of the key talks featured within them. New talks will be added regularly in the weeks leading up to the show, so be sure to keep an eye on www.gdconf.com for more.

INDEPENDENT GAMES SUMMIT

The Independent Games Summit represents the voice of the indie game developer at GDC. It features lectures, postmortems, and roundtables from some of the most notable independent game creators, including many former and current Independent Games Festival finalists and winners. The 2012 Independent Games Summit seeks to highlight the brightest and the best of indie development, with discussions ranging from game design philosophy, distribution, business, marketing, and much more.

FOLK GAMES, FESTIVITY, AND SUBVERSIVE GAME DESIGN

DOUGLAS WILSON (DIE GUTE FABRIK)

/// This talk encourages developers—especially indies—to embrace the limitations of motion control technology, rather than struggle against them. Drawing from his work on party games like B.U.T.T.O.N. and JOHANN SEBASTIAN JOUST, as well as a slew of offbeat case studies, researcher and developer Douglas Wilson describes how game designers might recapture the festivity and carnival atmosphere that have long been a hallmark of play culture. He argues that folk games offer useful precedents for developers working on physical games, and explains why he finds it fruitful to think about motion-controlled games in terms of slapstick and subversion.

LOCALIZATION SUMMIT

Localization is a cornerstone of the game industry, directly affecting a game's global revenue stream. Beyond the traditional game markets, the demand for interactive media is quickly increasing from emerging markets around the world, which is prompting publishers to partially or fully localize more products into more languages to maximize their ROI.

STARCRRAFT II—CARTE BLANCHE LOCALIZATION

WILLIAM BARNES (BLIZZARD)

/// When Blizzard launched STARCRRAFT II in July of 2010, it was lauded globally as one of the most thoroughly localized games ever. Jim Raynor enunciated perfect Korean, Dr. Hanson's French was sublime, and Tychus Findlay spoke flawless Russian. The multiplayer unit VO was every bit as entertaining in Italian as it was in English. In this lecture, Blizzard's William Barnes will pick apart the studio's approach to localization, and will explain why the team elected to localize STARCRRAFT II as comprehensively as it did.

AI SUMMIT

The AI Summit at GDC features panels and lectures from more than two dozen of the top game AI programmers in the industry. This event, which is organized as a collective effort by the AI Game Programmers Guild, promises to give you an inside look at key architectures and issues within successful commercial games, and will let attendees eavesdrop on conversations, debates, and rants about how game AI can move forward.

Some of the latest and greatest techniques will be on display in the AI disciplines of navigation, pathfinding, animation control, behavior selection, group tactics, and strategy. There will be sessions on AI tool development and the issue of crafting scalable AI architectures. Also, for the first time, this year's summit will introduce a session on the use of machine-learning

techniques—historically a taboo subject in game AI, but one whose time may have come.

This year also brings the return of two favorites; the AI postmortems and the rant session. As always, the postmortem session will feature presentations on recently released titles with the common theme of "challenges faced" (and overcome!). For example, Michael Dawe (Big Huge/38 Studios) will lift the AI curtain of the much-anticipated game, KINGDOMS OF AMALUR: RECKONING. The AI devs' rant session, while traditionally on the lighter side, always provides nuggets of wisdom in tiny, often humorous packages.

While the AI summit is targeted toward the intermediate-to-advanced programmer who wants deeper insight into the world of game AI, anyone who is interested in what AI can offer next-generation games will find invaluable insights and lessons from the speakers.



PHOTO COURTESY OF GAME DEVELOPERS CONFERENCE

GAME IT SUMMIT

The Game IT Summit is a daylong conference focused on the use of video games to tackle common organizational goals through enterprise-focused game development. As the power of play, video game design, and social systems meet up with ever more powerful browser-based application stacks, organizations are looking to create new ways to boost productivity, improve customer interactions, and take advantage of the disparity between the perceived power of video games versus traditional web and IT approaches to UI, engagement, and collaborative interactions. This summit addresses and facilitates that crossover.

APPLYING GAME AND SOCIAL MECHANICS TO SUSTAINABLE FASHION—CASE STUDY OF CLOSET SWAP

PAULINA BOZEK (INENSU)

/// We've seen more and more products and services look to games for inspiration of late, and this talk will illustrate one example by looking at the fashion app Closet Swap—an online fashion community that helps teens

swap clothes with their friends. While the app itself is meant to affect real-world relationships and styles, Closet Swap makes use of a game and social layer to influence behavior in a natural way—specifically, by celebrating personal style over disposable fashion and getting users to swap clothes, not shop for new ones. This talk will look at the design process, audience research and testing, and how game and social mechanics were employed to create an experience that is entertaining and authentic in its purpose for promoting sustainability.

GAMES FOR CHANGE @ GDC

Games for Change facilitates the creation and distribution of social impact games that serve as critical tools in humanitarian and educational efforts. Games for Change @ GDC represents, for the first time, an opportunity to bring together those funders, NGOs, government agencies, and educators seeking to leverage entertainment and engagement for social good with leading game developers from the independent and commercial sector. Through case studies, lectures, and demos, Games for Change @ GDC will highlight models for collaboration on game design, bridging the gap between commercial and issue-driven game development, distribution, and publishing alternatives, and much more.

MORE THAN FUN—DESIGNING GAMES WITH PURPOSE

PHIL STUART (PRELOADED)

/// While more often than not we think of games as entertainment, there exist a number of ways to use games for purposes other than just having fun. In particular, this talk explores the challenges faced when creating games with a specific educational or communication objective. Stuart will examine the balance between a game's content and its mechanics, breaking down how the two interact to create meaning for the player. In addition, he will explore the basic principles of education-focused design, and how to engage a target audience, regardless of a game's purpose.

GDC EDUCATION SUMMIT

At the 2012 GDC Education Summit, attendees will explore experimental and inventive educational approaches that established game curriculum builders can bring back to their faculty and classrooms. This program is aimed toward educators from established game development programs or new game course creators that want to understand the challenges they'll face in the next few years. It will bring scholars together with experienced professionals willing to learn and share ideas and achievements. The summit will ultimately explore how collaboration leads to success not only in the classroom but in all aspects of work and life.

WHAT IS A RESEARCH GAME?

NOAH WARDRIIP-FRUIJ (UC SANTA CRUZ) MICHAEL MATEAS (UC SANTA CRUZ) TRACY FULLERTON (UNIVERSITY OF SOUTHERN CALIFORNIA) ZORAN POPOVIC (UNIVERSITY OF WASHINGTON)

/// Research is a fundamental activity of higher education. But where does game making fit into the picture? Are games just a means of applying existing research techniques, or do games present a new area of expertise, introducing new questions and calling for new methods? This panel unites the directors of three very different game research groups. Each describes a recent project—and outlines why building complete, publicly released games is essential for their research. Together, they provide an arsenal for those considering making games within their own research, or simply needing to convince a skeptical dean.



PHOTO COURTESY OF GAME DEVELOPERS CONFERENCE

SMARTPHONE & TABLET GAMES SUMMIT

The Smartphone & Tablet Games Summit at GDC 2012 brings together top game developers from around the world to share ideas, introduce best practices, and discuss the future of gaming on established and emerging smartphone and tablet platforms, including iPhone/iPad, Android, Blackberry, Windows 7, and beyond.

SHADOW CITIES AND THE FUTURE OF LOCATION-BASED GAMES

MARKUS MONTOLA (GREY AREA)

/// SHADOW CITIES is among the world's first location-based MMORPGs. The game references players' physical locations, and allows them to interact with the game world via their smartphones. This presentation discusses the challenges the team faced when dealing with this sort of design. For instance, using the real world as your level design makes your game always inherently unfair. In addition, you play with your neighbors more than with your friends. In the landscape of the real world, the tried and true lessons of MUDs and MMORPGs no longer apply, and this talk explores how Grey Area dealt with these challenges.

SOCIAL & ONLINE GAMES SUMMIT

The market for interactive entertainment on social web platforms has matured rapidly, and now the features and tropes of connected games are catching on across the gaming landscape—encompassing Facebook games, web-based online games, downloadable persistent MMOs, casual titles, and more. The Social and Online Games Summit once again gathers the industry's established leaders and up-and-coming rebels for a series of illuminating sessions about the technology, design, business, marketing, and future of social and online games.

CHARACTER ASSASSINATION

KENNY SHEA DINKIN (PLAYDOM)

/// Zynga, Playdom/Disney, Playfish/EA, Wooga, Crowdstar—together these companies have reached over half a billion people and have arguably come to define the mass market for interactive play—but can you name a single well-known mass market character from these games?

In other game categories, characters like Sonic, Mario, and even Angry Birds have emerged as popular cultural icons. This talk will look in particular at characters within the sphere of narrative design for social games, and especially try to answer the question: What makes it so hard to create a compelling and beloved character in a social game?



Havok Moves East

AN INTERVIEW WITH DAVID COGLAN

GAME DEVELOPMENT IN CHINA HAS BEEN RAMPING UP IN A BIG WAY. Even five years ago, China was viewed primarily as a place to outsource art or animation work. But now, indigenous companies are making big strides, in original online game development. With a rise in game development, there's an inevitable rise in middleware providers trying to be first to market in the region.

We caught up with Havok managing director David Coghlan in Shanghai during GDC China, and asked about the company's strategy for the East, as well as Havok's ever-expanding empire, as it pushes its Vision engine, and makes partnerships with Autodesk.

Brandon Sheffield: What brings Havok here to China?

David Coghlan: We're really looking at it as a market with a lot of potential for the Havok portfolio of products. And, traditionally being console-focused, companies that we would've done business with in China would've been subsidiaries of the large Western publishers; people like EA, Ubisoft, and 2K. But increasingly we're seeing an interest from

Chinese indigenous publishers and developers in using the Havok technology. And one of the things that is sort of escalating that interest is the fact that we've recently acquired the Vision Engine from Tringy.

One thing we've seen in the Chinese market is that game teams are more interested in picking up the full engine solution, rather than piecemeal middleware where they still have to fill out a lot of the pieces around that. So with

the Vision Engine, I think we now have a bundle offering that includes everything you need to put together a game, and we see a lot of potential in the region over the next few years.

BS: Are these mostly social-oriented, or PC online, or what kinds of companies?

DC: They would tend to be very much the sort of 3D-based, MMO type titles. And also some titles that would have an online element, but

product news

ELDER SCROLLS V: SKYRIM Tools To Be Released
[WWW.BETHSOFT.COM](http://www.bethsoft.com)

Those interested in tinkering with the tools used to make Bethesda's just-released ELDER SCROLLS V: SKYRIM will soon have the opportunity, as the company says it will make these available to the public soon.

Publisher Bethesda says the game's official Creation Kit will become available for PC users in January 2012, alongside a Bethesda-organized Wiki intended to help modders become familiar with the new toolset.

In addition, the Creation Kit will include support for Valve Software's Steam Workshop, making SKYRIM the second game to support the service after TEAM FORTRESS 2.

The Steam Workshop will help users manage their in-game mods within Steam itself—modders will be able to upload their creations to the service, after which other users can browse, rate, and flag mods for download using any web-enabled device.

Microsoft Announces Kinect Accelerator Program, PC Peripheral
[WWW.MICROSOFT.COM](http://www.microsoft.com)

The Microsoft has announced the Kinect Accelerator program, which looks to support entrepreneurs, engineers, and innovators who want to leverage the Kinect hardware to implement a range of business ideas.

Developers are able to apply for the Kinect Accelerator grant now via the Microsoft web site, and ten finalists will be chosen early next year to receive the funding.

Finalists will receive \$20,000, an Xbox development kit, the Windows Kinect SDK and office space in Seattle, Washington, along with technical support and mentorship from Microsoft executives and investors. Each company will then be given the opportunity to present its ideas to Microsoft investors and venture capitalists at an Investor Demo Day.

Microsoft stated that it is looking for a wide range of

entries, including programs focusing on retail, education, health care, art installations, and games, as long as they can be turned into a commercial business.

Additionally, Microsoft will soon introduce a Windows PC-specific version of its Kinect peripheral, with updated firmware allowing the depth camera to see much closer than before.

According to the company, the new firmware allows the Kinect to see objects as close to 50 centimeters away without any accuracy or precision loss. A beta version of the Kinect Windows SDK is available now.

The company is calling the new feature "Near Mode." According to Kinect for Windows general manager Craig Eisler, the new mode "will enable a whole new class of close-up applications, beyond the living room scenarios for Kinect for Xbox 360."

GarageGames Releases Torque 3D Engine Version 1.2
[WWW.GARAGEGAMES.COM](http://www.garagegames.com)

GarageGames has released the latest version of its popular Torque 3D engine, adding new art, new tutorials, and new features for gameplay elements.

Version 1.2 provides a new demo environment, complete with building models, props, lighting variations, and additional documentation.

It also now includes a first-person shooter development tutorial, along with the ability for developers to separate first and third-person views. Additionally, the new version of the engine comes with a selection of new gameplay additions, including AI turrets, proximity mines, and teleporters.

This new version is available to download at a reduced price for existing Torque 3D owners, while those interested in purchasing the full version can visit the official web site to try out the free tutorial and demo.

maybe are FPS-based, that kind of thing. So rather than the social-type titles, you're talking about titles that have a high level of cinematic 3D content. That's really where we add the most value.

BS: I mentioned it because I know that Unreal and Unity are both pushing the browser side as well. Is that something you guys are looking to push further with the Vision engine?

DC: Yeah, I mean Vision has a web player available, which works in-browser, and it's an area we're looking at closely. I mean I think it's still to be proved out with a hit title that's really doing that in 3D, but we're certainly watching that area.

BS: You folks at Havok and Autodesk have been slowly absorbing every game tool, and now I saw that there will be some Autodesk integration into the

Vision engine. How much further do you think that sort of thing is going to go?

DC: Well, certainly on the Havok side it's really about rounding out the technology base, and I think at this stage we've got a nice suite of products that address both the engine side, and a lot of the key middleware components that anyone would need to build a game. We did announce a collaboration with Autodesk, where their Scaleform and Beast products are going to be available as part of Vision inside a bundle, and that just sort of further rounds out the offering.

I mean we're a company with the resources to look at acquisitions, but we're also a very discerning buyer. And we're really only interested in picking up companies where we know the team well, where we really feel they can become part of the Havok family and can contribute, and where it makes technical sense to do so. I mean we're not just trying to buy revenue for the sake of it.

BS: And have you found that you need to adjust your strategy at all coming to China, versus Western countries or even Japan?

DC: Yeah, I mean I think it's clearly a very different market, and I think we're still finding our feet to some extent. We've done business with a number of indigenous companies and we're finding our way there, but I think we're going to know more in a year than we do now; we're going to know a lot more in three years than we do then as well.

BS: Have you had any difficulty with the kind of governmental overhead that is involved with coming to China?

DC: Not yet, no. And you know it hasn't really been a barrier to date, but I think we're going to be going through a number of steps over the next while, including looking at actually putting on a presence on the ground here in Shanghai. So we need to see what the road ahead is like. ☺

MAKE MORE ENEMIES

Game Design at VFS lets you make more enemies, better levels, and tighter industry connections.

In one intense year, you design and develop great games, present them to industry pros, and do it all in Vancouver, Canada, a world hub of game development.

The LA Times named VFS a top school most favored by game industry recruiters.

VFS Find out more.
vfs.com/enemies

JOIN VFS AT

**GAME
DESIGN
EXPO**

JANUARY 21-22, 2012
VANCOUVER

gamedesignexpo.com

VFS student work by Aldo Martinez, Cazadilla



The best ideas evolve.

Great ideas don't just happen. They evolve. Your own development teams think and work fast.

Don't miss a breakthrough.
Version *everything* with Perforce.

Software and firmware. Digital assets and games. Websites and documents. More than 5,000 organizations and 350,000 users trust Perforce SCM to version work enterprise-wide.

Try it now. Download the free 2-user, non-expiring Perforce Server from perforce.com

Or request an evaluation license for any number of users.





PRACTICAL PTEX FOR GAMES

THE TROUBLE WITH TEXTURE UNWRAPS

IN 2008, WALT DISNEY ANIMATION STUDIOS (WDAS) REALIZED IT HAD a problem. Texture unwraps were serializing the production pipeline and preventing some production groups from using textures to represent their data efficiently. Compounding the effect, each new movie brought artist requests for higher-resolution textures and more texture layers. These demands provoked Brent Burley, a veteran engineer of WDAS, to propose a fundamentally different texturing scheme: Per-Face Texture Mapping, or Ptex.

Although novel and revolutionary, the soul of Ptex can be summed up with a simple sentence: Every face in a mesh should have its own entire UV space, and the orientation of that UV space should be determined implicitly from the ordering of its constituent vertices. With the imposition of a few modeling requirements, Ptex:

- **Eliminates explicit UV** parameterization of models
- **Eliminates texture** seams, even under high levels of magnification
- **Does more** with less—at SIGGRAPH '11, Brent Burley showed that WDAS has seen a roughly 15x increase in efficiency due to simultaneously increasing texture details while maintaining or reducing production costs.

As has become something of a norm, movie studios have again served as portents for the game industry. Texture unwraps have always been a headache in content production. And while there are tools to produce automated unwraps, they don't do the job nearly as well as a talented artist. What's more, the model unwrap causes subtle but noticeable serialization in content production.

Consider the specific example of a cutscene model. In order for a model to look the absolute best that it can for a cutscene, the scene must be rigorously storyboarded to determine which parts of the model will be visible. Those areas receive a disproportionate share of texture space compared to the areas that will not be featured. Changing this at a later date requires a re-unwrap and transferring of data between texture unwraps. This is a nontrivial operation for the art team, which is already the typical critical path. With Ptex, detail can be trivially added to any part of the model at any time, eliminating the serializing effect of UV unwraps

It is said that a picture is worth a thousand words, and so it is here.

Figure 1 shows a close-up of the tortoise mesh with implicit UV mappings overlaid adjacent to texture data.

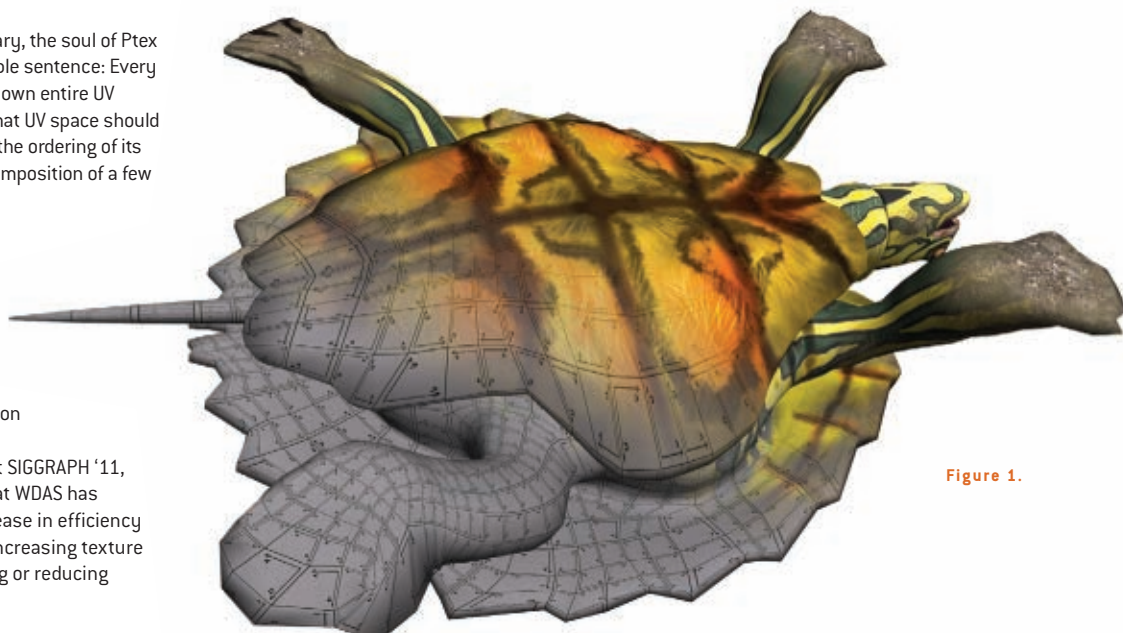


Figure 1.

PTEX REQUIREMENTS

Before discussing a real-time implementation of Ptex, it's useful to review its modeling requirements. The technique imposes a pair of restrictions on art assets that are not otherwise present, and reinforces the necessity of good modeling practice. First, Ptex requires that artists model with quads—not the ubiquitous triangles found in real-time rendering. Ptex has a workaround to deal with the occasional triangle, but it is inefficient, so their usage is heavily discouraged. In practice, it turns out that all cases where a triangle would be desirable can be replaced with 1–2 additional quads without adversely affecting model silhouettes. Moreover, the vertex portion of modern rendering hardware tends to be designed to not be a bottleneck, compared to the CPU and pixel portion of the pipeline—meaning that a little “waste” in vertices is reasonable if the result improves quality and overall performance, or reduces costs.

Beyond the model-with-quads requirement, Ptex additionally requires that textures applied to faces be power-of-two sized. Rectangular textures



are allowed, but each dimension must be a power of two. This requirement is necessary to ensure seam-free filtering across texture discontinuities, and is particularly important to Ptex where a texture discontinuity is guaranteed at the edge of every primitive in a mesh.

Finally, Ptex requires that artists follow long-established “best practices” for modeling. Ptex can only filter seamlessly against exactly one neighboring primitive for a particular edge. As a result, T junctions in a model would cause filtering artifacts, and must be avoided. A T junction results when one edge of a primitive has two or more neighboring primitives, where the other neighboring primitives are adjacent for only a portion of the shared edge.

THE PIECES FOR REAL-TIME PTEX

In August 2008, Microsoft released Direct3D 11 [D3D11]. D3D11 brings with it a lot of new functionality, but for real-time Ptex, the most important pieces of that functionality surround hardware tessellation. Hardware tessellation introduces three new pipe stages (two programmable and one fixed function), the ability to output primitives as quads, and new user-defined primitive types that can use anywhere from 1–32 indices per input primitive.

One other key piece of the real-time Ptex puzzle came earlier, with Direct3D 10: Texture2DArrays. Texture2DArrays are similar to volume textures, but with three important differences. First, Texture2DArrays only decrease in the U and V dimensions at smaller mipmap levels—the W dimension remains a fixed size. Secondly, Texture2DArrays do not allow filtering between array slices—all texels in a filtered sample will come from the same logical texture slice. Third, Texture2DArrays are indexed by an integer slice index—making indexing easier and less error prone than treating a volume texture as a Texture2DArray. The key takeaway here is that Texture2DArrays allow efficient access to texture resources that are similar in dimension, component layout, and type, but otherwise are completely unrelated. Combining these pieces yields a view of Ptex data sets that can be operated on efficiently by the GPU.

Presenting Ptex efficiently to the GPU means simultaneously optimizing for many things (in roughly decreasing order of importance):

- Batch counts
- Number of simultaneously bound textures
- Number of texture fetches
- Non-branchy code
- Keeping vertices thin and not making every vertex unique

“Batch, batch, batch!” has been the advice for the last eight years, and that advice hasn’t changed. D3D11 is much more efficient than Direct3D 9 on a per-batch basis, but keeping batch counts down is still critically important to the performance of modern games. This certainly eliminates the most naive implementation of Ptex, where we would simply bind a texture, draw a single quad, and then repeat for each quad in the base mesh.

D3D11 allows you to have 128 textures bound simultaneously. However, having that many textures bound simultaneously chokes the shader compiler. Moreover, this turns out to be completely unnecessary because of the advent of Texture2DArrays. In D3D11, the maximum array size of a Texture2DArray is 2,048 [as specified by the constant D3D11_REQ_TEXTURE2D_ARRAY_AXIS_DIMENSION]. Combined with the previous performance requirement, this means that a single Texture2DArray will allow rendering of up to 2,048 quads in a single batch.

In order to provide seam-free rendering across texture discontinuities, Ptex has to perform extra work at the edge of a primitive (and thus, at the edge of a texture page). Specifically, it needs to determine which primitive is the neighbor, determine the texture associated with that primitive, find the correct texels in that texture for this texture fetch, and filter them with the

textures from this texture page. In order to avoid seams entirely, it’s important that the limit of the filtering operation be identical when approaching from either side of the discontinuity. Fortunately, all of this is made straightforward as the necessary neighbor information is embedded in the Ptex file format. Unfortunately, a large filter kernel combined with low-resolution textures could effectively cause a single texture lookup for each sample in the filter kernel—with significant latency penalties between each lookup.

To avoid both of these issues (as well as avoiding needlessly branchy code), we can prepare a border region ahead of time that contains the neighbor texel data that might be needed for any texture fetches within a particular surface, including at the extremities. The size of this border is dependent on the maximum filter kernel size to be supported. For example, a bilinear footprint is only 2x2, and sampling exactly along an edge would require two texels of data from this texture and two texels of data from the adjacent neighbor across that edge. Anisotropic filter kernels similarly require a border size that is half their level of anisotropy. Using borders in this manner does increase the complexity of the real-time Ptex algorithm, but the benefits are tremendous.

The final piece of this efficiency puzzle is the data that lives in the vertices themselves. Cramming all of the necessary data for Ptex into vertices would destroy any vertex sharing presently enjoyed by the mesh. To avoid this, we can leverage arrays of structs in constant buffers, which can then be indexed by a new shader system value: SV_PrimitiveID.

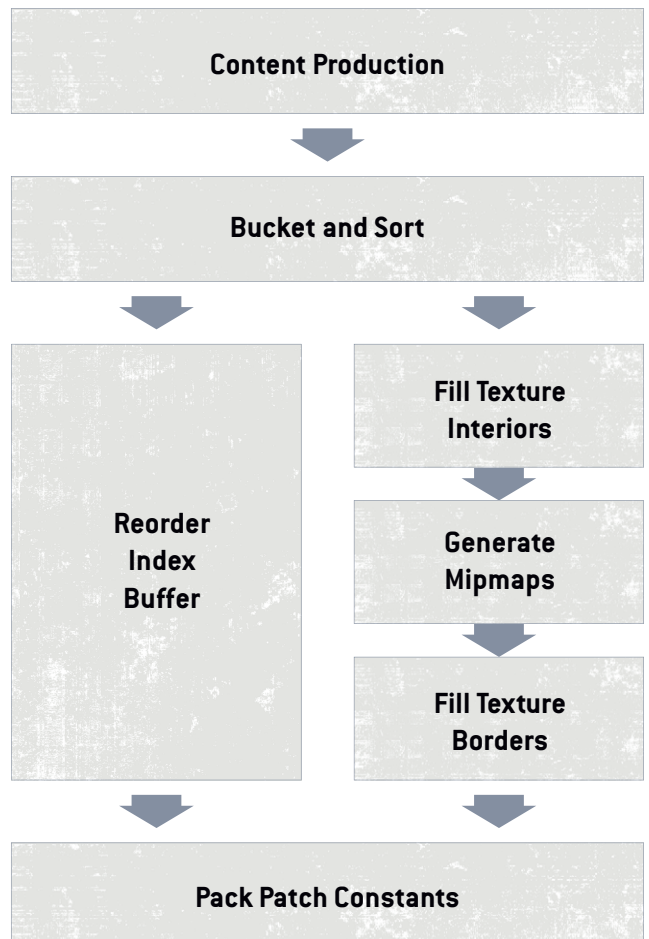


Figure 2.

SV_PrimitiveID is a counter the hardware provides which begins at 0 for each render batch and counts up to the number of primitives in that batch. This value is available for use in the Hull Shader, Domain Shader, and Geometry Shader. Combining the usage of SV_PrimitiveID with that of arrays of constants allows a new concept: patch constants. Patch constants are values that vary between different patches but are constant among the vertices that contribute to that patch.

PUTTING IT ALL TOGETHER

With all of these performance constraints in mind, we propose the following solution for real-time Ptex:

- **Source textures** are bucketed by aspect ratio and then sorted by decreasing maximum surface size. Each bucket will consist of at least one draw call (more if there are more than 2K surfaces with that aspect ratio). In order to reduce the number of buckets, we can group surfaces which have mirrored aspect ratios, that is to say that 2:1 and 1:2 surfaces can go into the same bucket. A simple flag can be specified to swap the UVs calculated by the domain shader.
- **For each aspect ratio** bucket, construct N Texture2DArray objects, where N is the maximum height of the mipmap pyramid for any one logical source texture within that bucket. The largest level of the mipmap pyramid for each texture is determined by its maximum resolution. However, the smallest mipmap level is no smaller than the smallest supported filter kernel size.
- **The dimensions** of each Texture2DArray within a bucket is $(2 * \text{BorderSize} + \text{width})$, $(2 * \text{BorderSize} + \text{height})$, $(\# \text{ of textures with a mipmap of this height, width for this bucket})$. In this case, $\text{BorderSize} = \frac{1}{2} * \text{largest supported filter kernel size}$.

This arrangement allows us to densely pack our Texture2DArray objects, avoiding any empty pages which would contribute to wasted memory.

A REAL-TIME PTEX PIPELINE

Transforming Ptex from its source pieces to a final model onscreen can be thought of as a logical pipeline, as seen in Figure 2.

Producing content for Ptex is relatively straightforward. Tools that already enjoy wide adoption in game development—including Maya and Mudbox—already support Ptex natively or via plug-ins.

Once a suitable model is acquired, the programmatic work begins. At export time, the mesh textures should be bucketed (based on aspect ratio) and sorted in decreasing order by texture size. If there are more than 2,048 elements in a single bucket, that bucket should be split into sub-buckets that are each no more than 2,048 elements long.

Based on the sorted texture order, a suitable index buffer is created. The simplest index buffer exactly matches the newly reordered texture ordering, although this order prohibits any vertex sharing. A better index ordering first matches the newly reordered texture buffers, and then sorts for sharing, while remembering the mapping between rendered face and texture buffer.

After (or in parallel with) index buffer reordering, texture interiors must be filled. This is mostly a simple copy process. Load the texture data from each face's texture and copy into its logical location in the Texture2DArrays. This is also the time that we note (and swap) the U and

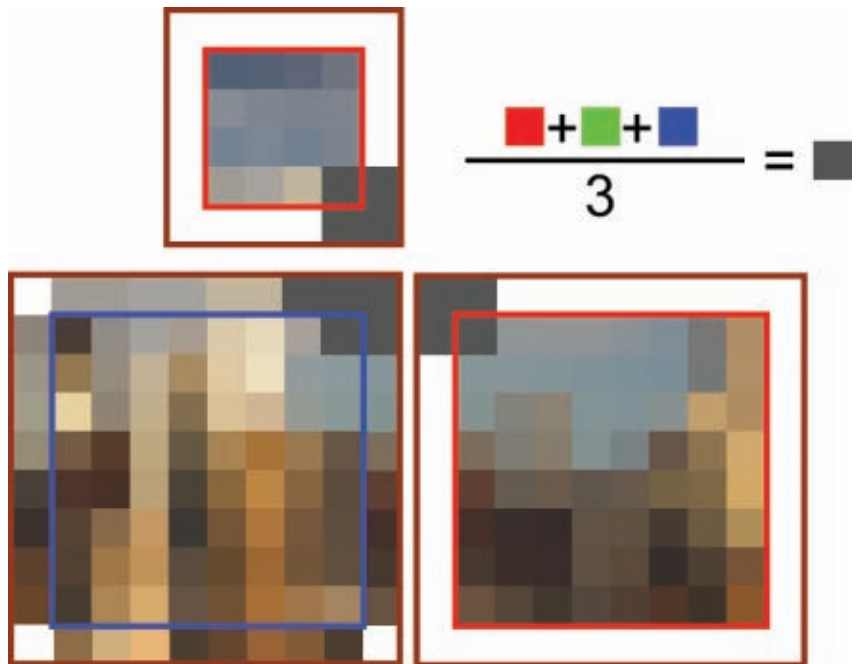


Figure 3.

V dimension, i.e. if we're preparing a 4:1 texture and the source texture is 1:4. Once this is done, we generate mipmaps. Every texture will wind up with a complete mipmap chain from its maximum surface size down to the smallest mipmap required. The smallest mipmap required will have dimensions no smaller than the smallest filter kernel required. Stopping the mipmap chain prematurely is necessary to avoid introducing holes where corners meet up, which we will discuss shortly. Note that during mipmap generation, it is important that only the interior of the texture be considered and written to—the border will be filled out entirely by a separate process, and should be ignored for now.

Once mipmap generation is complete, texture edges (excluding the corner region) must be filled. For each edge in each texture, find the neighboring texture with the common edge. Once located, the correct mipmap is selected from the neighbor. The correct mipmap is the one that has the closest number of pixels to our own along this edge, without going over. There are effectively three cases to worry about:

- **This edge has fewer pixels than the neighbor's edge**
- **This edge and the neighbor's edge have the same number of pixels**
- **This edge has more pixels than the neighbor's edge**

In the second case, data is simply copied across from that neighbor. In the third case, the data is copied across and then upscaled to match the resolution of the destination edge. The trickiest case to deal with is the first. In this situation, the neighbor has too much data for us to faithfully represent. The solution here is to ask for data from a smaller mipmap level that matches the required edge dimension. This step is repeated for each mipmap level, resulting in a full mipmap chain of textures with edge neighbor information fully filled out.

Note that edge-edge adjacency information is already stored in the Ptex file format, making this step relatively straightforward.

Next, border corners must be dealt with. Unlike edges, shared corner information is not stored in the Ptex file format. Therefore, we must first walk the mesh and note every shared corner. A shared corner is simply



LISTING 1

```
struct PatchInfo
{
    // The integer index of our texture
    // page, but written as a float.
    // This avoids a shading-time conversion
    // from integer (the type this really is)
    // to float (the type our texture lookup
    // requires).
    float fTextureIndex;

    // Whether the U and V value should be
    // flipped for this particular texture,
    // to allow 1:2 and 2:1 aspect ratio
    // surfaces to appear in the same bucket
    uint uSwapUVs;

    // The largest and smallest mipmap in
    // our mipmap chain, when used along the
    // interior
    // Could also store uBottomMipmapLevel
    // in a global constant--it is the same
    // for all surfaces
    uint uTopMipmapLevel,
        uBottomMipmapLevel;

    // The top mipmap level that can be used
    // when we approach each of the four
    // borders of our texture
    // Note that we do not need to store the
    // smallest because all textures will
    // stop at the same size.
    uint uU0TopMipmapLevel,
        uU1TopMipmapLevel,
        uV0TopMipmapLevel,
        uV1TopMipmapLevel;
};

cbuffer cbPatchInfo : register( b1 )
{
    PatchInfo gPatchInfo[MAXPRIMSPERDRAW];
};2
```

LISTING 2

```
float4 ptex(uint id, float2 uv)
{
    float texId =
        gPatchInfo[id].fTextureIndex;

    int topMipLevel;

    float fWMO, fWM1;
    _ptex_compute_mip_params(id, uv,
        topMipLevel, fWMO, fWM1);

    float4 f4SM0 = float4(0, 0, 0, 0),
        f4SM1 = float4(0, 0, 0, 0);

    if (fWMO != 0.0f) {
        float fMyMip = topMipLevel;
        float3 f3MyUVs = float3(
            _ptex_rebase_uv(uv, fMyMip),
            texId);

        f4SM0 = _ptex_sample(topMipLevel,
            f3MyUVs);
    }

    if (fWM1 != 0.0f) {
        float fMyMip = topMipLevel + 1;
        float3 f3MyUVs = float3(
            _ptex_rebase_uv(uv, fMyMip),
            texId);

        f4SM1 = _ptex_sample(fMyMip,
            f3MyUVs);
    }

    return fWMO * f4SM0 + fWM1 * f4SM1;
}
```

this data into an array in constant buffer memory and access the information via the system value semantic `SV_PrimitiveID`. Currently, we encode the data from [Listing 1](#).

ENOUGH PREPARATION, LET'S RENDER

Once we have the vertex, index, texture, and patch constant data prepared, we're ready to begin rendering. Splicing `Ptex` into existing Direct3D 11 shaders is a relatively straightforward affair. It works with all forms of quad-based tessellation, including displacement mapping, requiring only minimal patching to the hull and domain shaders (and their respective output structures), and slightly greater modification to the pixel shader (although most of this work can be hidden).

The hull shader constant function will be trivially modified adding only an input value (which will be provided by the hardware):

This assumes that your hull shader output structure is called "O," and you've added a field of type `uint` called `uPrimId`. Patch Constants will be referenced by the value `SV_PrimitiveID`, but the hull shader is the last shader stage that sees input primitives prior to expansion, where `SV_PrimitiveID` directly maps to input primitives. Therefore, we ask for that mapping here and pass it down as an interpolant for other shader stages to use.

the corner of a texture that shares an object space position with another corner (typically from another texture). Once these are determined, a dominant corner value is determined. This value can be computed through any number of means, the easiest of which is a simple average. Once computed, the dominant value is stamped back into each of its shared corners, in a (Maximum Supported Filter Kernel x Maximum Supported Filter Kernel). This can more easily be seen in [Figure 3](#). Note that this actually pollutes the interior texture in the corners! However, the result of this pollution is that all shared corners effectively "pin" to the same final value, ensuring a consistent output value regardless of which texture the corner is approached from.

After reordering the index buffer and filling in the border regions, we can prepare patch constant data. As previously mentioned, patch constant data is data that varies between patches—including it in our vertex declarations would destroy all vertex sharing. Instead, we encode

Modifying the Domain Shader is also straightforward. First, we simply convert the system value input `SV_DomainLocation` into texture coordinates for later use (either in the domain shader for displacement or in the pixel shader for normal texture lookups). Secondly, we need to swap the meaning of U and V if we're using a swapped texture. Finally, we need to pass the `uPrimId` field through to the downstream shaders. Assuming this domain shader function:

```
[domain("quad")]
DS_Out DS_PTex(HS_ConstOut cdata,
const OutputPatch<HS_CPOut, 4> I,
float2 f2DomLoc :SV_DomainLocation)
```

The modifications to the domain shader would be as below:

```
bool bSwapUVs =
gPatchInfo[cdata.uPrimId].uSwapUVs != 0;

0.f2TexCoord = bSwapUVs
? f2DomLoc.xy
: f2DomLoc.yx;

0.uPrimId = cdata.uPrimId;
```

Although not shown here, some modeling packages may require you to adjust the value of `SV_DomainLocation` before being used as a UV—surfaces from Mudbox 11, for example, are stored with an inverted U compared to the naive interpretation of the domain location. Additionally, the domain shader is responsible for UV swapping. Recall that we have stored mirrored aspect ratios (i.e. 1:8 and 8:1) in the same bucket. It is in the domain location that we swap the sense of what U and V mean, if necessary.

The most complex shader modifications lie in the Pixel Shader (although the complexity would be shared in the domain shader if displacement mapping were used). Fortunately, the complexity can be hidden in utility functions and not exposed to client code. First, the correct texture must be accessed. This is a simple matter of using the patch info constant data and the `uPrimId` value to get at our patch constants. Second, the mipmap parameters—including which mipmap to sample and the strength of each sample level—are computed. Third, the appropriate mipmap levels are sampled, and finally the resulting samples must be multiply-added together to return the final sample value. This functionality is placed in a function called “ptex,” as shown in Listing 2.

The function “ptex” replaces top-level sampling functions, such as `Texture2D::Sample`. Functions within the `ptex` function that begin with an underscore are private (by convention) and are not particularly useful to other code.

THE GUTS OF A PTEX TEXTURE LOOKUP

Before any other work can begin, `_ptex_compute_mip_params` is called to determine which mipmap levels need to be sampled and what the weight of the samples from each of those mipmaps needs to be. Listing 3 shows the implementation of this function, along with its helper functions.

I've omitted the body for `calc_lod_noclamp`. This function contains a mapping of `minLOD` to literal texture element entries by simply doing entries of the form “case N: `gTexColor[N]E`” for values of N in `[0, MAX_MIP_LEVELS]`. The same technique shows up again in `_ptex_get_dims` and `_ptex_sample`. This technique is necessary due to the fact that HLSL currently requires that arrays of textures must be accessed via literals. Therefore we use a switch statement to map from variable to a literal. The low-level compiler can typically do the correct behavior and fold the instruction back to a simple indirect texture lookup.

Beyond that, the functionality here is relatively simple and matches

the functionality of the fixed-function texture unit, computing an LOD, then clamping to available mipmap levels and finally returning the appropriate weighting values for a texture lookup.

The next function called by our “ptex” implementation is `_ptex_rebase_uv`s. Because our mipmaps include borders that are fixed in their number of texels (independent of which mipmap level we are on), we must adjust the UVs to stay in bounds for all texture fetches. We cannot do this until we know which mipmap level we are sampling from. The body of `_ptex_rebase_uv`s and its helper function are shown in Listing 4.

Again, I've omitted the body for `_ptex_get_dims`. This function behaves exactly the same as `_ptex_calc_lod_noclamp`, but calls a different function (in this case, `GetDimensions`). Once the dimensions are known, the texture coordinates are scaled and biased to ensure that the filter kernel touches (but never leaves) the border region.

The final piece of our “ptex” function is actually sampling using the computed mipmap levels and rebased UV values. This occurs in the `_ptex_sample` function. Unfortunately, this function is also omitted due to the ugly “case N: return `gtxColor[N]E`” switch statements—which are too wide for print. However, an example body for `_ptex_sample` is available at <http://gdmag.com/resources>. In this function, each case converts the variable `mip` to a literal value and calls the built-in texture function “`SampleLevel`.” `SampleLevel` takes a sampler, `float3`, and the level (0) as arguments. Using `SampleLevel` avoids a compiler warning regarding performance.

```
float4 _ptex_sample(int mip, float3 uvindex)
```

LISTING 3

```
float _ptex_calc_lod_noclamp(int minLOD,
float2 uv);

float _ptex_calc_lod(int id, float2 uv)
{
int minLOD = gPatchInfo[id].uTopMipLevel;
return _ptex_calc_lod_noclamp(minLOD, uv);
}

void _ptex_compute_mip_params(int id,
float2 uv,
out int outMipLevel0,
out float outWeightM0,
out float outWeightM1)
{
float fLod = _ptex_calc_lod(id, uv);
fLod = max(0, fLod);
outMipLevel0 = min(
gPatchInfo[id].uTopMipLevel + fLod,
MAX_MIP_LEVELS);

outWeightM1 =
(outMipLevel0 == MAX_MIP_LEVELS)
? 0
: frac(fLod);

outWeightM0 = 1 - outWeightM1;
}
```



LISTING 4

```
float2 _ptex_get_dims(int mip);

float2 _ptex_rebase_uv(float2 uv,
    int selectedMipLevel)
{
    float2 texDims =
        _ptex_get_dims(selectedMipLevel);

    float BorderSize = g_f4BorderSize.x;
    float BorderSizeTimes2 = g_f4BorderSize.y;

    float2 Bias = float2(
        BorderSize / texDimensions.x,
        BorderSize / texDimensions.y
    );

    float2 Scale = float2(
        (texDimensions.x - BorderSizeTimes2) /
        texDimensions.x,
        (texDimensions.y - BorderSizeTimes2) /
        texDimensions.y);

    return uv * Scale + Bias;
}
```

ADDITIONAL REFINEMENTS

The example code listed above omits one important detail: matching mipmaps exactly along primitive edges. Failure to match mipmaps exactly could result in a texture seam along those edges. For diffuse or normal maps, this would merely result in a color fault. However, in a displacement map this could result in a hole. Fortunately, fixing this is a straightforward modification. During mipmap parameter calculation, the fully-functional code would:

1. Compute the radius of the filter kernel in UV space for this mipmap level
2. Determine if the raw UVs +/- the filter kernel radius overlaps an edge [that is goes <0 or >1]

GPU	FILTER	TIME IN MS		OVERHEAD
		No Ptex	Ptex	
GTX 460	Trilinear	0.834	0.834	~0%
	8x Aniso	0.842	0.882	4.5%
GTX 480	Trilinear	0.825	0.824	~0%
	8x Aniso	0.835	0.835	~0%
AMD 6970	Trilinear	0.727	1.567	54%
	8x Aniso	0.727	1,481	51%

TABLE 1: LOWER IS BETTER.

3. Determine whether the current mipmap level is lower [larger] than the neighbor across that edge, and, if so, clamp to that mipmap level instead. Further, the weights are adjusted such that M0 would receive a weight of 1 and M1 would receive a weight of 0
4. Alternatively, determine if the neighbor across that edge has a higher maximum mipmap level than us, and if we are accessing our highest mipmap level. If so, we again adjust our weights such that M0 receives a weight of 1 and M1 receives a weight of 0

PERFORMANCE

Testing was performed on a Nehalem I7 at 2.94 GHz with 3 GB of RAM. The turtle mesh from the article's introduction was rendered at 1920x1080 with 4xAA using the latest public drivers available from each vendor. The turtle mesh contains 4,160 quads and uses 170M of texture data.

As you can see, even with high levels of anisotropic filtering, performance is quite reasonable and overhead is generally low. It is reasonable to assume that the high overhead experienced on the AMD hardware tested could be fixed in a future driver release—although the performance is still quite reasonable for character rendering.

OTHER CONCERNS

A common concern regarding Ptex's usage in games is how to deal with changing models during production. WDAS suggests using a combination of techniques to address this.

To begin, perform a comparison of mesh primitives between the old and new mesh. The primitives that are unchanged between versions of the mesh (within some threshold) have their texture data simply copied from the old surface to the new surface. For primitives that have been adjusted, use a point-cloud transfer process to effectively scatter the texture data from the old mesh and then reproject it onto the new mesh.

Another concern is texture waste. Ptex texture waste is heavily determined by two factors. First, the texture border size. Larger filter kernels require larger border regions, and those border regions are simply a duplication of the texture data (ie, waste). Second, there's the number of faces of each texture size. Smaller textures spend a larger proportion of their memory on border regions, leading to more waste. For a concrete example, a mesh with 2,048 quad faces made entirely of 32x32 textures (4 channel, uncompressed) that supported only trilinear filtering would use approximately 9 megs of memory, compared to 8 megs of the same texel density in traditional texturing. This represents a waste factor of 1.13. However, the same mesh supporting 16x anisotropic filtering would require 18 megs of memory, for a waste factor of 2.25.

One final concern regarding Ptex is backward compatibility: will Ptex run on Direct3D 9 class hardware? Although more research is needed to answer authoritatively, the short answer is no. The functionality can be emulated, but the performance would most likely be unacceptable. 🚫

JOHN McDONALD is a DevTech engineer with NVIDIA. He spends most of his days working with developers on performance while also devoting time to researching future technologies that will decrease game development costs without sacrificing quality. When not working, John spends time with his wife and pets, or out hiking and rock climbing. John can be reached via email at jmcdonald@nvidia.com. Special thanks to Johnny Costello for the turtle model featured here.

references

More information on Ptex, including a specification of the Ptex file format and the official open-source library, can be found at <http://ptex.us/>. For information specific to real-time Ptex, including slides and demos, check <http://developer.nvidia.com/>, or contact John McDonald directly.



THE 14TH ANNUAL
INDEPENDENT GAMES FESTIVAL
AWARDS

WEDNESDAY, MARCH 7, 2012 | 6:30-8:30PM
MOSCONE CONVENTION CENTER | SAN FRANCISCO



IGF AWARD CATEGORIES

IGF MAIN COMPETITION

- » Seamus McNally Grand Prize
- » Excellence In Design
- » Excellence In Visual Art
- » Excellence In Audio
- » Technical Excellence
- » Audience Award
- » Best Mobile Game

IGF STUDENT SHOWCASE

- » Student Showcase Finalist
- » Best Student Game

NUOVO AWARD

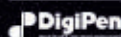


CELEBRATING OVER 800 INNOVATIVE GAMES ACROSS THIS YEAR'S
MAIN, STUDENT, AND NUOVO AWARD COMPETITIONS

VIEW THIS YEAR'S SUBMISSIONS AT WWW.IGF.COM

PLAY THE FINALISTS AT THE IGF PAVILION ON THE GDC 2012 EXPO FLOOR, MARCH 7-9, 2012

Platinum Sponsor





NEVER ENOUGH

AN INTERVIEW WITH DeNA'S KENJI KOBAYASHI

Kenji Kobayashi is director of DeNA, which runs the Mobage social mobile platform in Japan, and now the West. DeNA is one of the biggest social game companies in the world, and makes big money, boasting \$12 average revenue per user. We spoke briefly with Kobayashi at GDC China about how that huge revenue does or does not translate to employee compensation.

In a companion interview completed earlier in the year, Christian Nutt talks further with Kobayashi about the “pay for play” element of some social games, and how and why that works for DeNA.

Brandon Sheffield: *In your GDC China talk, you mentioned the importance of showing heavy non-paying users that it's easier to play the game if you buy items. Some users have a negative reaction to that "pay-for-exp" sort of thing. How do you view that?*

Kenji Kobayashi: I think the aim here is for the user to directly feel the effects of what he's paying for. If the user doesn't understand what this payment does for him, or what it lets him accomplish in the game, then he's not going to go through with it. So it's important to have the user think about the possibilities that become available. Users aren't going to immediately place money on the game the moment they start playing, so one thing you have to do is introduce something cheap at the start which will demonstrably make the gameplay experience easier. That way, even people who've always been dead-set against spending money might think “Well, this is at a big discount; maybe I'll give it a try.” Then, if they feel afterwards that making that purchase has made

things demonstrably easier for them, they'll be more open to that in the future.

BS: *But Western gamers don't like systems where you pay money in order to become stronger, or at least they say they don't. Is Japan and China different in that respect?*

KK: Well, in the case of Japan, it can make things easier, but I don't think it's that sort of thing so much as that it's important to make people feel like they got something real and good out of it. That's the case with FARMVILLE—it's important that you gain something palpable from it, and that palpable thing is something that's advantageous to you. In that respect, I think it's the same as how U.S.-made games work.

BS: *Perhaps, but Western games try to hide it to some degree, so it's not obvious that you're paying to win or get stronger.*

KK: That may certainly be the case, but I think the sort of people who are actively trying to avoid the payment system no matter what are not the sort of people who'd be interested in playing these sorts of games in the first place. The

audience is more people who are there in order to play games that are still playable for free—there's nothing inherently negative about not going into the payment system, in other words. The idea is to get people like that to maybe try out payments and then think "That made things really convenient." That opens up the possibility that they'll be heavier for-pay users in the future. Not everyone's going to be there, but I think this approach has the effect of getting more people there than usual.

BS: DeNA's social games are making a lot of money these days. How much does it cost to run the business versus what you're taking in?

KK: Well, with most social games we develop, there are two programmers, three engineers, and one art designer. That, and one database engineer. So about seven people overall, and it takes maybe four or five months to develop a game. These products have the potential to gross at least \$1 million a year, and the revenue can continue for upward of five years. So we can get at least \$5 million out of a game developed by seven people.

BS: Do those seven people share in the profit? What do you do with the rest of that money?

KK: Well, there is profit-sharing model with the third parties we work with, but as far as in-house staff goes, that is more regular compensation. The revenue of the title is related to the compensation of each employee, but not directly.

BS: Does that mean you guys are sitting on a huge mountain of cash? What will you do with it?

KK: I don't know if I'd call it a huge mountain, especially compared to our competitors like Zynga or EA. Those are huge companies with much more equity power, and that's what we're competing with, so we need a lot of money as well to keep up. So I think our cash position is not enough, if anything.

Christian Nutt: How important are analytics and data?

KK: It plays the most important role in our competitive abilities. Social

games are inherently doing pretty simple things gameplay-wise. You don't see the intense, intricate 3D graphics of CALL OF DUTY or something similar. It's really simple, and in some respects it's more like performing a task.

CN: So what part of that is fun for the gamer? That lies in the very delicate play balance that the games maintain.

KK: One example that often gets brought up is DRAGON QUEST, a game that's intensely popular in Japan even though it hasn't duplicated that in the U.S. It's a game that everybody knows, and the play balance there is really great. If you changed even a single parameter, though, it'd turn into just this terrible thing.

DRAGON QUEST is an RPG, and you run into enemies as you run around the world. If you changed the encounter rate... On the average, you run into an enemy once every seven or eight steps. Try to imagine it if it were once every three steps. It wouldn't be any good as a game—you couldn't finish it, the tempo would be off, it'd take too much time. The whole balance would collapse, over a single parameter.

Social games share a lot of aspects with that; the balance of the in-game parameters dictates whether it's a comfortable game, or an exciting game, or how fun fighting the enemy is. If you mess this up, it becomes a very uncomfortable game.

CN: Do you try to erase the user's stress with design?

KK: It's not that we erase it; we control and release it. When you think about what games are at the core, they are about delivering stress to the user. It wouldn't be fun if it's something that anyone could finish—you put up obstacles for the user, and they feel a sense of achievement when they overcome them, which is fun.

That applies to any games—the ability to do things that you couldn't before, or that other people can't. That feeling is very important, but, for example, if you have something like this [Kobayashi begins drawing]—if the capabilities of the user are here, and if the hurdle is here [not aligned]—then you can't clear it. Get a bit stronger, though, and you can overcome it and get

a sense of achievement. If it's like this [too far apart] though, then you would just give up.

If you were level 99 in DRAGON QUEST and killing (low-level) slimes, that isn't interesting. The user is always changing, and this applies to the balance of social games. You have to retain a balance such that it's fun to overcome the hurdles that come your way.

That, and lots of users are playing at the same time. In KAITO ROYALE [DeNA's MAFIA WARS-style game] you can have users just starting alongside people at level 2,000. It's extremely difficult to provide a play balance that can satisfy both extremes.

If you have this user and that user, then the solution seems to be to provide a difference balance; that seems to be the best idea. If you do that, though, at what time do you switch out? You could change the difficulty with each level or each stage, but where do you make the final decision? So the important thing that needs to be spread across the entire game is that users are enjoying what they're doing.

CN: In free-to-play games, there are a lot of different philosophies about whether items should give direct advantages or indirect advantages, from developers in the West.

KK: We're certainly different from Zynga in that respect. Both KAITO and NINJA ROYALE are games where you're gathering treasure, but if users could get this treasure via paid items, then the game balance would fall apart. There wouldn't be any meaning to it; you'd be ruining the core of the game.

What we sell instead is the opportunity to make treasure gathering easier. For example, have you played mahjong? This may work with poker, too. If you're one card away from a royal flush, you only get one chance to draw that card. Well, what if you paid 100 yen [\$1.31] and get three chances instead? [laughs] Which do you think is more interesting to the user: very difficult odds, or a chance at easier odds?

CN: The chance is what makes it interesting.

KK: It is. That in itself is interesting. Paying out has to be interesting in itself, I think. We're selling chances,

and it results in some interesting reactions from the users. For example, if you go out to buy toilet paper, there's nothing really fun about that. You pay money, and you get toilet paper. There's no volatility, so there's no feeling of excitement.

But let's say, though this isn't quite volatility, when you go to the Apple Store and buy a MacBook, that's more exciting. Giving users the opportunity to spend money and get something really exciting as a result is really neat, I think. The act of spending money, in itself, becomes appealing and fun. Like "Okay, here goes!" So it's a similar sentiment when you're near that royal flush; you're excited. That's the important thing.

CN: Some of these examples are a bit close to gambling, though.

KK: Well, I don't think gambling is a bad thing, but the difference from gambling is that with gambling, you potentially have money coming back at you. Pachinko wouldn't be such a huge thing if money weren't coming back to players.

CN: It's not a bad thing, but don't you think it's a little dangerous?

KK: The thing that I worry about the most is whether the user is getting excitement or entertainment that's commensurate with the money he's using. For example, having three chances at a royal flush for 100 yen; that's not a terrible way of balancing it. You approach it from the same philosophy as if you're buying a can of juice, except it has the potential to bring even more excitement.

If you tried to find a way to get a similar feeling of excitement for 100 yen with some other form of entertainment, that's going to be tough. Social games offer just a certain amount of stress and release to users.

Beating the final boss of FINAL FANTASY XIII would certainly produce a lot more excitement, but you have 60 hours of gameplay to get through to that, which also contributes to that big release. I think that users were looking for a lighter, more accessible form of entertainment, and providing that sort of entertainment, or sense of achievement, for the price of a can of juice is something I think is very neat. ☺



HOPE AND CHANGE

COPYRIGHT 101

ROLLING INTO A NEW ELECTION YEAR, YOU'LL probably see many variations on the famous "Hope" poster from the last presidential race. Artist Shepherd Fairey (of "Obey" fame) distributed more than half a million of the red, beige, and blue posters during the 2008 campaign. Political junkies and art critics alike regarded "Hope" as an instant icon, one of the most successful political images of modern times. It's also a cautionary tale for working artists in the age of Google.

/// The "Hope" image began life as a news photo from AP News Service photographer Mannie Garcia. Fairey found that image, like so many of us do, on Google Image Search. He then went to work, like so many of us do, without worrying too much about where the picture came from or who owned it. He had a point to make, and that was what interested him. The AP, however, was very interested in who owned the image. Unsurprisingly, the dispute ended up in court, and while the settlement last year was private, you can be sure that the AP is going to get a few nickels for every "Hope"-themed mouse pad sold this time around.

The Hope case is a perfect example of copyright law controversy, embodying the tension between the two sides of the law. The constitution explicitly endorses "securing for limited times to authors and inventors the exclusive right to their respective writings and discoveries." U.S. law has always tried to ensure that creators can control and profit from their work. At the same time, the law also tries to safeguard the right of the public to discuss and debate copyrighted works. In the digital age, the tension between these two goals has erupted into open war. On the one hand you have people, like the AP, who want to protect and profit from the valuable properties they've created. On the other hand you have the postmodern digital artists who want to make art by sampling, remixing, and mashing up (and, of course, the teenagers who want free music and movies).

For us, this copyright schizophrenia is internalized. Few game artists like the idea of pirates playing our games without paying. On the other hand, most of us also have grabbed useful textures or a striking image off the web without stopping to think of the legalities. Modern artists are trapped in the middle: Shepherd Fairey, whose whole career has been based on appropriating pop culture images, has himself sued others for re-using his own work. He denounced the creator of an anti-SARS poster featuring Andre the Giant wearing a respirator mask as a "parasite." He, in turn, has been denounced as a plagiarist. We've all got a bit of "do as I say, not as I do" when it comes to copyright.

OBHEY GIANT

» No matter what you think of the politics, paying attention to copyright issues is necessary for professional self-defense. You really, really don't want to be the artist who cost the company a few hundred grand because you guessed wrong about how to use something you downloaded.

Ask a lawyer for a summation of how copyright impacts you as an artist and they'll say something like this: If you didn't make it yourself, you must contact the person or company that did, and get their permission to use it. Don't think you need to see the copyright symbol, a legal warning, or a watermark; don't figure that it's OK because it's on Usenet; don't go hunting for images from some other country. Just assume that everything you see, every photo on the internet, every image in a printed book, every scrap of audio or video you ever come across—belongs to somebody who's going to sue you if you don't get their written permission to use it. End of story.

Pretty depressing, huh? Kind of kills the thrill of Flickr. But that's what the lawyers will tell you. It's the safe option.

The law is supposed to protect "transformative" uses of other people's work—Marcel Duchamp, in drawing a mustache on the Mona Lisa, created something new enough to deserve its own protection. However the boundary between "transforming" and "stealing" is infuriatingly vague. You can get into trouble for manually reproducing a photo, like Shep Fairey. Or, like jet-setting artist Jeff Koons, you could be found liable for hiring somebody to turn a kitschy post-card into even kitschier sculpture (so, it should be pointed out, making a 3D model from scratch is still "copying" if you work from reference). Snippets of borrowed audio as short as two seconds have been found to be copyright violations. It's really, really hard to know what's acceptable. Not even the lawyers can predict how the courts will rule on a given case. That's why they tell you to license everything—it's safer. Alas, it can also be very expensive.

Nowadays, with more photos on Flickr than there are people on the planet, you might contemplate gambling that you won't get caught.

If you do, though, it's a mug's game—the odds get worse for you the more successful you are. If your iOS game sells 10,000 copies, perhaps your use of somebody else's photo for a background won't be noticed. If you're on a big console game with TV ads and an endcap in every Best Buy, you'll find out that people can have remarkably sharp eyes and long memories—and, unfortunately, lawyers who work for a cut of the settlement.

FAIR USE

» If you didn't get a license (translation: pay money) for something you've found on the net, you can only use it within the confines of the legal doctrine known as fair use.

The idea behind fair use is pretty simple—while an author or a creator has the right to profit from their work, they are not supposed to use their copyright to prevent other people from discussing it (or, for that matter, making fun of it.) In copyright law, fair use is the name for the set of guidelines which distinguish between allowable discussion or parodies and illegitimate copying.

Fair use is a great idea. If not for fair use, we'd have no way to publicly discuss copyrighted work without the permission of the copyright holders. Like most great ideas, it's also an ideological battlefield. For anti-copyright activists, fair use is the living heart of the law—when the rights of readers, viewers, and artists who want to critically re-interpret culture conflict with the rights of the original creators, they would side with the consumers over the producers. Conversely, companies and creators with a strong economic interest in retaining control of what they make want fair use to be minimized and the rights of content owners to be protected first.

Fittingly, for such a controversial topic, the law of fair use is complex and sometimes contradictory. You've probably noticed the form that accompanies images on Wikipedia, spelling out why the image is allowed under fair use—all those bullet points are a good indication of how elaborate the rules governing fair use can be.

One thing's for sure: Fair use is not a magic amulet of protection. For starters, fair use is a defense against claims of copyright infringement.



PHOTO: ISTOCK.COM

This means that you may not get to explain why, say, copying a photo from a news web site to put on a poster in your game is a clever bit of commentary that's allowable through fair use, until you are already in court. At that point you've already blown a lot of time and money on lawyers, depositions, and explaining to your boss why you didn't just make your own damn pictures.

If, by some mischance, you do go to court with a fair use defense, the court will use several different measurements to see if your work falls under the protection of fair use. There are four classic considerations in any fair use case: purpose, amount of borrowing, the source, and money.

PURPOSE

» Fair use is designed to protect public debate, education, and academic research. This means academics, news organizations, teachers, and critics will have an easier time with fair use defenses. For-profit enterprises—including, perhaps, your company—typically have a tough time proving fair use. The rules that govern what a game review web site can do are going to be looser than those for a game; the rules for a student game will be less onerous than those for a big commercial title.

AMOUNT OF BORROWING

» Fair use generally protects citations and short excerpts, rather than wholesale copying. A reviewer can quote a paragraph out of a novel to make a point about a book, but copying entire chapters is not allowed. A photograph can be

hung on the wall in a movie set, but the film can't present it in a detailed closeup. Small borrowings are generally more likely to pass the fair use test than big ones, but there's no rule of thumb to tell you how much is too much.

Likewise, you can't count on image-manipulation tricks to turn a copied image into something unique. One of the earliest digital copyright fights (FDG vs Newsday, back in 1993) established that tricks like cutting a source picture apart and flipping the images still counted as copying, even though the remixed image could never be mistaken for the original.

THE SOURCE

» Some sources are better protected than others. Purely factual data—for example, the contents of a telephone book or a bus schedule—receives almost no protection at all. Published works are more protected than unpublished ones, for obvious practical reasons. A photo that's been published in a newspaper or magazine will be better protected against copying than, say, fan art that was anonymously posted online.

A few sources, such as federal government publications, are actually exempt from copyright protection and can be freely used. NASA and the military, in particular, are goldmines of copyright-free material. Be careful, though. The government often licenses images from commercial sources, and those don't lose their copyright status just by appearing on a .mil or .gov web site. As always, do your homework before using an image.

Interestingly, it's not possible to renounce your copyright under current law. You can give your work away, but you always have the right to change your mind. However, many copyleft fans will let you use their work for free with few or no restrictions. The Creative Commons project (www.creativecommons.org) and Flickr's Creative Commons License Search are easy sources for open source images. Of course, you should double check the specific permissions included with any images you download (some, for example, can't be used in for-profit products, while others require a credit for the author).

MONEY

» The last fair use test concerns the effect of the copying on the marketability of the original work. If the borrowing causes the original copyright owner to lose sales, whether actual or merely potential, it's very difficult to establish fair use. Because copyright is ultimately about providing an economic incentive to authors and artists, the marketability test is given a lot of weight in court cases.

Direct competition, of the street-corner Rolex

variety, is obviously illegal. "Competition" here is usually defined very broadly. You can't, for example, claim that re-using a character in a different genre or new platform means you're not competing with the original. If you put an obscure N64 game character into your PS3 title, you may not touch the sales of the original, but you've still diminished the original authors' chances to create a new PS3 product, so you're still not protected by fair use.

Ironically, the one form of economic impact that copyright can't protect against is the most direct one: reviews. Even though a review can impact the sales of a game—to take an obvious example—developers can't refuse reviewers permission to use screenshots or quotes in a negative review. Courts have traditionally held that the negative impact here comes from the new, unique review content and not from the borrowings.

OF CATS AND COPIES

» If there's a one-word summary for the state of copyright law, it's "messy." Getting involved with the legal system, even if the worst thing that happens is an exchange of nasty letters, is expensive and time consuming. Be cautious when you re-use anything from the web, because a mistake can be very costly. Unless your bosses are ardent copyleft activists, they probably won't thank you for the opportunity to meet the complex nuances of the fair use doctrine in practice.

Unless you're actually trying to build a game on a critical appraisal or parodic critique of somebody else's work, fair use is probably not of much use to you. If you are on a project that really wants to spotlight the appropriation of published material, you should probably talk to a lawyer, and early. (Indie teams and passion projects can and should talk to an organization like Volunteer Lawyers for the Arts, at www.VLANY.org.)

For most of us, though, the real lesson is to think before we download. As working artists, we're uniquely positioned to see both sides of the copyright conflict. We want to make money on our work, but we'd like to use all the wonderful resources of the web to make our own lives easier. Until the law catches up with the digital realities of our jobs, we have to use our heads to stay out of trouble. At least making it hard to copy textures will keep our Photoshop skills in demand for a while! 🎮

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 the technical art director at Seattle's *Undead Labs*.



THE 12TH 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.

AWARD FINALISTS WILL BE ANNOUNCED IN JANUARY.
Stay updated at: www.gamechoiceawards.com.

AWARDS ARE PRESENTED IN THE FOLLOWING CATEGORIES:

2011 AWARD CATEGORIES

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

SPECIAL AWARDS CATEGORIES

- Lifetime Achievement
- Pioneer
- Ambassador

PRESENTED BY

GameDevelopers
Conference

IN ASSOCIATION WITH





WHAT'S NEXT FOR THE NEXT NEXTGEN?

A SURVEY OF AUDIO'S NEW FRONTIERS

AS BOTH A CREATIVE AND TECHNICAL DISCIPLINE, INTERACTIVE AUDIO HAS made tremendous strides over the last console cycle. When I took over writing this column six years ago, most companies were still laboring to build proprietary audio engines. Handheld games were primarily in the control of first-party console manufacturers. Rich, interactive music was a cutting-edge feature rarely found in-game. iTunes' video game marketplace, Kongregate.com, and Zynga didn't exist yet.

Six years on, smartphones now dominate the handheld market, and games have a thriving independent developer community again. Proprietary audio engines have lost ground to Wwise and FMOD. Interactive music is now the rule rather than the exception, and the internet is now the frontier of game platform innovation.

As we gear up for the next generation of consoles and the next burst of technical and creative advancement, it's worth looking at what the next areas of focus should be for game audio.

FIX IT IN THE MIX

» At the start of the next generation, sound designers are finally in a place where the battle in AAA development for detail and variation is a waning concern. Complex instance culling and stream management systems are already a must, and will only continue to become more of a fundamental need. Sound designers can expect the usual fidelity increases to both asset bit and sample rates and, subsequently, the usual increased impact on both memory and disc footprints.

But that's the small stuff. The next generation of game sound is going to be about maturity. We've shown how big the worlds we can create can be. Now we need to show how well we can get them to sound.

As such, the new frontier of sound design is mixing and mature implementation. The days are gone where it's acceptable to simply have static master levels for sound and music that are occasionally ducked by voice. Nuanced mixes and intelligent systemic mixing systems are the next big focus. As we gain the ability to add more real-time convolution reverbs and more detailed surround ambiances, we're going to need the ability to deftly sculpt frequency space and create situational mixes that change depending on player feedback and myriad shifting game states.

CLOUD CONTROL

» Game music only continues to get more sophisticated and more complex, both compositionally and technically. Wwise and FMOD's considerable acceptance across the industry has given game composers an advantage that was sorely missing for years: standardized tools. When the gig can be about composing interactive music—as opposed to building



technology that facilitates composing interactive music—composers as a group can begin to focus on innovation rather than reinvention.

The ubiquitous music loop was once the undisputed king of game scoring. King Loop, however, has become tiresome, and audio teams across the industry are now working with game music systems that focus on more variety. Interactive music scores are starting to be more about stitching non-looping material together, rather than wall-to-wall loops of repetitive music. This increase in variety is bringing with it an increase in the amount of music needed to cover a game.

By the end of the next generation of games, conversations regarding “disc footprint” will be a thing of the past. Cloud storage and cloud streaming of content will not only be the norm for digital distribution of game software, but also for game content. In-game radio stations, streamed level music for Facebook games, and even faction-specific multiplayer music no longer needs to live on a physical disc or even on the end user's machine. If game developers and publishers don't offer the technology themselves, expect to see a rise in streaming services akin to YouTube

and SoundCloud that can be used to propagate in-game musical content.

VOX POPULUS

» In-game text is an endangered species. We've reached a point as an industry where even a gigantic MMO like STAR WARS: THE OLD REPUBLIC is fully voiced. As such, expectations from players are shifting. Increases in storage space mean that everything that can be voiced should be voiced. Additionally, players are coming to expect a wider range of localized languages within games. Including Chinese subtitles simply won't be enough anymore.

As graphics and animation technology improve, game voice is going to be massively impacted by the further proliferation of facial motion capture. Our industry was once closely related to the animation world, in which VO actors ruled the day. We're now shifting into a camera- and physical performance-oriented field. More and more, game industry casting directors are looking for their actors to have previous motion capture experience as well as a physical likeness that can be used in-game. The traditional role of voice over-only talent isn't going to go away. However, this influx of new film and TV-vetted actors, mixed with an increasing importance placed on cast records and actor involvement in script and character development, is already bringing a new level of nuanced drama to our games.

After six years of writing for “Aural Fixation,” this column will be my last. Thanks to everyone who's taken the time to read and comment over the years. See you all in the trenches. 🙌

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.



WRITE FOR THE PLAYER

A SIMPLE TECHNIQUE FOR FIRST-PERSON NARRATIVE DESIGN

ONE OF THE MORE DIFFICULT ITEMS ON MY LENGTHY TO-DO LIST IS TO explain to first-person narrative designers why our players are not enthralled by our stories. Why, in fact, our stories don't reach players at all, and, more importantly, what we are going to do to fix that. This is especially difficult since, more often than not, narrative designers are certain of the correctness of their current path.

Over the years, I've developed a philosophy of first-person narrative design. That philosophy (and the simple technique and practice it inspires) is the subject of this article.

THE WHOLE ARTICLE IN ONE PARAGRAPH

» I assert that first-person interactive storytelling has a fundamentally different constraint on the hero/avatar than other types of entertainment. Specifically, players project themselves into the game before they learn to identify with your hero—and you can't prevent this. Because of this constraint, you can save yourself endless pain and suffering by writing your narrative docs in first person, ditching all traces of omniscient voice, and by ensuring that when you say "I" in those docs, you mean the player more than you mean your hero.

Let's break it down.

GROUNDWORK

» To understand the differences between first-person narrative and other kinds of storytelling, it's necessary to thoroughly understand the process of projection that players go through when "becoming" a character in any game. More importantly, we must understand how this process is affected by the first-person camera.

Let's take it from the top. You [the player] boot the game, skip all the intro material, [haha] and select "new game." Perhaps there is an introduction to orient you ... and then you are in the game proper.

THIRD PERSON: CHARACTER FIRST

» Fade up. If the game is third person, what you will see first is your character, roughly center screen. What happens here?

Subconsciously, the player immediately starts to digest this hero as a person and establish a "mental model" of who this person is. How does he stand? What is he wearing? What is his body type? Can I see his face? What does his expression tell me? In just a few seconds, the player makes some judgments about who this guy is, and generates a mental model of that person. We're humans: it's what we do.

If we have played a game before, then we know at this moment that this character is "for us." He's our avatar. So, we begin to project ourselves into him.

He's probably a badass, so we project our personal would-be-badass self forward. If he's hurt or in trouble, we begin to take that on for ourselves. We add our agendas to this mix as we decide how we want to play.

Then (and this is where the magic happens), we push the left stick forward (or whatever input method is required), and we take control of this person. Click. Our projection is confirmed: now, he is us, and from this moment forward, we can be Link (or Dante, or whoever) any time we wish.

But what happens if the camera is first person?

FIRST PERSON: PLAYER FIRST

» Fade up. What is the first thing we see?

We see the world, through the window of the first-person camera. Maybe a pair of hands will be visible, but action certainly occurs before the player's eyes in some way, drawing her forward into the game; crucially, there is no hero on screen to model.

What happens here to the player?

First: the view is restricted. The player has a small view angle on the world, no more than 90 degrees. This is confining, and this unavoidable fact triggers the first emotion that every player of first-person games feels when they start in a new space: anxiety. What is behind me? What is to my left and right? Generally, the first action any player will take in first person is to move the view right and left to take a quick, orienting look around...

...and so, here, the reverse of the process described above happens: the player takes control before they build a model of who they are. However, they do project themselves forward into the game. Onto what? Who are they?

The answer is the key to understanding first person narrative constraints. It is no one. First-person players automatically, unconsciously, and inevitably project themselves into the game first, ahead of any process of developing empathy for the hero who they are playing.

I want to emphasize the "inevitably" above. I would very much like to be proven wrong about this point by some bright-spark genius designer,

and perhaps someday I shall be. But, for now, every single example of widely successful, engaging-to-most-people-who-play-it first-person narrative we have in our history is piloted by a semi-faceless hero who works better as a vehicle for the player's consciousness than as a target for curiosity and empathy. Tell me: does Gordon Freeman have a family? Desires of any kind? Did you ever care to ask such questions during play? Does he even have a voice?

In first person, it is the player who "starts" the experience. Like it or not, and regardless of whether you have a huge, expensive, universe-setting cinematic at the beginning, your first person experience will start with a "hero" who is very much like the player, and who is experiencing anxiety about what is to her right and left. Design accordingly.

Why does this matter? Because this is not how we have been taught to think about narratives by our decades of exposure to other mediums. In no other format does the hero automatically and inevitably begin as the viewer/reader/player, and the techniques we have been given to develop narratives from other formats were not developed with this constraint in mind. Some will work, but others will lead us horribly astray. We need to re-examine our techniques.

TECHNIQUES

» Well, okay, what do we do about that? First, it's not so bad. Many of the skills we have developed around characterization, such as world building, backstory, arc, metaphor—many of these still apply. It's not a disaster. But, what techniques must change?

WRITE IN FIRST PERSON

» Write all your narrative treatments, scripts, and story documents in first person. The first words in every first-person narrative document should be "I can see..." or "I can hear..."

Now, this might sound dumb to you. Over the years that I have taught these techniques to teams, I have found that, to some people, making such a fuss over changing "he" or "she" to "I"



ILLUSTRATION BY JUAN RAMIREZ

sounds really stupid. The response I have often heard goes something like this: "well, now that I have the story doc, I'll go write the first-person one for Jason, SIGH."

If that's you, well, my advice is that you get over it. It will save you from making potentially multi-million-dollar mistakes.

WRITE WITHOUT OMNISCIENCE

» The technique is not to just use "I," of course. The technique is to restrict yourself in your writing to the same tools the player will have to understand your story: what they see and hear. But specifically, switching to first person writing removes the most destructive part of game narrative design: the omniscient voice.

Game stories in general are difficult enough to tell without relying on the player's inner knowledge of your world and characters. Writing in the omniscient voice leads to situations where the narrative designer was expecting that "of course this scene will make sense," when in fact a key piece of knowledge is necessary to comprehend the plot in action ... and sometimes it is not possible (or is horribly clumsy) to get that information to the player.

These mistakes are fine if they are caught while in paper form. When they are discovered in playtest for a nearly complete level, the costs start to go up.

So, write in first person. And drop the omniscient voice.

WRITE AS THE BLENDED HERO

» When you write the word "I" in these documents, remember always that you are writing for what I refer to as the blended hero. That is, "I" means the player first and "I" also means your game's hero, as the player will experience him/her. The player and your hero must become blended in your mind.

What in the hell am I talking about? It's simpler than it sounds. Do you play first person games? When you explain to a friend about your exploits in DEUS EX, do you use the word "I"? I'll wager you do. And, do you use this "I" word to mean yourself *and* the hero? As in, "I walked up to the guy and asked him about the mission. But I didn't want to piss off his boss, so I was friendly." This is a "blended" statement in that you are referring both to your own player motivation and to the "hero" that you are playing.

Write as that blended hero.

HIT PUREE

» Your first-person walkthrough of your game's narrative should read something like this:

"I can see only a dim darkness. Whispered, frightened voices reach me—other people. There's a metal tone—are we in a container of some kind? I push forward, exploring—yup, it's about the right size for a container. The people are hard to make out, and they're speaking a language I don't understand.

With a CLANK, one end of the container splits open, letting in plenty of light. Rough-looking men with AKs and pistols push in, looking for someone. They spot me—uh-oh. I'm grabbed by the arm and pulled out of the darkness onto a beach..."

Remember, the closer what you write is to the eventual creation, the less trouble you will have along the way from narrative issues you overlooked. 🎮

JASON VANDENBERGHE has been making other people's games since 1996, and shows no signs of stopping any time soon. He's currently a creative-director-type at Ubisoft, but he has long, meaningful relationships (followed by dramatic, heart-wrenching breakups) with Activision and EA as well. He can be discovered and/or contacted at www.darklorde.com.

GAME DEVELOPER MAGAZINE

the best of
postmortems,
product reviews,
and standout
columns

gd

NOW AVAILABLE FOR
DIGITAL DOWNLOAD AND
FOR IOS DEVICES.

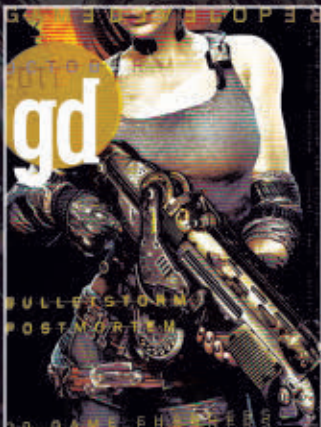


DOWNLOAD THE GAME
DEVELOPER APP

bit.ly/gdmag_ios

SUBSCRIBE TODAY!

WWW.GDMAG.COM/SUBSCRIBE





BUSINESS BOOKS FOR A TURBULENT TIME

GAUGING WHERE THE GAME BUSINESS IS GOING THROUGH LITERATURE

THE GAME INDUSTRY IS DOING EXTRAORDINARILY WELL BY ANY NUMBER OF MEASURES. There are record revenues, a broader number of people playing games, and developers have a choice of multiple conduits through which they can publish their games on a substantial and growing number of platforms. At the same time, it's precisely this number of choices and the rate of change that make it difficult to decide what to do next. Develop a Facebook game, or is that channel now closed to entrants? An iOS game, or is that space over saturated? What will be the Next Big Thing?

When trying to solve complex problems, we employ familiar tools like analogies (e.g., looking at similar circumstances in other industries) or employing models (systems that help us break down complex problems into simpler well-understood parts). It was with this in mind that I thought it a good time to offer a list of book recommendations that I've found helpful when trying to glean a view of where we may be heading.

I try to read a pretty wide range of material. The following are on a short list of books that I find myself regularly recommending to colleagues. In fact, the first one on my list is one I've purchased at least a dozen times in order to thrust it into the hands of those I strongly wanted to read it.

INSIDE THE TORNADO

Geoffrey Moore / Collins Business Essentials

Geoffrey Moore is probably best known for his "Crossing the Chasm" concept, and the book by the same name. However, *Inside the Tornado* (which boils the Chasm concept down to a single chapter) is a larger and more thorough work. I believe it to be one of the best collections of disruption and reinvention models in the technology industry.

While games aren't technology, games use technology. More importantly, their financial success often rides on the success of the platforms for which they are written. As new platforms emerge, Moore provides us with a framework to gauge whether to jump on board or wait for better opportunities.

THE MASTER SWITCH: THE RISE AND FALL OF INFORMATION EMPIRES

Tim Wu / Vintage

The Master Switch is a masterful work that looks at the history of the telephone, film, radio, and television industries and demonstrates how they all exemplified what he calls "The Cycle"—an inevitable drift from a dawn of open to a dusk of closed. Those raised in the age of the internet tend to believe that open wins in the end—but Wu's book reminds us just how resilient the incumbents are in trying to quash any threat that open platforms might represent.

INVENTING THE MOVIES: HOLLYWOOD'S EPIC BATTLE BETWEEN INNOVATION AND THE STATUS QUO, FROM THOMAS EDISON TO STEVE JOBS

Scott Kirsner / CreateSpace

Comparisons between movies and games are nothing new. If anything, they are too often made when speaking of the structure of the medium. However, this history of film and its industry offers a few perspectives that are somewhat analogous to elements of what we see in games.

First, it offers additional background and perspective on the transitions that the business side of films went through (for example, distributors with monopoly power exerting control over the creative side of the business).

Secondly, it offers unique perspective on the play between technology advancements, evolution of the medium, and effects each had (or didn't have) on both the medium and the business. Kirsner looks at the transition to color, talkies, and stereoscopy, to name a few.

MORAL PANICS AND THE COPYRIGHT WARS

William Patry / Oxford University Press

In a business steeped in discussions of intellectual property, piracy, game clones, and the like, it's helpful to have an understanding not just of the law, but of its evolution. William Patry is Google's senior copyright counsel and has written multiple highly technical works on copyright.

Moral Panics is a shorter (but not short!) and more layman-digestible work on copyright that takes a very centrist position and provides a fairly thorough history of copyright over the past few hundred years. It'll provide both the background and structured understanding of copyright issues to help the reader to better understand the issues at play.

LEAVE ROOM FOR FICTION!

» Business books can give us tools to make sense of the world around us and give us rules by which it might function and by which it might change. But they do a fairly poor job of understanding the innovations that may come to change how we interact with technology and in turn open new avenues for games. For this near-term futurism I like to turn to science fiction.

A few favorites I'll point to in brief here: **HALTING STATE** (Charles Stross / Ace) covers a broad range of game-related matter including MMOs, ARGs, Augmented Reality, real-money transactions, and more. **SUPER SAD TRUE LOVE STORY** (Gary Shteyngart / Random House) uses a rather dystopian view of the future of social networks and location-based apps as part of the backdrop for a romantic tragedy. **FOR THE WIN** (Cory Doctorow / Tor) covers MMOs from the perspective of self-organizing gold farmers who decide to unionize, and in the process touches on many of the issues involved with internationally spanning games-as-service offerings. All of these are entertaining works that will inspire reflection on the future of games.

It's hard to imagine where games will bring us in the next five years. I don't pretend that these, or any books can provide those answers. What they will do though, is provide you with tools and perspective to round out your view and better gauge the near term as you decide your next move.

Note: If you'd like more detail, I've provided longer reviews of each of these on my blog at www.kimpallister.com 📖

KIM PALLISTER works at Intel doing game industry forecasting and requirements planning. When not prepping the world for super-cool hardware, he blogs at www.kimpallister.com. His views in this column are his and do not reflect those of his employer.

GDC China 2011 Closes With Record Attendance

Organizers of the fourth annual GDC China, which concluded, November 14, 2011 in Shanghai, have announced that more than 3,000 international attendees were present over the three-day event, marking an all-time high for this branch of the Game Developers Conference.

This year, GDC China hosted more than 70 international speakers, more than 45 sponsors and exhibitors, as well as the third annual Independent Games Festival China, and once again gave game developers in Asia a chance to meet other industry professionals, gain practical experience, and share valuable ideas with their peers.

GDC China's main conference featured four tracks, covering global game development, online game development, and business and social gaming, alongside two specialized summits focusing on indie games and mobile games.

Game Developer sister site Gamasutra covered the event in-depth throughout the show, with featured talks including a keynote from Interplay veteran and inXile founder Brian Fargo, who examined the

history of role-playing games and offered his take on the most important elements of the genre.

Other standout talks from the show include Independent Game Summit talks with Capy cofounder Nathan Vella on the importance of creative risk, and CANABALI creator Adam Saltsman on key opportunities for indie developers.

In addition, the show featured notable talks from DeNA's Kenji Kobayashi on the [lack of] competition in the smartphone market, Amir Rao of BASTION dev Supergiant Games on leveraging new IP, and Microsoft's Brian Prince on the current trajectory of cloud gaming, among many others.

Within the coming weeks, lectures and slides from the show's numerous sessions are set to debut on the GDC Vault website (www.gdcvault.com), in both free and member-based tiers.

GDC China took place from Saturday, November 12 to Monday, November 14 at the Shanghai Exhibition Center in Shanghai, and is currently the only official developer event supported by the Ministry of Culture of the People's Republic of China.

2011 Independent Games Festival China Winners Announced



The winners have been announced for the indie showcase in Shanghai at GDC China, with Feng Li's 2D action beat-em-up PIXEL MAY CRY taking home the prize for best game, in addition to a host of other notable winners.

Following the announcement of the IGF China finalists in September, the selected teams attended a special awards show at the Shanghai Exhibition Center in November, where the winners took home a prestigious IGF award, and a cash prize ranging from RMB3,000 (\$450 USD) to RMB20,000 (\$3,060 USD).

Guest presenters from the independent games community on hand to help give out awards included GDC China Independent Games Summit speakers such as Amir Rao (BASTION), Baiyong (PIXELJUNK EDEN, 4AM) and Jenova Chen (FLOWER, JOURNEY).

Winners announced at the show include aBit Games' rhythm counting game SUPER SHEEP TAP, WitOne Games' fantasy RPG POCKET WARRIORS, and Ant Hive Games's THE LINE HD, which took home the award for best mobile game.

As the event took shape, IGF China received high-quality submissions from multiple Chinese provinces, Hong Kong, Taiwan, Macao, Korea, Singapore, Malaysia, Australia, New Zealand, India, Iran, and beyond.

The winners were chosen by a panel of expert jurors, including Kevin Li (CEO, TipCat Interactive), Monte Singman (CEO, Radiance Digital Entertainment), Xubo Yang (director of digital art lab and assistant professor at Shanghai Jiaotong University's School of Software), Haipeng Yu (producer, Tencent Shanghai), and jury chairman Simon Carless, IGF chairman emeritus and EVP of the GDC shows and Gamasutra. [@](#)

THE 2011 IGF CHINA WINNERS:

MAIN COMPETITION

BEST GAME: PIXEL MAY CRY, by Feng Li, China [RMB20,000 ¥ \$3,060 USD]

BEST MOBILE GAME: THE LINE HD, by Ant Hive Games, China [RMB10,000 ¥ \$1,530 USD]

EXCELLENCE IN AUDIO: SUPER SHEEP TAP, by aBit Games, China [RMB5,000 ¥ \$760 USD]

EXCELLENCE IN TECHNOLOGY: VOID, by DigiPen Institute of Technology, Singapore [RMB5,000 ¥ \$760 USD]

EXCELLENCE IN VISUAL ARTS: POCKET WARRIORS, by WitOne Games, China [RMB5,000 ¥ \$760 USD]

STUDENT COMPETITION

IGF CHINA BEST STUDENT GAME: VOID, by DigiPen Institute of Technology, Singapore [RMB10,000 ¥ \$1,530 USD]

IGF CHINA EXCELLENT STUDENT WINNER: PIXI, by DigiPen Institute of Technology, Singapore [RMB3,000 ¥ \$450 USD]

IGF CHINA EXCELLENT STUDENT WINNER: ROBOTANY, by Singapore-MIT GAMBIT Game Lab, Singapore [RMB3,000 ¥ \$450 USD]



Seattle Punch!

JAIME GRIESEMER MOVES FROM BUNGIE TO SUCKER PUNCH

Jaime Griesemer is primarily known for his design work on the HALO series, and is very attuned to timing and pacing in games. We caught up with him now that he's made the move to Sucker Punch, working on an as-yet-unannounced project.

Brandon Sheffield: *As a designer, have you felt a shift in thinking as you move from working on a first-person shooter to a third-person action game? Please elaborate on anything that might have changed for you there.*

Jaime Griesemer: Game design is game design. It doesn't matter if you are working on a first-person shooter or a turn-based puzzle game, the fundamental process is the same. Controls, mechanics, dynamics, progression, rewards—tuning and balancing and nailing the details. Camera perspective is just one factor among many, and a lot of the aiming tricks we used in HALO will apply just as well. The biggest difference is that in a third-person game it's easy for players to tell where they are getting shot from. I'm not going to miss dealing with that problem!

Professionally, designing first-person games was getting a little stale, anyway; it's great for aiming a gun but not much else. It'll be great to bring that expertise on precision aiming to a genre that has a lot more freedom of action and expression.

BS: *You seem to care a lot about timing as a designer. What are your thoughts on jump heights in that regard?*

JG: Timing is at the heart of interactivity. Certain cadences are more interesting, or fit specific types of gameplay. Buttons are just on/off switches, so timing is the only way to give them an analog range, which is where the depth comes in.

And timing has such an enormous impact on difficulty. I mean, anybody could handle Legendary HALO at half-speed, but even Chess gets harder with a strict time limit. TETRIS gets steadily faster until it gets pretty much impossible. The really competitive fighting game players have timing that is accurate to a single frame! That's why reliable performance is so important for AAA games; when your frame rate isn't consistent, neither is your gameplay.

BS: *What prompted the move to Sucker Punch?*

JG: When I joined the original HALO team we fit comfortably in a single, small room. On a team of that size, everybody has a chance to make a substantial impact on the game. I balanced the multiplayer, worked on the AI, scripted several

missions, and tuned all the weapons and vehicles. As the franchise grew and the games increased in scope, the team got huge. By the end I was working on such a narrow slice of the project ... it wasn't very satisfying and I needed to look for a new challenge. Turns out Bungie management had the same idea.

Sucker Punch is a much leaner organization. Staying small is a priority for them, so it's a natural fit. They have a long history of excellent games, but they haven't quite broken into that top-tier of game developers. The talent and the drive is there, they have all the pieces, so hopefully I can bring my experience and design focus and help them reach the next level. It's just the challenge I was looking for.

BS: *Have you felt any cultural change after the Sony acquisition, or was it a natural fit?*

JG: It's funny. I was at Bungie when they were acquired by Microsoft. We were worried because they had bought studios before and it hadn't gone well; they were a software company trying to learn the entertainment business. So we immediately put up a wall and isolated ourselves. It was a culture war and we were ready for a long siege. It was never a comfortable fit and the eventual split was probably inevitable.

The Sucker Punch acquisition was totally different. There's no cultural conflict because it's the same culture. Sony is an entertainment company and the internal studios are flourishing. The whole process has gone through without a hitch, really.

And personally, I love working for a platform holder. First MS and now Sony. A third-party publisher is risk-averse—they want the sure bet and are always looking at what the competition is doing—so that can have a chilling effect on the design. Not always, but it does happen. The great thing about working for a platform holder is that they want a unique game, something different from what everyone else is doing. They encourage well-considered risks because they want stand-out exclusives. As a designer, it's great to be able to take a chance on a bold design decision and not have to worry that the studio is going to go under if it doesn't pan out. 

who went where

Disney Interactive Media Group has hired OnLive VP for games and media John Spinale as its new senior vice president of social games, where he'll lead all aspects of the company's social business, including Playdom.

Redwood City-based RIFT developer Trion Worlds has announced new hirings as part of its move to expand overseas, including former EA senior manager of PR Jonathan Goddard, and Jeff Pabst, who previously served as head of international product management for Xbox.

Former Ubisoft Reflections managers Gareth and Martin Edmondson revealed that they have joined UK-based mobile games publisher Thumbstar Games.



Yuji Korekado.

Kojima Productions producer Kenichiro Imaizumi revealed that Shigenobu Matsuyama is no longer the producer on the upcoming METAL GEAR SOLID: RISING, with Yuji Korekado taking his place.

new studios

Stockholm, Sweden-based JUST CAUSE developer Avalanche Studios has announced the official opening of a new branch in New York City's SoHo district.

Tokyo-based Square Enix revealed plans for further expansion in Montreal, Canada, with the creation of a new studio dubbed Square Enix Montreal.

Nordic Games announced that it has relaunched PAINKILLER and SAFECRACKER publisher DreamCatcher as a separate publishing label under the Nordic Games name.



A CLOSED WORLD

<http://gambit.mit.edu/loadgame/aclosedworld.php>

IT'S RARE TO SEE GAMES WRESTLE WITH COMPLEX, AND PERHAPS CONTROVERSIAL SOCIAL ISSUES. A CLOSED WORLD, WHICH COMES FROM A TEAM OF RESEARCHERS AT THE SINGAPORE MIT GAMBIT GAME LAB, DOES JUST THAT, AND OPENLY EXPLORES LGBT AND QUEER THEMES IN A JRPG FORMAT. IN THE GAME, PLAYERS TAKE CONTROL OF A YOUNG PERSON WHO DECIDES TO ENTER HIS VILLAGE'S FORBIDDEN FOREST. ALONG THE WAY, THE PROTAGONIST MUST BATTLE INTERNAL DEMONS THAT QUESTION WHAT WE CONSIDER "NORMAL" AND WHAT IT MEANS TO BE TRUE TO ONESELF. WE SPOKE TO PRODUCT OWNER TODD HARPER TO LEARN MORE ABOUT THIS INTROSPECTIVE TITLE.

Tom Curtis: *Where did the inspiration for A CLOSED WORLD come from?*

Todd Harper: Last September in 2010 there were a number of queer youth and bullying victim suicides in the news, including the story of Tyler Clementi. The mission of GAMBIT, the lab where I work, is to use game design to do research, and my background is both in game studies and queer theory. So I thought, "How would a game deal with talking about this stuff?" The eventual end product didn't have all that much to do with that issue (though I hope to come back to it someday), but that was part of the spark. Another influence was a story in the October issue of *Edge* magazine called "Playing It Straight" (www.next-gen.biz/features/playing-it-straight) that dealt with the question of where all the LGBT/queer content in games was. A lot of inspiration came from reading some of the reasons the people they spoke to there gave and going, "Alright. So how can we overcome some of this?"

TC: *To what extent did the team draw from personal experience when creating the game?*

TH: This is a difficult question to answer. I know my own past experience went into discussions I had with the team about things like coming out and issues like that, as did one of the interns who identified as gay. I don't think that the game's content came as much from their actual personal experience as it did from the team making a big effort to come to grips with what the issues involved with being a queer person even are. They spent a lot



of time in the first few weeks just brainstorming and sharing ideas about what that meant.

The room they were in at GAMBIT has a glass wall that can be written on with dry erase marker. The team drew this amazing brainstorming thing that was dubbed The LGBT Spider, and it took up the entirety of the eight-foot high wall by the end of the project, having morphed into the LGBT Kraken. There's this big, amorphous concept of "the LGBT experience" that they—like any other dev team—had to wrap their heads around. We did a lot of mapping onto situations they did have experience with, and I think that worked out well.

TC: *In general, how have audiences responded to the title? Were you surprised by how people reacted?*

TH: Responses have run the gamut. There are players who really like everything, from story and gameplay on down. There are those who thought the gameplay was too simplistic, and the story was too cliché. And there's a lot of middle ground, which is the most common.

In general, feedback told us that the combat system was an interesting idea, but perhaps too simple, which I'd expect from something with an eight-week development cycle. Some people were big fans of the story, others though it was too simplistic—observe Anna Anthropy's parody game A CLOSED MIND (www.auntiepixelante.com/?p=1276) on that front. The really consistent thing, though, regardless of opinions on the game, is that this topic and this idea are important. Lots of people who weren't the biggest fans of the game still said "I'm glad someone tried this."

In all honesty, the tide of responses and coverage did take me by surprise; I expected a few mentions here and there, and that was all. But I think that sudden outpouring of interest says a lot about whether this was worth doing, and if it's worth it for other devs to try. I think it is. There's clearly a desire and an appreciation out there that needs to be addressed.

TC: *While the team seems to openly acknowledge A CLOSED WORLD's underlying themes, the game itself is fairly subdued and ambiguous. Was this a conscious decision on the team's part?*

TH: I'd say so. We had this constant battle with what we called "the hammer," where we come down with some top-heavy, word of God message about insert-LGBT-theme-here. Nobody likes that in a story or a game. Plus, our big conceit was this procedurally generated gender concept, where the genders of the people you meet (and in the backstory) are just picked for you at random. So, we needed a narrative that supported that, but that still got across the message we wanted to send, which is this

Team

TODD HARPER
Product Owner

ABE STIEN
Game Director

SOPHIA YUEN SHU HUI
Producer

LEX JOHNSON
QA Lead

PRAVEEN NAMASIVAYAM
Designer

BREE WESTPHAL
Artist

PETER TAN
Artist

KEVIN LAUGHLIN
Artist

TRAN HUNG NGUYEN
Programmer

JOVI TAN
Programmer

CASEY MERHIGE
Audio Designer

idea that being true to yourself is hard, and might require sacrifice, but in the end is a much better option than the alternative.

TC: *The game seems to focus more on its themes and message than it does gameplay itself. Why did you choose to emphasize narrative so heavily over the mechanics?*

TH: Before the summer we had a six-month prototyping period, where we worked over various ideas about the shape of the game. At first we really wanted to do all the work in a procedural way. We spent a lot of time looking at various parts of what a queer


individual's experience might be and saying "Okay, how can we turn this into a gameplay mechanic?" The real problem is we couldn't find one that was satisfying to us, and part of that was really just the huge amorphousness of grasping "a queer experience" in the first place.

Gameplay, a procedural thing, is about modeling ideas through [among other things] a system of rules ... but if we didn't have a compelling notion of "a queer experience," it'd be difficult to find a core mechanic that could express one. A question we'd constantly ask ourselves was "How do you establish a character, or a theme, or an idea as 'queer'

using primarily gameplay?" The answer we came to was that we couldn't ... not in the time allotted us, anyway. So in the end, we decided to let the narrative do the heavy lifting, ideologically speaking, and tried to focus on having the gameplay deliver and reinforce the narrative.

TC: *If you were able to go back and do one thing differently during the game's development, what would it be?*

TH: I'd come into the process having already worked out the raw mechanics of our combat system. I think developing the combat took a really huge amount

of our very limited time, and a lot of it was fine-tuning and tail-chasing of ideas that were inevitably scrapped for the sake of simplicity. The problem was that dealing with that "queer content" issue I described was also a time-consuming thing, and we couldn't set the story down until we had it resolved, and the later we settled the story, the more stress went onto coders and art/music asset creators both. I think if we'd had combat more or less settled and had more time to work on the game's portrayal and packaging of the story and message, we could have added some additional nuance to it in the end. 



gd

GAME DEVELOPER MAGAZINE

the best of postmortems, product reviews, and standout columns

GET THE **PRINT+DIGITAL ACCESS BUNDLE FOR ONLY \$49.95 /YEAR**

- + DIGITAL ACCESS TO BACK ISSUES
- + EXCLUSIVE INTERACTIVE EXTRAS

INCLUDES:

- PRINT SUBSCRIPTION
- DIGITAL + GAME DEVELOPER APP

BONUS!

- BEST OF POSTMORTEMS PRINT ISSUE

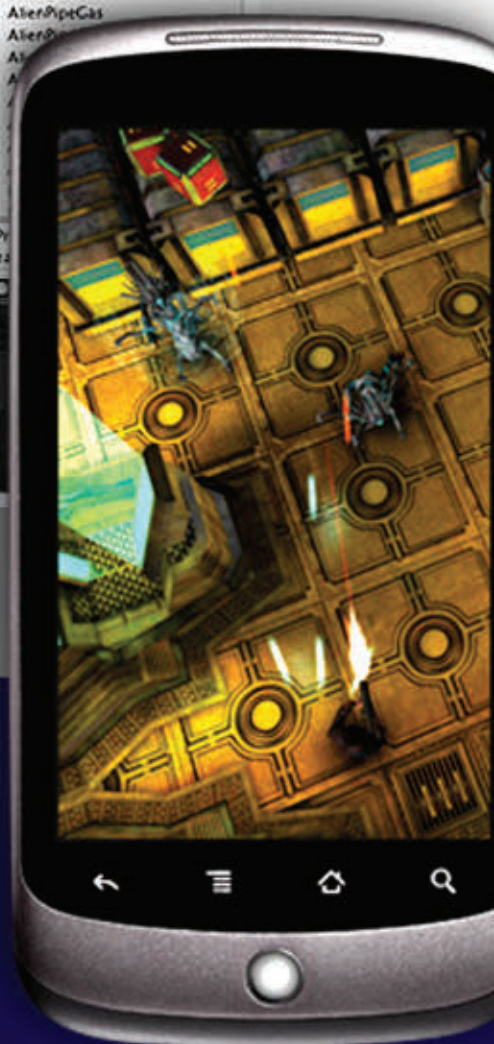
SUBSCRIBE TODAY! GDMAG.COM/SUBSCRIBE



Now Available from Studica.com at Academic Prices



Student Offer!
\$99*



Create Amazing Games

Unity 3 is a game development tool that is designed to let you focus on creating amazing games. Students and educators can now purchase Unity products at Studica.com!

- Special Student Offer \$99*
- Classroom Licenses
- Student Full Commercial Version

*Limited time offer

www.studica.com/GDM



Studica
The Education Source for
Software & Technology Products

DO AS WE SAY, AND AS WE DO.

It's simple. Creating art for video games is our day job, teaching you is our night job. Don't you think learning from artists who actually work in the industry is a good idea?

We do.



Student Work by Eric Wiley

FUTUREPOLY | Relevant Education | www.futurepoly.com

TURN YOUR PASSION FOR GAMING INTO A CAREER

Game Art

Bachelor's Degree Program
Campus & Online

Game Design

Master's Degree Program
Campus

Game Development

Bachelor's Degree Program
Campus

Game Design

Bachelor's Degree Program
Online



Campus Degrees

Master's

- Entertainment Business
- ▶ Game Design

Bachelor's

- Computer Animation
- Creative Writing for Entertainment
- Digital Arts & Design
- Entertainment Business
- Film
- ▶ Game Art
- ▶ Game Development
- Music Business
- Recording Arts
- Show Production
- Sports Marketing & Media
- Web Design & Development

Associate's

- Graphic Design
- Recording Engineering

Online Degrees

Master's

- Creative Writing
- Education Media Design & Technology
- Entertainment Business
- Internet Marketing
- Media Design
- New Media Journalism

Bachelor's

- Computer Animation
- Creative Writing for Entertainment
- Digital Cinematography
- Entertainment Business
- ▶ Game Art
- ▶ Game Design
- Graphic Design
- Internet Marketing
- Mobile Development
- Music Business
- Music Production
- Sports Marketing & Media
- Web Design & Development



FULL SAIL
UNIVERSITY.

fullsail.edu

Winter Park, FL

800.226.7625 • 3300 University Boulevard

Financial aid available to those who qualify • Career development assistance • Accredited University, ACCSC
To view detailed information regarding tuition, student outcomes, and related statistics,
please visit fullsail.edu/outcomes-and-statistics.

PREPARING THE NEW LEADERS OF THE DIGITAL MEDIA INDUSTRY

The Masters of Digital Media program (MDM) is Canada's first professional graduate degree program of its kind in digital media and entertainment technology.

Offered at Vancouver's Centre for Digital Media, this 16-month program and internship engages students in real world projects where they gain valuable leadership experience, hands-on training, and top industry connections.

Find out more. mdm.gnwc.ca/leaders

**CENTRE FOR DIGITAL MEDIA
MASTERS OF DIGITAL MEDIA PROGRAM**



Start Living The Dream!

A.S. Degree in Game Production



Learn to create the future of games with an Associate's Degree in Game Production from The Los Angeles Film School. Your education will give you the knowledge to view every piece of a game artistically, analyze its programming and learn the tools & techniques to create the worlds we play every day.

Learn the Science of Game Production and have the following skill set:

- Create Game Art
- Design Characters, Objects & Environments
- Develop Game Programming Skills
- Discover the Art of Storytelling

Learn from The Los Angeles Film School's experienced industry professionals.

- On-site housing coordinator
- Accredited College, ACCSC, VA-Approved
- Financial Aid & Military Education Benefits (including BAH) available to those who qualify



THE
LOS ANGELES[®]
FILM SCHOOL

ANIMATION + AUDIO + FILM + GAMES

Create Your Future Today. Call:

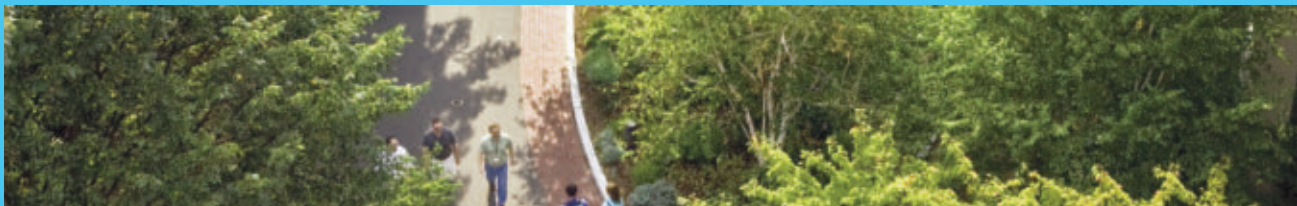
800.406.7485

www.DesignLAFilm.com



*Length of program and start dates are dependent on course of study and degree option. For more information on our programs and their outcomes visit www.lafilm.edu/disclosures.

©2011 The Los Angeles Film School. All rights reserved. The term "The Los Angeles Film School" and The Los Angeles Film School logo are either service marks or registered service marks of The Los Angeles Film School. Accredited by ACCSC



Creative Industries Game Design & Interactive Media

Collaborate
and create in
the heart of
Boston

Northeastern University
360 Huntington Avenue
Boston, MA 02115-5000

www.northeastern.edu/ci

Check us out on 

Our Creative Industries program creates, fosters and implements digital media innovation.

Through research, education and groundbreaking team-based interdisciplinary projects our students and faculty challenge each other with one single goal.

Excellence.

Create your own Combined Major or choose one of our most popular.

BFA in Game Design / Digital Arts
BFA in Game Design / Graphic Design
BS in Game Design / Computer Science

BFA in Interactive Media / Digital Arts
BFA in Interactive Media / Graphic Design
BS in Interactive Media / Computer Science
BS in Interactive Media / Music Technology

Creative Industries Minor



Northeastern
College of Arts, Media & Design

ADVERTISER INDEX

COMPANY NAME	PAGE	COMPANY NAME	PAGE
AUTODESK.....	C2	PERFORCE SOFTWARE.....	38
EPIC GAMES.....	29	RAD GAME TOOLS.....	C4
FULL SAIL REAL WORLD EDUCATION.....	61	SCOTTISH DEVELOPMENT	
FUTUREPOLY INC.....	61	INTERNATIONAL.....	62
HANDELABRA STUDIO LTD.....	3	SOCIETY FOR THE ADVANCEMENT OF	
HAVOK.....	30	SCIENCE OF DIGITAL GAMES.....	12
LOS ANGELES FILM SCHOOL.....	62	STUDICA.....	60
MASTERS OF DIGITAL MEDIA PROGRAM..	62	TECHEXCEL.....	6
NORTHEASTERN UNIVERSITY.....	63	VANCOUVER FILM SCHOOL.....	37

gd Game Developer (ISSN 1073-922X) is published monthly by UBM LLC, 303 Second Street, Suite 900 South, South Tower, San Francisco, CA 94107, (415) 947-6000. Please direct advertising and editorial inquiries to this address. Canadian Registered for GST as UBM LLC, GST No. R13288078, Customer No. 2116057, Agreement No. 40011901. **SUBSCRIPTION RATES:** Subscription rate for the U.S. is \$49.95 for twelve issues. Countries outside the U.S. must be prepaid in U.S. funds drawn on a U.S. bank or via credit card. Canada/Mexico: \$69.95; all other countries: \$99.95 (issues shipped via air delivery). Periodical postage paid at San Francisco, CA and additional mailing offices. **POSTMASTER:** Send address changes to Game Developer, P.O. Box 1274, Skokie, IL 60076-8274. **CUSTOMER SERVICE:** For subscription orders and changes of address, call toll-free in the U.S. (800) 250-2429 or fax (847) 647-5972. All other countries call (1) (847) 647-5928 or fax (1) (847) 647-5972. Send payments to *gd Game Developer*, P.O. Box 1274, Skokie, IL 60076-8274. Call toll-free in the U.S./Canada (800) 444-4881 or fax (785) 838-7566. All other countries call (1) (785) 841-1631 or fax (1) (785) 841-2624. Please remember to indicate *gd Game Developer* on any correspondence. All content, copyright *gd Game Developer* magazine/UBM LLC, unless otherwise indicated. Don't steal any of it.



HELPDEV, THE FRIENDLY PUBLISHER

WE'RE THE FRIENDLIEST, MOST HELPFUL PUBLISHER AROUND!

Hello there, game developers! I'm a representative from HelpDev, the Friendly Publisher! Here at HelpDev, we love to help developers—and we're friendly!

"What can HelpDev do for me?" you ask. "After all, there are plenty of other publishers out there. Not to mention the fact that it's never been easier for developers to bring their games to market themselves."

Well, I'd be happy to explain how wrongheaded that line of thinking is. First of all, without my help, putting your game out there into the big, mean, scary world is dooming it to die miserably! It's the truth! "Why, developer? Why? I loved you!" your game will cry as it is pecked to death by horrible screeching ravens. You don't want that! This is why you need a publisher like us to protect your game so that it gets the warm, supporting, and caring treatment that it deserves.

And HelpDev is not like those unscrupulous other publishers that will take your precious hard work and exploit it for their own gain. We're different! We genuinely respect all the blood, sweat, and Mountain Dew you developer guys pour into your games. And we want you to be successful on your own terms. In fact, you might not have noticed this earlier, but HelpDev is actually a combination of the words "help" and "dev." With a name like that, how could you not trust us?

We have a load of concrete benefits and game industry expertise to help you achieve success in the barren post-apocalyptic landscape of today's game market. Here's just a sampling of them!

FUNDING

Since you're a tiny little game developer, you may require funding

to complete your game. That happens sometimes. Not all of us are independently wealthy, after all ... not yet, anyway! But you're in luck: now you've got a publisher friend who just loves to fund interesting new games! I especially love to fund them if the game is close to done and just needs that extra \$1–2 in postage to get that master disc to our office!

Ha-ha. I'm just joking, of course ... but really, there's so much more



Taquitos.

to funding than just monetary amounts. Getting things done in this business is all about who you know, your connections—your guy on the inside. Know what I mean? And you'll be happy to know that I have great relationships with all of the big players in the industry. There are countless examples, such as our discreet, very highly placed and powerful contacts at Apple that might help get your game on the iTunes App Store! Apple doesn't just let anyone onto the store. This is a big deal!

MARKETING

We will ensure that your game gets the attention it deserves. HelpDev has a battery of top-flight, finely honed marketing minds ready

to unleash at a moment's notice on whatever we decide to sell. These guys know all about CPMs and stuff, and regularly purchase advertising on many major gaming web sites, such as [bobsgameblog.info](#) and [members.geocities.com/totallyunbiasedgamereviews](#). We also have a Twitter account!

Another benefit of working with HelpDev is our brand. We've spent weeks establishing our brand in gamer circles and cultivating

that and still take the occasional break for some tasty Doritos and Hot Pockets? Am I right, friend?

HelpDev's got your back. We can figure out how to successfully monetize and license your work, making even more money from sources you might have never imagined before. There are all kinds of exciting new platforms to think about these days, like those little kiosks they have in bars or the back seats of taxi cabs. Did you know they can even put video games on T-shirts now? And have you thought about translating your game into another language, like French or Australian? It's truly a global market these days.

Gamers also love licensed goods! Beach towels, drinkware, and of course, the requisite Slurpee flavor. Imagine drinking the Slurpee flavor of your own game as you work on it! Mind blowing.

So yeah, one last little thing ... you're probably wondering about IP rights. And as a smart developer, you *should* be wondering about that! We here at HelpDev are happy to say that we are more than happy to let developers happily retain the rights to their own IP! All we ask is for the *rights* to those IP rights!

I see those little gears turning in your head right now, and I'm going to ask you to hold on a second. Stop that right now. Visualize what you'll do with that giant pile of money you're about to make with us. Buy a new computer, maybe? Get your WORLD OF WARCRAFT character power-leveled? Do an epic Costco run for frozen taquitos? Imagine the possibilities! The future is a star-filled wonderland!

Just sign here. ☺

MATTHEW WASTELAND writes about games and game development at his blog, *Magical Wasteland* ([www.magicalwasteland.com](#)). Email him at [mwasteland@gdmag.com](#).



Scotland. Home to software innovation and highly skilled graduates.

We could teach
you a thing or two.

We've got quite a reputation for education, technology and invention. With the highest concentration of universities in Europe, we have produced innovations from the ATM to MRI scanning. Our education system has been the platform for our contribution to the world. It has helped build one of the world's most competitive, reliable and dynamic technology infrastructures, with particular strengths in implementing intelligence in to communication

and software technologies. Whether it's informatics, cloud computing or games development, we strive to set standards. Our passion for success and hunger to win, combined with our world-class academic institutions, outstanding research and superb facilities make Scotland truly irresistible location. We can develop your products and help shape your business. And that's what makes Scotland such a secure investment.

To see what we can do for your business, visit www.sdi.co.uk

SCOTLAND. SUCCESS LIKES IT HERE.





THERE'S NOTHING LIKE BEING A RAD GAME DEVELOPER



It feels like you have **SUPER POWERS** when you use Bink, our amazing, super **FAST** video and audio codec.



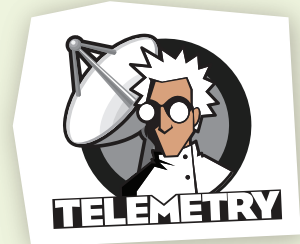
You'll **LOVE** using Granny, our run-time dynamic **3D ANIMATION SYSTEM**.



There's nothing like Miles, our multichannel sound system.



And our newest tool, Telemetry, is a library for profiling, tuning and visualizing application **PERFORMANCE**



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

Well, nothing except having a killer



www.radgametools.com
425-893-4300